

Le package `lymath` : des formules plus sémantiques

Code source disponible sur <https://github.com/bc-latex/ly-math>.

Version 0.7.0-beta développée et testée sur Mac OS X.

Christophe BAL

2020-06-08

Table des matières

1	Introduction	6
2	Comment lire cette documentation ?	6
3	A propos des macros	6
3.1	Règles de nommage	6
3.1.1	Les macros de même « type »	6
3.1.2	Les macros standards redéfinies	6
3.1.3	Les macros en mode <code>displaystyle</code>	7
3.2	Versions étoilées	7
3.3	Les arguments : deux conventions à connaître	7
3.3.1	Avec un nombre fixé d'arguments	7
3.3.2	Avec un nombre variable d'arguments	7
4	Couleurs	7
5	Deux séparateurs d'arguments par défaut	7
6	Quelques gestions d'espaces	7
6.1	Espace et fraction	7
6.2	Espace et racines n-ièmes d'un réel	8
6.3	Sommes et produits en mode ligne	8
6.4	Espace et point-virgule avec l'option <code>french</code> de <code>babel</code>	8
7	Logique et fondements	8
7.1	Différents types d'égalités « standard »	8
7.1.1	Définir quelque chose	8
7.1.2	Indiquer une identité	8
7.1.3	Une égalité à vérifier ou non, une hypothèse, une condition	9
7.1.4	Une égalité indiquant le choix d'une valeur	9

7.1.5	Une égalité indiquant l'équation d'une courbe que l'on utilise	9
7.1.6	Différents types d'inéquations	9
7.1.7	Une table récapitulative	9
7.1.8	Textes utilisés	10
7.2	Équivalences et implications	10
7.2.1	Des symboles supplémentaires	10
7.2.2	Une table récapitulative	10
7.2.3	Équivalences et implications verticales	10
7.3	Tables des décorations possibles des opérateurs	11
7.4	Détailler un raisonnement – Version pour le lycée et après	11
7.5	Détailler un raisonnement – Version pour les collégiens	13
7.6	Des versions alternatives du quantificateur \exists	16
8	Ensembles et applications	16
8.1	Différents types d'ensembles	16
8.1.1	Ensembles versus accolades	16
8.1.2	Ensembles pour la géométrie	17
8.1.3	Ensembles probabilistes	17
8.1.4	Ensembles pour l'algèbre générale	17
8.2	Ensembles classiques en mathématiques et en informatique théorique	18
8.2.1	La liste complète	18
8.2.2	Ensembles classiques suffixés	18
8.3	Des suffixes à la carte	19
8.4	Intervalles	19
8.4.1	Intervalles réels - Notation française (?)	19
8.4.2	Intervalles réels - Notation américaine	19
8.4.3	Intervalles discrets d'entiers	19
8.5	Unions et intersections	20
8.6	Cardinal, image et compagnie	20
8.7	Application totale, partielle, injective, surjective et/ou bijective	21
9	Géométrie	22
9.1	Points et lignes	22
9.1.1	Points	22
9.1.2	Lignes	22
9.1.3	Droites parallèles ou non	23
9.2	Vecteurs	23
9.2.1	Les écrire	23
9.2.2	Norme	23
9.2.3	Produit scalaire – Écriture minimaliste	24
9.2.4	Produit scalaire – Écriture « à la physicienne »	24
9.2.5	Produit vectoriel	24
9.2.6	Plan – Déterminant de deux vecteurs	25
9.3	Coordonnées	25
9.4	Nommer un repère	26
9.5	Arcs circulaires	27
9.6	Angles	27
9.6.1	Angles géométriques « intérieurs »	27
9.6.2	Angles orientés de vecteurs	28

10	Analyse	29
10.1	Constantes	29
10.1.1	Constantes classiques	29
10.1.2	Constantes latines personnelles	29
10.2	La fonction valeur absolue	29
10.3	Fonctions nommées spéciales	29
10.3.1	Sans paramètre	29
10.3.2	Avec un paramètre	30
10.3.3	Toutes les fonctions nommées en plus	30
10.4	Des notations complémentaires pour des suites spéciales	30
10.5	Calcul différentiel	30
10.5.1	Les opérateurs ∂ et d	30
10.5.2	Dérivation totale	30
10.5.3	Dérivation partielle	31
10.6	Calcul intégral	32
10.6.1	L'opérateur crochet – 1 ^{ère} version	32
10.6.2	L'opérateur crochet – 2 ^{nde} version	32
10.6.3	Intégrales multiples	33
10.7	Tableaux de variation et de signe	33
10.8	Comparaison asymptotique de suites et de fonctions	34
10.8.1	Les notations \mathcal{O} et \mathcal{o}	34
10.8.2	La notation Ω	35
10.8.3	La notation Θ	35
11	Probabilité	35
11.1	Probabilité « simple »	35
11.2	Probabilité conditionnelle	35
11.3	Espérance	36
11.4	Arbres pondérés	36
12	Arithmétique	39
12.1	Opérateurs de base	39
12.2	Fractions continuées	39
12.2.1	Fractions continuées standard	39
12.2.2	Fractions continuées généralisées	39
12.2.3	Comme une fraction continuée isolée	40
12.2.4	L'opérateur \mathcal{K}	40
13	Algèbre	41
13.1	Polynômes, séries formelles et compagnie	41
13.1.1	Polynômes et fractions polynômiales	41
13.1.2	Séries formelles et leurs corps de fractions	41
13.1.3	Polynômes de Laurent et séries formelles de Laurent	41
13.2	Matrices	42
13.2.1	Calculs expliqués des déterminants 2×2	42
13.2.2	Quelques exemples pour bien démarrer	42
14	Historique	45

15 Toutes les fiches techniques	48
15.1 Deux séparateurs d'arguments par défaut	48
15.2 Quelques gestions d'espaces	48
15.2.1 Fractions	48
15.2.2 Racines n-ièmes	48
15.2.3 Sommes et produits	48
15.3 Logique et fondements	48
15.3.1 Les textes pour les opérateurs de « comparaison algébrique » et de logique . . .	48
15.3.2 Les opérateurs de « comparaison algébrique »	49
15.3.3 Les opérateurs de logique	50
15.3.4 Les opérateurs de logique « verticaux »	51
15.3.5 Les textes pour détailler un raisonnement – Version pour le lycée et après . . .	51
15.3.6 Détailler un raisonnement – Version pour le lycée et après	51
15.3.7 Détailler un raisonnement – Version pour les collégiens	51
15.3.8 Des versions alternatives du quantificateur \exists	52
15.4 Ensembles et applications	52
15.4.1 Ensembles versus accolades	52
15.4.2 Ensembles pour la géométrie	52
15.4.3 Ensembles probabilistes	52
15.4.4 Ensembles pour l'algèbre générale	52
15.4.5 Ensembles classiques	53
15.4.6 Des suffixes à la carte	54
15.4.7 Intervalles réels - Notation française (?)	54
15.4.8 Intervalles réels - Notation américaine	54
15.4.9 Intervalles discrets d'entiers	55
15.4.10 Unions et intersections	56
15.4.11 Cardinal, image et compagnie	56
15.4.12 Application totale, partielle, injective, surjective et/ou bijective	56
15.5 Géométrie	56
15.5.1 Points	56
15.5.2 Lignes	56
15.5.3 Droites parallèles ou non	57
15.5.4 Vecteurs	57
15.5.5 Norme	57
15.5.6 Produit scalaire – Écriture minimaliste	57
15.5.7 Produit scalaire – Écriture « à la physicienne »	57
15.5.8 Produit vectoriel	58
15.5.9 Plan – Déterminant de deux vecteurs	58
15.5.10 Coordonnées	59
15.5.11 Nommer un repère	59
15.5.12 Arcs circulaires	60
15.5.13 Angles géométriques « intérieurs »	60
15.5.14 Angles orientés de vecteurs	60
15.6 Analyse	60
15.6.1 Constantes classiques	60
15.6.2 Constantes latines personnelles	61
15.6.3 Sans paramètre	61
15.6.4 Avec un paramètre	61
15.6.5 Des suites spéciales	62
15.6.6 Calcul différentiel	62

15.6.7	Dérivation totale	62
15.6.8	Dérivation partielle	63
15.6.9	L'opérateur crochet – 1 ^{ère} version	63
15.6.10	L'opérateur crochet – 2 ^{nde} version	63
15.6.11	L'opérateur d'intégration standard	63
15.6.12	Les notations \mathcal{O} et \mathcal{o}	63
15.6.13	La notation $\mathbf{\Omega}$	63
15.6.14	La notation $\mathbf{\Theta}$	64
15.7	Probabilité	64
15.7.1	Probabilité « simple »	64
15.7.2	Probabilité conditionnelle	64
15.7.3	Espérance	64
15.7.4	Arbres pondérés	64
15.8	Arithmétique	64
15.8.1	Opérateurs de base	64
15.8.2	Fractions continuées standard	65
15.8.3	Fractions continuées généralisées	65
15.8.4	Comme une fraction continuée isolée	65
15.8.5	L'opérateur \mathcal{K}	65
15.9	Algèbre	65
15.9.1	Polynômes, séries formelles et compagnie	65
15.9.2	Calculs expliqués des déterminants 2×2	65

1 Introduction

L^AT_EX est un excellent langage, pour ne pas dire le meilleur, pour rédiger des documents contenant des formules mathématiques. Malheureusement toute la puissance de L^AT_EX permet d'écrire des codes très peu sémantiques. Le modeste but du package `lymath` est de fournir quelques macros sémantiques pour la rédaction de formules mathématiques élémentaires. Considérons le code L^AT_EX suivant.

```
Sachant que  $\frac{df}{dx}(x) = \cos(x^2)$  sur  $[a ; b]$  , nous avons :  
 $\int_a^b \cos(x^2) dx = \left[ f(x) \right]_a^b$ .
```

Avec `lymath`, vous pouvez écrire le code suivant.

```
Sachant que  $\frac{df}{dx}(x) = \cos(x^2)$  sur  $\text{intervalC}\{a\}\{b\}$ , nous avons :  
 $\int_a^b \cos(x^2) \, dd{x} = \text{hook}\{f(x)\}\{a\}\{b\}$ .
```

Même si certaines commandes sont plus longues à écrire que ce que permet L^AT_EX, il y a trois avantages à utiliser des commandes sémantiques.

1. La mise en forme dans votre document devient consistante.
2. Il est facile de changer une mise en forme sur l'ensemble d'un document.
3. `lymath` résout certains problèmes "complexes" pour vous.

2 Comment lire cette documentation ?

Le choix a été fait de fournir des exemples comme documentation du package suivis de fiches techniques des macros-commandes (*voir la section 15*). Les exemples se présentent comme ci-dessous.

```
 $\displaystyle$   
 $\int_a^b \cos(x^2) dx$   
 $= \left[ f(x) \right]_a^b$ 
```

$$\int_a^b \cos(x^2) dx = [f(x)]_a^b$$

3 A propos des macros

3.1 Règles de nommage

3.1.1 Les macros de même « type »

Les macros partageant une même fonctionnalité mathématique suivent les deux règles suivantes.

1. Un gros préfixe est utilisé : par exemple, `\derpow` et `\derfrac` sont des macros pour rédiger des dérivées de fonctions¹.
2. Un mini préfixe permet d'indiquer une spécialisation d'une macro : par exemple, `\nparallel` est la version négative de `\parallel` (*le mini préfixe **n** est pour **n**-ot soit « non » en anglais*).

3.1.2 Les macros standards redéfinies

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

1. Ce choix est assumé même si pour les macros du type `\set...` on obtient un nom faisant penser à « régler ... » au lieu de « ensemble de type ... ».

3.1.3 Les macros en mode `displaystyle`

Les macros évitant d'avoir à taper `\displaystyle` auront un nom commençant par la lettre d.

3.2 Versions étoilées

Les versions étoilées proposent des mises en forme correspondant aux cas les moins usuels ou les moins pratiques : par exemple, la macro `\derpow` utilise un exposant entre des parenthèses pour rédiger une dérivée n^e tandis que la version étoilée, si c'est possible, affichera des primes en exposant.

3.3 Les arguments : deux conventions à connaître

3.3.1 Avec un nombre fixé d'arguments

Dans ce cas, c'est la syntaxe L^AT_EX usuelle qui sera à utiliser comme dans `\derfrac{f}{x}`.

3.3.2 Avec un nombre variable d'arguments

Certaines macros offrent la possibilité de fournir un nombre variable d'arguments comme dans `\coord{x | y | z | t}` et `\coord{x | y}`. Ceci se fait en utilisant un seul argument, au sens de L^AT_EX, dont le contenu est formé de morceaux séparés par des traits verticaux `|`. Ainsi dans `\coord{x | y | z | t}`, l'unique argument `x | y | z | t`, au sens de L^AT_EX, sera analysé par `lymath` comme étant formé des quatre arguments `x`, `y`, `z` et `t`.

4 Couleurs

Certaines macros utilisent de la couleur. Les choix faits sont tels que l'impression en noir et blanc ne soit pas impactée.

5 Deux séparateurs d'arguments par défaut

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

6 Quelques gestions d'espaces

6.1 Espace et fraction

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.

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

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

6.2 Espace et racines n-ièmes d'un réel

`\sqrt` a été redéfini pour ajouter un peu d'espaces. Le comportement par défaut se retrouve en utilisant la macro `\stdsqrt`. Voici un exemple.

```
\sqrt{2} = \stdsqrt{2}$ ,  
\sqrt[n]{45} = \stdsqrt[n]{45}$
```

$$\sqrt{2} = \sqrt{2} , \sqrt[n]{45} = \sqrt[n]{45}$$

6.3 Sommes et produits en mode ligne

Pour limiter l'espace, L^AT_EX affiche $\sum_{k=0}^n$ et non $\sum_{k=0}^n$ sauf si l'on utilise la commande `\displaystyle`. Les macros `\dsum` et `\dprod` permettent de se passer de `\displaystyle`. Voici un exemple.

```
\dsum_{k=0}^n 2^k =  
\sum_{k=0}^n 2^k$
```

$$\sum_{k=0}^n 2^k = \sum_{k=0}^n 2^k$$

```
\dprod_{k=1}^n k =  
\prod_{k=1}^n k$
```

$$\prod_{k=1}^n k = \prod_{k=1}^n k$$

6.4 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 dans $A(x;y)$. Que c'est beau!

7 Logique et fondements

7.1 Différents types d'égalités « standard »

D'un point de vue pédagogique, il peut être intéressant de disposer de différentes façon d'écrire une égalité, une non égalité ou une inégalité. Bien entendu on tord les règles de typographie avec ce type de pratique mais c'est pour le bien de la communauté.

7.1.1 Définir quelque chose

L'exemple suivant montre trois façons de rédiger une égalité signifiant une définition² (la section 7.1.8 explique comment est défini le texte « déf »).

```
$f(x) \eqdef x^3 + 1$ ou  
$f(x) \eqdef* x^3 + 1$
```

$$f(x) \stackrel{\text{déf}}{=} x^3 + 1 \text{ ou } f(x) := x^3 + 1$$

7.1.2 Indiquer une identité

L'exemple suivant montre deux façons de rédiger des identités, la notation symbolique n'étant pas standard (la section 7.1.8 explique comment est défini le texte « id »).

2. Le symbole peu courant $:=$ est utilisé par le langage B qui permet de spécifier et prouver certains programmes.

`$(a + b)^2 \eqid a^2 + b^2 + 2 a b$ ou
$(a + b)^2 \eqid* a^2 + b^2 + 2 a b$.`

$$(a+b)^2 \stackrel{\text{id}}{=} a^2+b^2+2ab \text{ ou } (a+b)^2 \stackrel{\text{id}}{=} a^2+b^2+2ab$$

7.1.3 Une égalité à vérifier ou non, une hypothèse, une condition

Se reporter à la section 7.1.8 pour savoir comment sont définis les textes « *cond* » et « *hyp* ».

`$(a + b)^3 \eqtest a^3 + b^3 + 3 a b$`

`$(a + b)^3 \neqid a^3 + b^3 + 3 a b$`

`$x \neqhyp 0$ ou $x \neqcond 0$`

`$x \eqcond 0$ ou $x \eqhyp 0$?`

$$(a+b)^3 \stackrel{?}{=} a^3 + b^3 + 3ab$$

$$(a+b)^3 \stackrel{\text{id}}{\neq} a^3 + b^3 + 3ab$$

$$x \stackrel{\text{hyp}}{\neq} 0 \text{ ou } x \stackrel{\text{cond}}{\neq} 0$$

$$x \stackrel{\text{cond}}{=} 0 \text{ ou } x \stackrel{\text{hyp}}{=} 0 ?$$

7.1.4 Une égalité indiquant le choix d'une valeur

La section 7.1.8 permet de savoir comment le texte « *choix* » est défini.

`$x \geqcond 4$ implique
$x^2 \geqcons 16$.`

`Alors $x \eqchoice 123$ donne
$123^2 \eqappli 16$.`

$$x \stackrel{\text{cond}}{\geq} 4 \text{ implique } x^2 \stackrel{\text{cons}}{\geq} 16.$$

$$\text{Alors } x \stackrel{\text{choix}}{=} 123 \text{ donne } 123^2 \stackrel{\text{appli}}{\geq} 16.$$

7.1.5 Une égalité indiquant l'équation d'une courbe que l'on utilise

La section 7.1.8 permet de savoir comment les textes « *graph* » et « *appli* » sont définis (la macro `\setgeo` est définie dans la section 8.1.2).

`$M \in \setgeo{C}: y \eqplot x^2 + 3$
donne
$y_M \eqappli x_M^2 + 3$.`

$$M \in \mathcal{C} : y \stackrel{\text{graph}}{=} x^2 + 3 \text{ donne } y_M \stackrel{\text{appli}}{=} x_M^2 + 3.$$

7.1.6 Différents types d'inéquations

Le principe reste le même pour les symboles d'équations excepté qu'il n'y a ici aucune écriture purement symbolique. Voici un code « fourre-tout » montrant quelques exemples.

`$x \leqtest x^2$ ou $x \lttest x^2$ ou
$x \geqhyp 1$ ou $x \gcond 2$.`

$$x \stackrel{?}{\leq} x^2 \text{ ou } x \stackrel{?}{<} x^2 \text{ ou } x \stackrel{\text{hyp}}{\geq} 1 \text{ ou } x \stackrel{\text{cond}}{>} 2.$$

7.1.7 Une table récapitulative

La table 1 page 11 fournit toutes les associations autorisées entre opérateurs de comparaison et décorations.

7.1.8 Textes utilisés

Voici les macros définissant les textes utilisés qui tiennent compte de l'utilisation ou non de l'option `french` de `babel`. Nous ne donnons que les versions françaises.

<code>\textopappli</code> donne « <i>appli</i> »	<code>\textophyp</code> donne « <i>hyp</i> »
<code>\textopchoice</code> donne « <i>choix</i> »	<code>\textopid</code> donne « <i>id</i> »
<code>\textopcond</code> donne « <i>cond</i> »	<code>\textopplot</code> donne « <i>graph</i> »
<code>\textopcons</code> donne « <i>cons</i> »	<code>\textoptest</code> donne « ? »
<code>\textopdef</code> donne « <i>déf</i> »	

7.2 Équivalences et implications

7.2.1 Des symboles supplémentaires

Un premier exemple – Implication réciproque

En plus des opérateurs `\iff` et `\implies` proposés par L^AT_EX, il a été ajouté l'opérateur `\liesimp`, où l'on a inversé les groupes syllabiques de `\implies`, un opérateur pour obtenir \Leftarrow ³, ainsi que des versions négatives. Voici un exemple d'utilisation.

<code>\$(A \implies B)</code> <code>\iff (B \liesimp A)\$</code>	$(A \implies B) \iff (B \Leftarrow A)$
<code>\$(A \implies B)</code> <code>\notiff (A \notimplies B)\$</code>	$(A \implies B) \notiff (A \not\Leftarrow B)$

Un deuxième exemple – Des opérateurs décorés

Tout comme pour les égalités, il existe des versions décorées de type test, hypothèse, condition ... Elles sont toutes présentes dans l'exemple suivant.

<code>\$A \iffappli B \notiffchoice C\$</code>	$A \overset{\text{appli}}{\iff} B \overset{\text{choix}}{\notiff} C$
<code>\$A \impliescond B \notimpliescons C\$</code>	$A \overset{\text{cond}}{\implies} B \overset{\text{cons}}{\notimplies} C$
<code>\$A \liesimphyp B \notliesimptest C\$</code>	$A \overset{\text{hyp}}{\Leftarrow} B \overset{?}{\not\Leftarrow} C$

7.2.2 Une table récapitulative

La table 1 page suivante montre toutes les associations autorisées entre opérateurs logiques et décorations.

7.2.3 Équivalences et implications verticales

À quoi cela sert-il ?

Dans la section 7.4 est expliqué comment détailler les étapes d'un raisonnement. Avec cet outil, il devient utile d'avoir des versions verticales non décorées des symboles d'équivalence et d'implication. Voici comment les obtenir (*tous les cas possibles ont été indiqués*). Bien entendu le préfixe `v` est pour vertical.

3. Penser aussi aux preuves d'équivalence par double implication.

<pre> \begin{tabular}{ccccc} \$A\$ & & \$B\$ & \$C\$ & \$D\$ & \$E\$ & \$F\$ \\ \$\viff\$ & & \$\vimplies\$ & \$\vliesimp\$ & \$\notviff\$ & \$\notvimplies\$ & \$\notvliesimp\$ \\ \$A\$ & & \$B\$ & \$C\$ & \$D\$ & \$E\$ & \$F\$ \end{tabular} </pre>	$ \begin{array}{cccccc} A & B & C & D & E & F \\ \Downarrow & \Downarrow & \Uparrow & \Downarrow & \Downarrow & \Downarrow \\ A & B & C & D & E & F \end{array} $
---	---

7.3 Tables des décorations possibles des opérateurs

La table 1 de la présente page donne toutes les associations autorisées entre opérateurs et décorations. Il faut retenir que les versions étoilées produisent des écritures symboliques.

TABLE 1 – Décorations

	appli	choice	cond	cons	def	def*	hyp	id	id*	plot	test
<code>\eq</code>	×	×	×	×	×	×	×	×	×	×	×
<code>\neq</code>	×	×	×	×			×	×			×
<code>\l</code>	×	×	×	×			×			×	×
<code>\g</code>	×	×	×	×			×			×	×
<code>\leq</code>	×	×	×	×			×			×	×
<code>\geq</code>	×	×	×	×			×			×	×
<code>\iff</code>	×	×	×	×			×				×
<code>\notiff</code>	×	×	×	×			×				×
<code>\implies</code>	×	×	×	×			×				×
<code>\notimplies</code>	×	×	×	×			×				×
<code>\liesimp</code>	×	×	×	×			×				×
<code>\notliesimp</code>	×	×	×	×			×				×

7.4 Détailler un raisonnement – Version pour le lycée et après

Exemple 1 – Avec les réglages par défaut

L'environnement `explain` permet de détailler les étapes principales d'un calcul ou d'un raisonnement en s'appuyant sur la macro `\explnext` dont le nom vient de « *expl-ain next step* » soit « *expliquer la prochaine étape* » en anglais⁴. Voici un exemple avec les réglages par défaut où `\explnext*` permet d'utiliser deux lignes au lieu d'une seule⁵. Notons au passage que ce type de présentation n'est

4. Le gros du travail est fait par l'environnement `flalign` du package `amsmath` qui est automatiquement chargé par `lymath`.

5. Les explications données ne doivent pas être trop longues car ce serait contre productif. Avec deux lignes on peut distinguer une phase montante et une autre descendante.

sûrement pas bien adaptée à un jeune public pour lequel une 2^e façon de détailler des calculs ou un raisonnement est proposée dans la section 7.5 un peu plus bas.

```
\begin{explain}
  (a + b)^2
  \explnext{On utilise  $x^2 = x \cdot x$ .}
  (a + b) (a + b)
  \explnext*{ $\Downarrow$  Double développement depuis la parenthèse gauche.}%
              { $\Uparrow$  Double factorisation non nécessairement évidente.}
  a^2 + a b + b a + b^2
  \explnext{Commutativité du produit.}
  a^2 + 2 a b + b^2
\end{explain}
```

$$\begin{aligned}
 & (a + b)^2 \\
 = & \quad \{ \text{On utilise } x^2 = x \cdot x. \} \\
 & (a + b)(a + b) \\
 = & \quad \{ \Downarrow \text{Double développement depuis la parenthèse gauche.} \} \\
 & \quad \{ \Uparrow \text{Double factorisation non nécessairement évidente.} \} \\
 & a^2 + ab + ba + b^2 \\
 = & \quad \{ \text{Commutativité du produit.} \} \\
 & a^2 + 2ab + b^2
 \end{aligned}$$

Remarque. La macro `\explnext` utilise les macros constantes suivantes.

- `\textexplainleft` et `\textexplainright` qui donnent `{` et `}` respectivement par défaut.
- `\textexplainspacein` est l’espacement entre le symbole et la courte explication. Par défaut, cette macro vaut `2em`.

Exemple 2 – Utiliser un autre symbole globalement

L’environnement `explain` possède un argument optionnel qui est `=` par défaut. Ceci permet de faire ce qui suit sans effort.

<pre>\begin{explain}[\viff] x^2 + 10 x + 25 = 0 \explnext{Identité remarquable.} (x + 5)^2 = 0 \explnext{$P^2 = 0$ si et seulement si $P = 0$.} x = -5 \end{explain}</pre>	$ \begin{aligned} & x^2 + 10x + 25 = 0 \\ & \Updownarrow \quad \{ \text{Identité remarquable.} \} \\ & (x + 5)^2 = 0 \\ & \Updownarrow \quad \{ P^2 = 0 \text{ si et seulement si } P = 0. \} \\ & x = -5 \end{aligned} $
--	---

Exemple 3 – Juste utiliser des symboles

Si l’argument obligatoire de la macro `\explnext` est vide alors seul le symbole est affiché (*ne pas oublier les accolades vides*). Voici un court exemple de ceci.

```
\begin{explain}[\viff]
  a^2 = b^2
  \explnext{}
  a = \pm b
\end{explain}
```

$$a^2 = b^2$$

$$\Updownarrow$$

$$a = \pm b$$

Exemple 4 – Utiliser un autre symbole localement

La macro `\explnext` possède un argument optionnel qui utilise par défaut celui de l’environnement. En utilisant cette option, on choisit alors localement le symbole à employer. Voici un exemple d’utilisation complètement farfelu bien que correct.

```
\begin{explain}[\viff]
  0 \leq a < b
  \explnext[\vimplies]{%
    {Croissance de $x^2$
    sur $\mathbb{R}_+$}}
  a^2 < b^2
  \explnext{}
  a^2 - b^2 < 0
  \explnext{Identité remarquable.}
  (a - b)(a + b) < 0
  \explnext[\vimplies]{}
  a \neq b
\end{explain}
```

$$0 \leq a < b$$

$$\Downarrow \quad \{ \text{Croissance de } x^2 \text{ sur } \mathbb{R}_+. \}$$

$$a^2 < b^2$$

$$\Updownarrow$$

$$a^2 - b^2 < 0$$

$$\Updownarrow \quad \{ \text{Identité remarquable.} \}$$

$$(a - b)(a + b) < 0$$

$$\Downarrow$$

$$a \neq b$$

7.5 Détailler un raisonnement – Version pour les collégiens

L’environnement `aexplain`⁶ utilise des flèches pour indiquer les explications (*a* est pour *a*-rrow soit « flèche » en anglais). Son fonctionnement est similaire à celui de `explain` si ce n’est que la macro `\explnext*` permet d’avoir une flèche unidirectionnelle, vers le haut ou le bas au choix, ou bien d’écrire deux indications dont l’une est montante et l’autre descendante.

Il existe aussi l’environnement `aexplain*` lorsque la toute 1^{re} étape n’est pas expliquée. Par contre ceci nécessite au tout début de l’environnement l’usage de la macro très spéciale `\explsymb`.

Exemple 1 – Flèche à double sens ou sans flèche

```
\begin{aexplain}
  (a + b)^2
  \explnext{Identité remarquable}
  a^2 + 2 a b + b^2
  \explnext{}
  a^2 + b^2 + 2 a b
\end{aexplain}
```

$$(a + b)^2$$

$$= a^2 + 2ab + b^2 \quad \Updownarrow \text{Identité remarquable}$$

$$= a^2 + b^2 + 2ab$$

Exemple 2 – Des flèches unidirectionnelles

Ce qui suit est juste là comme démo. car les explications y sont un peu farfelues.

6. Cet environnement utilise le package `witharrows` qui est très sympathique pour expliquer des étapes de calcul.

```

\begin{aexplain}
  (a + b) (a + b)
    \explnext*{Double développement.}%
    {Double factorisation (pas simple).}
  a^2 + a b + b a + b^2
    \explnext*{Commutativité du produit.}%
    {}
  a^2 + 2 a b + b^2
    \explnext*{}%
    {Commutativité de l'addition.}
  a^2 + b^2 + 2 a b
\end{aexplain}

```

$$\begin{array}{lcl}
 (a + b)(a + b) & & \\
 = a^2 + ab + ba + b^2 & \left. \vphantom{\begin{array}{l} (a + b)(a + b) \\ = a^2 + ab + ba + b^2 \end{array}} \right\} \begin{array}{l} \text{Double développement.} \\ \text{Double factorisation (pas simple).} \end{array} \\
 = a^2 + 2ab + b^2 & \left. \vphantom{\begin{array}{l} (a + b)(a + b) \\ = a^2 + ab + ba + b^2 \\ = a^2 + 2ab + b^2 \end{array}} \right\} \text{Commutativité du produit.} \\
 = a^2 + b^2 + 2ab & \left. \vphantom{\begin{array}{l} (a + b)(a + b) \\ = a^2 + ab + ba + b^2 \\ = a^2 + 2ab + b^2 \end{array}} \right\} \text{Commutativité de l'addition.}
 \end{array}$$

Exemple 3 – Ne pas expliquer le tout début

Ci-dessous, la macro `\explsymb` n'est à utiliser qu'une seule fois !

```

\begin{aexplain*}
  (a + b) (a + b)
    \explsymb{}
  (a + b)^2
    \explnext{Identité remarquable.}
  a^2 + b^2 + 2 a b
\end{aexplain*}

```

$$\begin{array}{lcl}
 (a + b)(a + b) & = & (a + b)^2 \\
 & = & a^2 + b^2 + 2ab \quad \left. \vphantom{\begin{array}{l} (a + b)(a + b) \\ = (a + b)^2 \end{array}} \right\} \text{Identité remarquable.}
 \end{array}$$

Exemple 4 – Choisir son symbole

Voici comment faire avec la version non étoilée.

```

\begin{aexplain}[\iff]
  a^2 + 2 a b + b^2 = 0
    \explnext{}
  (a + b)^2 = 0
    \explnext*{$P^2 = 0$ ssi $P = 0$}.
  a + b = 0
\end{aexplain}

```

$$\begin{array}{lcl}
 & a^2 + 2ab + b^2 = 0 & \\
 \iff & (a + b)^2 = 0 & \left. \vphantom{\begin{array}{l} a^2 + 2ab + b^2 = 0 \\ (a + b)^2 = 0 \end{array}} \right\} P^2 = 0 \text{ ssi } P = 0. \\
 \iff & a + b = 0 &
 \end{array}$$

Avec la version étoilée, on obtient ce qui suit.

```

\begin{aexplain*}[\iff]
  a^2 + 2 a b + b^2 = 0
  \explsymb{}
  (a + b)^2 = 0
  \explnext{P^2 = 0$ ssi $P = 0$.}
  a + b = 0
\end{aexplain*}

```

$$\begin{array}{lcl}
 a^2 + 2ab + b^2 = 0 & \iff & (a + b)^2 = 0 \\
 & & \iff a + b = 0
 \end{array}
 \left. \vphantom{\begin{array}{lcl}} \right\} P^2 = 0 \text{ ssi } P = 0.$$

Exemple 6 – Un mini hack très utile

Vous pouvez écrire très facilement des calculs alignés comme suit sans trop vous fatiguez (*chacune des trois solutions présentées laisse toujours la possibilité d'expliquer une étape intermédiaire après coup*).

```

\begin{aexplain*}
  (a + b) (a + b)
  \explsymb{}
  (a + b)^2
  \explnext{}
  a^2 + b^2 + 2 a b
  \explnext{}
  a^2 + 2 a b + b^2
\end{aexplain*}

```

$$\begin{aligned}
 (a + b)(a + b) &= (a + b)^2 \\
 &= a^2 + b^2 + 2ab \\
 &= a^2 + 2ab + b^2
 \end{aligned}$$

On a accès à une autre mise en forme (*ceci peut rendre aussi service*).

```

\begin{aexplain}
  (a + b) (a + b)
  \explnext{}
  (a + b)^2
  \explnext{}
  a^2 + b^2 + 2 a b
  \explnext{}
  a^2 + 2 a b + b^2
\end{aexplain}

```

$$\begin{aligned}
 (a + b)(a + b) \\
 &= (a + b)^2 \\
 &= a^2 + b^2 + 2ab \\
 &= a^2 + 2ab + b^2
 \end{aligned}$$

Enfin dans le cadre de calculs à faire expliquer par des élèves, ce qui suit peut être utile.

```

\begin{explain}
  (a + b) (a + b)
  \explnext{}
  (a + b)^2
  \explnext{}
  a^2 + b^2 + 2 a b
  \explnext{}
  a^2 + 2 a b + b^2
\end{explain}

```

$$\begin{aligned}
 (a + b)(a + b) \\
 &= \\
 &(a + b)^2 \\
 &= \\
 &a^2 + b^2 + 2ab \\
 &= \\
 &a^2 + 2ab + b^2
 \end{aligned}$$

7.6 Des versions alternatives du quantificateur \exists

Quantifier l'existence

Voici deux versions, l'une classique, et l'autre beaucoup moins, permettant de préciser le quantificateur \exists .

<pre> \exists\existssone x \in E\$ pour \og il existe un seul \$x\$ \fg. \exists\existmulti{\leq 1} y \in F\$ pour \og il existe au plus un \$y\$ \fg. \exists\existmulti{=1} \eqdef \existssone\$ </pre>	$\exists! x \in E$ pour « il existe un seul x ». $\exists_{\leq 1} y \in F$ pour « il existe au plus un y ». $\exists_{=1} \stackrel{\text{déf}}{=} \exists!$
--	---

Versions négatives

<pre> \nexists\nexistssone\$ pour \og il n'existe pas un unique \fg. \nexists\nexistmulti{>4}\$ pour \og il n'existe pas plus de quatre \fg. \nexists\nexistss\$ est proposé par \verb+amssymb+. </pre>	$\nexists!$ pour « il n'existe pas un unique ». $\nexists_{>4}$ pour « il n'existe pas plus de quatre ». \nexists est proposé par amssymb.
---	--

8 Ensembles et applications

8.1 Différents types d'ensembles

8.1.1 Ensembles versus accolades

Exemple d'utilisation 1

<pre> $\setgene{1 ; 3 ; 5}\$.$</pre>	$\{1;3;5\}$.
---	---------------

Exemple d'utilisation 2

<pre> \displaystyle \setgene{\frac{1}{3} ; \frac{5}{7} ; \frac{9}{11}}\$ ou \displaystyle \setgene*{\frac{1}{3} ; \frac{5}{7} ; \frac{9}{11}}\$. </pre>	$\left\{ \frac{1}{3} ; \frac{5}{7} ; \frac{9}{11} \right\}$ ou $\left\{ \frac{1}{3} ; \frac{5}{7} ; \frac{9}{11} \right\}$.
--	--

8.1.2 Ensembles pour la géométrie

Exemple d'utilisation 1

<code>\setgeo{C}</code> , <code>\setgeo{D}</code> ou <code>\setgeo{d}</code>	\mathcal{C} , \mathcal{D} ou d
--	--------------------------------------

Remarque. Pour le moment, il n'est pas possible de taper `\setgeo{ABC}` avec plusieurs lettres.

Exemple d'utilisation 2 – Avec des indices

<code>\setgeo*{C}{1}</code> ou <code>\setgeo*{C}{2}</code>	\mathcal{C}_1 ou \mathcal{C}_2
---	------------------------------------

8.1.3 Ensembles probabilistes

Exemple d'utilisation 1

<code>\setproba{E}</code> ou <code>\setproba{G}</code>	\mathcal{E} ou \mathcal{G}
---	--------------------------------

Remarque. Pour le moment, il n'est pas possible de taper `\setproba{ABC}` avec plusieurs lettres.

Exemple d'utilisation 2 – Avec des indices

<code>\setproba*{E}{1}</code> ou <code>\setproba*{E}{2}</code>	\mathcal{E}_1 ou \mathcal{E}_2
---	------------------------------------

8.1.4 Ensembles pour l'algèbre générale

Exemple d'utilisation 1

<code>\setalge{A}</code> , <code>\setalge{K}</code> , <code>\setalge{h}</code> ou <code>\setalge{k}</code>	\mathbb{A} , \mathbb{K} , \mathbb{h} ou \mathbb{k}
---	--

Remarque. Pour le moment, il n'est pas possible de taper `\setalge{ABC}` avec plusieurs lettres.

Exemple d'utilisation 2 – Avec des indices

<code>\setalge*{k}{1}</code> ou <code>\setalge*{k}{2}</code>	\mathbb{k}_1 ou \mathbb{k}_2
--	----------------------------------

8.2 Ensembles classiques en mathématiques et en informatique théorique

8.2.1 La liste complète

Dans l'exemple suivant, \mathbb{P} désigne l'ensemble des nombres premiers, \mathbb{H} celui des quaternions, \mathbb{O} celui des octonions et \mathbb{F} un ensemble de nombres flottants (*notation à préciser suivant le contexte*).

$\backslash\text{nullset}$, $\backslash\text{NN}$, $\backslash\text{ZZ}$, $\backslash\text{PP}$	\emptyset , \mathbb{N} , \mathbb{Z} , \mathbb{P}
$\backslash\text{DD}$, $\backslash\text{QQ}$, $\backslash\text{RR}$, $\backslash\text{CC}$	\mathbb{D} , \mathbb{Q} , \mathbb{R} , \mathbb{C}
$\backslash\text{HH}$, $\backslash\text{OO}$	\mathbb{H} , \mathbb{O}
$\backslash\text{FF}$	\mathbb{F}

8.2.2 Ensembles classiques suffixés

L'ensemble \mathbb{R} nous permet de voir tous les cas possibles.

$\backslash\text{RRn}$, $\backslash\text{RRp}$, $\backslash\text{RRs}$, $\backslash\text{RRsn}$ ou $\backslash\text{RRsp}$	\mathbb{R}_- , \mathbb{R}_+ , \mathbb{R}^* , \mathbb{R}_-^* ou \mathbb{R}_+^*
---	---

Nous avons utilisé les suffixes **n** pour **n**-égatif, **p** pour **p**-ositif et **s** pour **s**-tar soit « étoile » en anglais. Il y a aussi les suffixes composites **sn** et **sp**.

Notez qu'il est interdit d'utiliser $\backslash\text{CCn}$ pour \mathbb{C}_- car l'ensemble \mathbb{C} ne possède pas de structure ordonnée standard. Jetez un oeil à la section suivante pour apprendre à taper \mathbb{C}_- si vous en avez besoin. L'interdiction est ici purement sémantique !

Remarque. La table 2 de la présente page montre les associations autorisées entre ensembles classiques et suffixes.

TABLE 2 – Suffixes

	n	p	s	sn	sp
$\backslash\text{NN}$			×		
$\backslash\text{PP}$					
$\backslash\text{ZZ}$	×	×	×	×	×
$\backslash\text{DD}$	×	×	×	×	×
$\backslash\text{QQ}$	×	×	×	×	×
$\backslash\text{RR}$	×	×	×	×	×
$\backslash\text{CC}$			×		
$\backslash\text{HH}$			×		
$\backslash\text{OO}$			×		
$\backslash\text{FF}$	×	×	×	×	×

8.3 Des suffixes à la carte

Exemple d'utilisation

Dans cet exemple, il faut savoir que le 2^e argument ne peut prendre que les valeurs `n`, `p`, `s`, `sn` ou `sp`.

```
\setspecial{\CC}{n}$ ,
\setspecial{\HH}{sp}$ ou
\setspecial*{\setproba{P}}{n}$
```

\mathbb{C}_- , \mathbb{H}_+^* ou $\mathcal{P}_{\leq 0}$

8.4 Intervalles

8.4.1 Intervalles réels - Notation française (?)

Exemple d'utilisation 1

Dans cet exemple, la syntaxe fait référence à `O-pened` et `C-losed` pour « *ouvert et fermé* » en anglais. Nous verrons que `CC` et `OO` sont contractés en `C` et `O`. Notez au passage que la macro utilisée résout un problème d'espacement vis à vis du signe `=`.

```
$I = ]a ; b] = \intervalOC{a}{b}$
```

$I =]a ; b] =]a ; b]$

Exemple d'utilisation 2

Les crochets s'étendent verticalement automatiquement. Pour empêcher cela, il suffit d'utiliser la version étoilée de la macro. Dans ce cas, les crochets restent tout de même un peu plus grands que des crochets utilisés directement. Voici un exemple.

```
$\displaystyle
\intervalC{ \frac{1}{2} }{ 1^{2^3} }
=
[ \frac{1}{2} ; 1^{2^3} ]
=
\intervalC*{ \frac{1}{2} }{ 1^{2^3} }$
```

$\left[\frac{1}{2} ; 1^{2^3} \right] = \left[\frac{1}{2} ; 1^{2^3} \right] = \left[\frac{1}{2} ; 1^{2^3} \right]$

8.4.2 Intervalles réels - Notation américaine

Exemple d'utilisation

Dans cet exemple, la syntaxe fait référence à `P-arenthèse`. Cette notation est utilisée États Unis.

```
$\intervalPC{a}{b} = \intervalOC{a}{b}$
et
$\intervalP{a}{b} = \intervalO{a}{b}$.
```

$(a ; b] =]a ; b]$ et $(a ; b) =]a ; b[$.

8.4.3 Intervalles discrets d'entiers

Exemple d'utilisation

Dans l'exemple, la syntaxe fait référence à \mathbb{Z} l'ensemble des entiers relatifs.

```

 $\ZintervalC{-1}{4}$ 
=  $\{-1; 0; 1; 2; 3; 4\}$ 

 $\ZintervalC{-1}{4}$ 
=  $\ZintervalO{-2}{5}$ .

```

$$\llbracket -1; 4 \rrbracket = \{-1; 0; 1; 2; 3; 4\}$$

$$\llbracket -1; 4 \rrbracket = \llbracket -2; 5 \rrbracket.$$

8.5 Unions et intersections

Exemple d'unions

Ci-dessous est utilisée la macro `\bigcup` proposée par le package `amssymb`.

```

 $A \cup B$ 

 $\dcup_{k=1}^n A_k$ 

 $\bigcup_{k=1}^n B_k$ 

 $\displaystyle \bigcup_{k=1}^n C_k$ 

```

$$A \cup B$$

$$\bigcup_{k=1}^n A_k$$

$$\bigcup_{k=1}^n B_k$$

$$\bigcup_{k=1}^n C_k$$

Exemple d'unions disjointes

Ci-dessous sont utilisées les macros `\sqcup` et `\bigsqcup` proposée par le package `amssymb`.

```

 $A \sqcup B$ 

 $\dsqcup_{k=1}^n A_k$ 

 $\bigsqcup_{k=1}^n B_k$ 

 $\displaystyle \bigsqcup_{k=1}^n C_k$ 

```

$$A \sqcup B$$

$$\bigsqcup_{k=1}^n A_k$$

$$\bigsqcup_{k=1}^n B_k$$

$$\bigsqcup_{k=1}^n C_k$$

Exemple d'intersections

Ci-dessous est utilisée la macro `\bigcap` proposée par le package `amssymb`.

```

 $A \cap B$ 

 $\dcap_{k=1}^n A_k$ 

 $\bigcap_{k=1}^n B_k$ 

 $\displaystyle \bigcap_{k=1}^n C_k$ 

```

$$A \cap B$$

$$\bigcap_{k=1}^n A_k$$

$$\bigcap_{k=1}^n B_k$$

$$\bigcap_{k=1}^n C_k$$

8.6 Cardinal, image et compagnie

Cardinal – Un exemple

```

 $\card* E = \card E$ 

```

$$\#E = \text{card } E$$

Image et compagnie – Un exemple

`\ker f` , `\dom f` ,
`\im f` ou `\codom f`

$\ker f$, $\text{dom } f$, $\text{im } f$ ou $\text{codom } f$

8.7 Application totale, partielle, injective, surjective et/ou bijective

Voici des symboles qui, bien que très techniques, facilitent la rédaction de documents à propos des applications totales ou partielles⁷ (*on parle aussi d'applications, sans qualificatif, et de fonctions*).

Applications totales – Un exemple complet

`f: A \to B` est une application totale, c'est à dire définie sur `A` tout entier.

`i: C \onetoone D` est une application totale injective.

`s: E \onto F` est une application totale surjective.

`b: G \bijet H` est une application totale bijective.

$f : A \rightarrow B$ est une application totale, c'est à dire définie sur A tout entier.

$i : C \rightarrowtail D$ est une application totale injective.

$s : E \twoheadrightarrow F$ est une application totale surjective.

$b : G \rightarrowtailtail H$ est une application totale bijective.

Applications partielles – Un exemple complet

`f: A \pto B` est une application partielle, c'est à dire définie sur un sous-ensemble de `A`.

`i: C \ponetoone D` est une application partielle injective.

`s: E \ponto F` est une application partielle surjective.

`b: G \pbijet H` est une application partielle bijective.

$f : A \rightarrowtail B$ est une application partielle, c'est à dire définie sur un sous-ensemble de A .

$i : C \rightarrowtailtail D$ est une application partielle injective.

$s : E \rightarrowtailtailtail F$ est une application partielle surjective.

$b : G \rightarrowtailtailtailtail H$ est une application partielle bijective.

7. $a : E \rightarrow F$ est une application totale si $\forall x \in E, \exists ! y \in F$ tel que $y = a(x)$. Plus généralement, $f : E \rightarrow F$ est une application partielle si $\forall x \in E, \exists \leq 1 y \in F$ tel que $y = f(x)$, autrement dit soit $f(x)$ existe dans F , soit f n'est pas définie en x .

9 Géométrie

9.1 Points et lignes

9.1.1 Points

Exemple d'utilisation 1

<code>$\backslash pt{I}$</code>	I
--	-----

Exemple d'utilisation 2

<code>$\backslash pt*{I}{1}$</code> ou <code>$\backslash pt*{I}{2}$</code>	I_1 ou I_2
---	----------------

9.1.2 Lignes

Exemple 1 – Les droites

Dans l'exemple suivant, le préfixe **g** est pour **g**-éométrie tandis que **p** est pour **p**-oint.

<code>$\backslash gline{A}{B}$</code> , <code>$\backslash gline{\backslash pt{A}}{\backslash pt{B}}$</code> ou <code>$\backslash pgline{A}{B}$</code>	(AB) , (AB) ou (AB)
--	---------------------------

Exemple 2 – Les segments

Les macros `$\backslash segment$` et `$\backslash psegment$` ont un comportement similaire à `$\backslash gline$` et `$\backslash pgline$` .

<code>$\backslash segment{A}{B}$</code> , <code>$\backslash segment{\backslash pt{A}}{\backslash pt{B}}$</code> ou <code>$\backslash psegment{A}{B}$</code>	$[AB]$, $[AB]$ ou $[AB]$
--	---------------------------

Exemple 3 – Les demi-droites

Dans l'exemple suivant, le préfixe **h** est pour **h**-alf soit « *moitié* » en anglais.

<code>$\backslash hgline{A}{B}$</code> , <code>$\backslash hgline{\backslash pt{A}}{\backslash pt{B}}$</code> ou <code>$\backslash phgline{A}{B}$</code>	$[AB)$, $[AB)$ ou $[AB)$
---	---------------------------

Exemple 4 – D'autres demi-droites

Ce qui suit nécessite d'utiliser l'argument optionnel de `$\backslash gline$` et `$\backslash pgline$` . La valeur **OC** provient de **O**-pened – **C**-losed soit « *ouvert* – *fermé* » en anglais.

<code>$\backslash gline[OC]{A}{B}$</code> , <code>$\backslash gline[OC]{\backslash pt{A}}{\backslash pt{B}}$</code> ou <code>$\backslash pgline[OC]{A}{B}$</code>	$(AB]$, $(AB]$ ou $(AB]$
--	---------------------------

Remarque. Les segments utilisent en fait l'option `C` et les demi-droites standard l'option `CO`. La valeur par défaut est 0.

9.1.3 Droites parallèles ou non

Les opérateurs `\parallel` et `\nparallel` utilisent des obliques au lieu de barres verticales comme le montre l'exemple qui suit où `\stdnparallel` est un alias de `\nparallel` fourni par le package `amssymb`, et `\stdparallel` est un alias de la version standard de `\parallel` proposée par L^AT_EX.

<pre><code>\pgline{A}{B} \parallel \pgline{C}{D}\$</code></pre> <p>au lieu de</p> <pre><code>\pgline{A}{B}</code></pre> <pre><code>\stdparallel \pgline{C}{D}\$</code></pre>	$(AB) \parallel (CD)$ au lieu de $(AB) \parallel (CD)$ $(EF) \nparallel (GH)$ au lieu de $(EF) \nparallel (GH)$
<pre><code>\pgline{E}{F} \nparallel \pgline{G}{H}\$</code></pre> <p>au lieu de</p> <pre><code>\pgline{E}{F}</code></pre> <pre><code>\stdnparallel \pgline{G}{H}\$</code></pre>	

9.2 Vecteurs

9.2.1 Les écrire

Exemple d'utilisation 1

<pre><code>\vect{ABCDEF}\$</code> ,</pre> <pre><code>\vect*{e}{rot}\$</code> ou</pre> <pre><code>\vect{e_{rot}}\$</code></pre>	\overrightarrow{ABCDEF} , \vec{e}_{rot} ou $\overrightarrow{e_{rot}}$
--	---

Exemple d'utilisation 2

<pre><code>\vect{i}\$</code> ou</pre> <pre><code>\vect*{j}{2}\$</code></pre>	\vec{i} ou \vec{j}_2
--	--------------------------

9.2.2 Norme

Exemple d'utilisation

<pre><code>\norm{\vect{i}}\$</code></pre> <pre><code>\displaystyle</code></pre> <pre><code>\norm{\frac{2}{7} \vect*{e}{k}}\$</code></pre> <pre><code>\displaystyle</code></pre> <pre><code>\norm*{\frac{2}{7} \vect*{e}{k}}\$</code></pre>	$\ \vec{i}\ $ $\left\ \frac{2}{7} \vec{e}_k \right\ $ $\left\ \frac{2}{7} \vec{e}_k \right\ $
--	---

Remarque. Le code L^AT_EX vient directement de ce message : <https://tex.stackexchange.com/a/43009/6880>.

9.2.3 Produit scalaire – Écriture minimaliste

Exemple d'utilisation - Version longue

<code>$\dotprod{\dfrac{1}{2} \vect{i}}{\vect{j}}$</code>	$\frac{1}{2} \vec{i} \cdot \vec{j}$
---	-------------------------------------

Exemple d'utilisation - Version courte mais restrictive

Dans l'exemple suivant, le préfixe v est pour v-ecteur.

<code>$\vdotprod{i}{j}$</code>	$\vec{i} \cdot \vec{j}$
---	-------------------------

9.2.4 Produit scalaire – Écriture « à la physicienne »

Dans l'exemple suivant, le préfixe a est pour a-ngle et v pour v-ecteur. Les physiciens aiment bien cette notation.

<code>$\displaystyle \adotprod{\frac{1}{2} \vect{i}}{\vect{j}}$</code>	$\left\langle \frac{1}{2} \vec{i} \mid \vec{j} \right\rangle$
<code>$\displaystyle \adotprod*{\frac{1}{2} \vect{i}}{\vect{j}}$</code>	$\left\langle \frac{1}{2} \vec{i} \mid \vec{j} \right\rangle$
<code>$\vdotprod{i}{j}$</code>	$\left\langle \vec{i} \mid \vec{j} \right\rangle$
<code>$\vdotprod*{i}{j}$</code>	$\left\langle \vec{i} \mid \vec{j} \right\rangle$

9.2.5 Produit vectoriel

Exemple d'utilisation - Version longue

<code>$\crossprod{\dfrac{1}{2} \vect{i}}{\vect{j}}$</code>	$\frac{1}{2} \vec{i} \wedge \vec{j}$
---	--------------------------------------

Exemple d'utilisation - Version courte mais restrictive

<code>$\vcrossprod{i}{j}$</code>	$\vec{i} \wedge \vec{j}$
---	--------------------------

Exemple d'utilisation - Explication des calculs

Dans l'exemple suivant, le préfixe c, qui double cette lettre au début, est pour c-oordonnées et v pour v-ecteur.

<pre> $\backslash\text{ccrossprod}\{\text{vect}\{u\}\{x\}\{y\}\{z\}\%$ $\{\text{vect}\{v\}\{x'\}\{y'\}\{z'\}\}$ = $\backslash\text{vccrossprod}\{u\}\{x\}\{y\}\{z\}\%$ $\{v\}\{x'\}\{y'\}\{z'\}\\$ </pre>	$\begin{array}{c} \vec{u} \quad \vec{v} \\ \left \begin{array}{cc} x & x' \\ y & y' \\ z & z' \\ x & x' \end{array} \right = \left \begin{array}{cc} x & x' \\ y & y' \\ z & z' \\ x & x' \end{array} \right $
--	---

9.2.6 Plan – Déterminant de deux vecteurs

Exemple d'utilisation - Version décorée

Dans l'exemple suivant, le préfixe c est pour c-ordonnées.

<pre> $\backslash\text{cdetplane}\{\text{vect}\{u\}\{x\}\{y\}\%$ $\{\text{vect}\{v\}\{x'\}\{y'\}\\$ </pre>	$\begin{array}{c} \vec{u} \quad \vec{v} \\ \left \begin{array}{cc} x & x' \\ y & y' \end{array} \right $
---	---

Exemple d'utilisation - Version non décorée

<pre> $\backslash\text{cdetplane}*\{\text{vect}\{u\}\{x\}\{y\}\%$ $\{\text{vect}\{v\}\{x'\}\{y'\}\\$ </pre>	$\begin{array}{c} \vec{u} \quad \vec{v} \\ \left \begin{array}{cc} x & x' \\ y & y' \end{array} \right $
--	---

Exemple d'utilisation - Versions courtes

Dans l'exemple suivant, le préfixe v est pour v-ecteur.

<pre> $\backslash\text{vcdetplane}\{u\}\{x\}\{y\}\%$ $\{v\}\{x'\}\{y'\}$ = $\backslash\text{vcdetplane}*\{u\}\{x\}\{y\}\%$ $\{v\}\{x'\}\{y'\}\\$ </pre>	$\begin{array}{c} \vec{u} \quad \vec{v} \quad \vec{u} \quad \vec{v} \\ \left \begin{array}{cc} x & x' \\ y & y' \end{array} \right = \left \begin{array}{cc} x & x' \\ y & y' \end{array} \right $
--	---

9.3 Coordonnées

Exemple d'utilisation 1

<pre> $\backslash\text{displaystyle}$ $\backslash\text{pt}\{I\} \backslash\text{coord}\{\frac{1}{3} \mid -4 \mid 0\}\\$ ou $\backslash\text{displaystyle}$ $\backslash\text{pt}\{I\} \backslash\text{coord}*\{\frac{1}{3} \mid -4 \mid 0\}\\$ </pre>	$I\left(\frac{1}{3}; -4; 0\right) \text{ ou } I\left(\frac{1}{3}; -4; 0\right)$
--	---

Exemple d'utilisation 2

Dans l'exemple suivant, le préfixe v est pour v-ertical.

```

 $\vec{i}$  \vcoord{3 | -4 | 0}$ ,
 $\vec{i}$  \vcoord*{3 | -4 | 0}$
à la place de
 $\vec{i}$  \coord{3 | -4 | 0}$

```

$\vec{i} \begin{pmatrix} 3 \\ -4 \\ 0 \end{pmatrix}$, $\vec{i} \begin{bmatrix} 3 \\ -4 \\ 0 \end{bmatrix}$ à la place de $\vec{i} (3; -4; 0)$

9.4 Nommer un repère

Exemple d'utilisation 1 – La méthode basique

Commençons par la manière la plus basique d'écrire un repère (*nous verrons d'autres méthodes qui peuvent être plus efficaces*).

```

 $\text{\axes{\pt{0} %$ 
 $\text{\pt{I} | \pt{J}}}$ 

```

$(O; I, J)$

Exemple d'utilisation 2 – La méthode basique en version étoilée

Dans l'exemple ci-dessous, on voit que la version étoilée produit des petites parenthèses.

```

 $\displaystyle$ 
 $\text{\axes{\pt{0} %$ 
 $\text{\frac{7}{3} \vect{i} %$ 
 $\text{\vect{j}}}$ 
ou
 $\displaystyle$ 
 $\text{\axes*{\pt{0} %$ 
 $\text{\frac{7}{3} \vect{i} %$ 
 $\text{\vect{j}}}$ 

```

$\left(O; \frac{7}{3} \vec{i}, \vec{j}\right)$ ou $\left(O; \frac{7}{3} \vec{i}, \vec{j}\right)$

Exemple d'utilisation 3 – La méthode basique en dimension quelconque

Il faut au minimum deux "morceaux" séparés par des barres |, cas de la dimension 1, mais il n'y a pas de maximum, cas d'une dimension quelconque $n > 0$.

```

 $\text{\axes{\pt{0} %$ 
 $\text{\vect*{i}{1} %$ 
 $\text{\vect*{i}{2} %$ 
 $\text{\vect*{i}{3} %$ 
 $\text{\dots %$ 
 $\text{\vect*{i}{9} %$ 
 $\text{\vect*{i}{10} %$ 
 $\text{\vect*{i}{11} %$ 
 $\text{\vect*{i}{12}}}$ 

```

$(O; \vec{i}_1, \vec{i}_2, \vec{i}_3, \dots, \vec{i}_9, \vec{i}_{10}, \vec{i}_{11}, \vec{i}_{12})$

Exemple d'utilisation 4 – Repère affine

Dans l'exemple suivant, le préfixe p est pour p-oint.

`$\paxes{O | I | J | K}$`
 au lieu de
`$\axes{\pt{O} %`
`| \pt{I} | \pt{J} | \pt{K}}$`

$(O; I, J, K)$ au lieu de $(O; I, J, K)$

Exemple d'utilisation 5 – Repère vectoriel (méthode 1)

Dans l'exemple suivant, le préfixe v est pour v-ecteur.

`$\vaxes{\pt{O} | i | j}$`
 au lieu de
`$\axes{\pt{O} | \vect{i} | \vect{j}}$`

$(O; \vec{i}, \vec{j})$ au lieu de $(O; \vec{i}, \vec{j})$

Exemple d'utilisation 6 – Repère vectoriel (méthode 2)

Dans l'exemple suivant, le préfixe pv permet de combiner ensemble les fonctionnalités proposées par les préfixes p et v.

`$\pvaxes{O | i | j}$`
 au lieu de
`$\axes{\pt{O} | \vect{i} | \vect{j}}$`

$(O; \vec{i}, \vec{j})$ au lieu de $(O; \vec{i}, \vec{j})$

9.5 Arcs circulaires

Exemple d'utilisation 1

`\circarc{ABCDEF} ,`
`$\circarc*{A}{rot}$ ou`
`$\circarc{A_{rot}}$`

\widehat{ABCDEF} , \hat{A}_{rot} ou $\widehat{A_{rot}}$

Exemple d'utilisation 2

`\circarc{i} ou`
`$\circarc*{j}{2}$`

\hat{i} ou \hat{j}_2

9.6 Angles

9.6.1 Angles géométriques « intérieurs »

Exemple d'utilisation 1

`\anglein{ABCDEF} ,`
`$\anglein*{A}{rot}$ ou`
`$\anglein{A_{rot}}$`

\widehat{ABCDEF} , \hat{A}_{rot} ou $\widehat{A_{rot}}$

Exemple d'utilisation 2

```
$\anglein{i}$ ou  
$\anglein*{j}{2}$
```

\widehat{i} ou \widehat{j}_2

9.6.2 Angles orientés de vecteurs

Sans chapeau - Version longue

```
$\displaystyle  
\angleorient{\frac{1}{2} \vect{i}}%  
{\vect{j}}$  
ou  
$\displaystyle  
\angleorient*{\frac{1}{2} \vect{i}}%  
{\vect{j}}$ .
```

$\left(\frac{1}{2}\vec{i}; \vec{j}\right)$ ou $\left(\frac{1}{2}\vec{i}; \vec{j}\right)$.

Sans chapeau - Version courte mais restrictive

Dans l'exemple suivant, le préfixe **v** est pour **v**-ecteur qui permet de simplifier la saisie quand l'on a juste des vecteurs nommés avec des lettres (*notez que la version étoilée n'apporte rien de nouveau*).

```
$\vangleorient{i}{j}$  
ou  
$\vangleorient*{i}{j}$
```

$(\vec{i}; \vec{j})$ ou $(\vec{i}; \vec{j})$

Avec un chapeau

Dans l'exemple suivant, le préfixe **h** est pour **h**-at soit « *chapeau* » en anglais, et **v** pour **v**-ecteur. Ceci permet d'avoir les angles orientés avec un chapeau (*la dernière écriture n'apporte rien de neuf mais elle évite de tout reprendre en cas de changement de mise en forme dans un document*).

```
$\displaystyle  
\hangleorient{\frac{1}{2} \vect{i}}%  
{\vect{j}}$  
  
$\displaystyle  
\hangleorient*{\frac{1}{2} \vect{i}}%  
{\vect{j}}$  
  
$\hvangleorient{i}{j}$  
  
$\hvangleorient*{i}{j}$
```

$\overline{\left(\frac{1}{2}\vec{i}; \vec{j}\right)}$
 $\overline{\left(\frac{1}{2}\vec{i}; \vec{j}\right)}$
 $\overline{(\vec{i}; \vec{j})}$
 $\overline{(\vec{i}; \vec{j})}$

10 Analyse

10.1 Constantes

10.1.1 Constantes classiques

La liste complète

<code>\gamma</code> , <code>\pi</code> , <code>\tau</code> , <code>\epsilon</code> , <code>\ii</code> , <code>\jj</code> et <code>\kk</code> où <code>\tau = 2 \pi</code>	γ , π , τ , ϵ , \mathbf{i} , \mathbf{j} et \mathbf{k} où $\tau = 2\pi$
---	---

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

10.1.2 Constantes latines personnelles

Exemple d'utilisation

<code>\ct{a} x^2 + \ct{b} x + \ct{c}</code> ou <code>\a x^2 + b x + c</code>	$ax^2 + bx + c$ ou $ax^2 + bx + c$
--	------------------------------------

10.2 La fonction valeur absolue

Un exemple d'utilisation

<code>\abs{2}</code> , <code>\displaystyle \abs{\frac{3}{5}}</code> ou <code>\displaystyle \abs*{\frac{3}{5}}</code>	$ 2 $, $\left \frac{3}{5}\right $ ou $\left \frac{3}{5}\right $
--	--

Remarque. Le code L^AT_EX vient directement de ce poste : <https://tex.stackexchange.com/a/43009/6880>.

10.3 Fonctions nommées spéciales

10.3.1 Sans paramètre

Un exemple d'utilisation

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.

<code>\fch x \neq ch x</code> , <code>\ppcm(x;y)</code> ou <code>\lg x</code>	$ch\,x \neq chx$, $ppcm(x;y)$ ou $\lg x$
---	---

10.3.2 Avec un paramètre

Un exemple d'utilisation

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

$\log_2 x = \lg x$ ou $\exp_6 y = 6^y$

10.3.3 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 ...

10.4 Des notations complémentaires pour des suites spéciales

Exemple d'utilisation

`\seqplus{F}{1}{2}$`
`\seqhypergeo{F}{1}{2}$`
`\seqsuprageo{F}{1}{2}{3}{4}$`
`pour les fous\dots :-)`

F_1^2
 ${}_1F_2$
 ${}_1F_2^3$ pour les fous... :-)

10.5 Calcul différentiel

10.5.1 Les opérateurs ∂ et d

Exemple d'utilisation

`\dd{f}$` , `\pp{t}$` ,
`\dd[5]{x}$` ou `\pp[n]{x}$`

df , ∂t , d^5x ou $\partial^n x$

10.5.2 Dérivation totale

Exemple d'utilisation 1

`\displaystyle`
`\derpow{f} (a)`
`= \derpow{f} (a)`
`= \derfrac{f}{x} (a)`
`= \dersub{f}{x} (a)$`

$f'(a) = f^{(1)}(a) = \frac{df}{dx}(a) = d_x f(a)$

Exemple d'utilisation 2

```


$$\begin{aligned} & \backslash\mathrm{derpow}[3]{f}(a) \\ &= \backslash\mathrm{derpow}[3]{f}(a) \\ &= \backslash\mathrm{derfrac}[3]{f}{x}(a) \\ &= \backslash\mathrm{dersub}[3]{f}{x}(a) \end{aligned}$$


$$\begin{aligned} & \backslash\mathrm{derpow}[10]{\cos} a \\ &= \backslash\mathrm{derfrac}[10]{\cos}{x}(a) \end{aligned}$$


```

$$\begin{aligned} f'''(a) &= f^{(3)}(a) = \frac{d^3 f}{dx^3}(a) = d_x^3 f(a) \\ \cos^{(10)} a &= \frac{d^{10} \cos}{dx^{10}}(a) \end{aligned}$$

Exemple d'utilisation 3

```


$$\begin{aligned} & \backslash\mathrm{derpow}[3]{f}(a) \\ &= \backslash\mathrm{derfrac}[3]{\left(\frac{1}{x}\right)}{x}(a) \end{aligned}$$


```

$$f^{(3)}(a) = \frac{d^3}{dx^3} \left(\frac{1}{x} \right) (a)$$

Exemple d'utilisation 4

```


$$\begin{aligned} & \backslash\mathrm{derpar}[3]{\frac{1}{2} uv} \\ &= \backslash\mathrm{derpar}[3]{\frac{1}{2} uv} \end{aligned}$$


$$\begin{aligned} & \backslash\mathrm{sderpar}[3]{\frac{1}{2} uv} \\ &= \backslash\mathrm{sderpar}[3]{\frac{1}{2} uv} \end{aligned}$$


```

$$\begin{aligned} \left(\frac{1}{2} uv \right)^{(3)} &= \left(\frac{1}{2} uv \right)''' \\ \left(\frac{1}{2} uv \right)^{(3)} &= \left(\frac{1}{2} uv \right)''' \end{aligned}$$

10.5.3 Dérivation partielle

Exemple d'utilisation 1

```


$$\begin{aligned} & \backslash\mathrm{partialfrac}{f}{x}(a;b) \\ &= \backslash\mathrm{partialsub}{f}{x}(a;b) \\ &= \backslash\mathrm{partialprime}{f}{x}(a;b) \end{aligned}$$


```

$$\frac{\partial f}{\partial x}(a;b) = \partial_x f(a;b) = f'_x(a;b)$$

Exemple d'utilisation 2

```


$$\frac{\partial^3 G}{\partial f^2 \partial v}(a; b) = \frac{\partial G}{\partial f^2 \partial v}(a; b) = \dots$$


$$\dots = \partial_{f(2)v} G(a; b) = G'_{f(2)v}(a; b)$$


```

$$\frac{\partial^3 G}{\partial f^2 \partial v}(a; b) = \frac{\partial G}{\partial f^2 \partial v}(a; b) = \dots$$

$$\dots = \partial_{f(2)v} G(a; b) = G'_{f(2)v}(a; b)$$

Exemple d'utilisation 3

```


$$\frac{\partial^2 f}{\partial x \partial y} = \frac{\partial^2}{\partial x \partial y} \left( \frac{1}{\cos(xy)} \right)$$


```

$$\frac{\partial^2 f}{\partial x \partial y} = \frac{\partial^2}{\partial x \partial y} \left(\frac{1}{\cos(xy)} \right)$$

10.6 Calcul intégral

10.6.1 L'opérateur crochet – 1^{ère} version

Exemple d'utilisation 1

Il faut savoir que `hook` signifie « *crochet* » en anglais mais la bonne traduction est en fait « *square bracket* ».

```


$$\int_a^b f(x) \, dx = [F(x)]_a^b$$


$$[F(x)]_a^b = F(b) - F(a)$$


```

$$\int_a^b f(x) \, dx = [F(x)]_a^b$$

$$[F(x)]_a^b = F(b) - F(a)$$

Exemple d'utilisation 2

Par défaut, les crochets s'étirent verticalement si besoin, mais si cela vous dérange, vous pouvez faire appel à la version étoilée de la macro comme dans l'exemple suivant

```


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


```

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

10.6.2 L'opérateur crochet – 2^{nde} version

Exemple d'utilisation 1

Ci-dessous `v` est pour `v`-ertical.


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

$$F(x)|_a^b = [F(x)]_a^b.$$

Exemple d'utilisation 2

Tout comme avec la première version de l'opérateur crochet, vous pouvez utiliser une version étoilée pour empêcher l'étirement verticalement du trait vertical. Voici un exemple.

```
\displaystyle
\hook{\frac{x - 1}{5 + x^2}}{a}{b}
= \hook*{\frac{x - 1}{5 + x^2}}{a}{b}
```

$$\left. \frac{x - 1}{5 + x^2} \right|_a^b = \frac{x - 1}{5 + x^2} \Big|_a^b$$

10.6.3 Intégrales multiples

Le package réduit les espacements entre des symboles \int successifs. Voici un exemple.

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

$$\iiint F(x;y;z) \, dx \, dy \, dz = \int_a^b \int_c^d \int_e^f F(x;y;z) \, dx \, dy \, dz$$

Remarque. Par défaut, L^AT_EX affiche $\int \int \int F(x;y;z) \, dx \, dy \, dz = \int_a^b \int_c^d \int_e^f F(x;y;z) \, dx \, dy \, dz$. Nous avons obtenu ce résultat en utilisant `\stdint` qui est l'opérateur proposé de façon standard par L^AT_EX.

10.7 Tableaux de variation et de signe

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. Nous vous demandons donc de vous reporter à la documentation de `tkz-tab` pour savoir comment s'y prendre.

Un exemple de tableaux de signes

```
\begin{tikzpicture}
\tkzTabInit{$x$ / 1 , $\cos(x)$ / 1}%
{$0$, $\frac{\pi}{2}$, $\pi$}

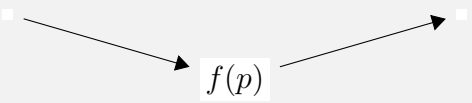
\tkzTabLine{ , + , z , - , }
\end{tikzpicture}
```

x	0	$\frac{\pi}{2}$	π
$\cos(x)$	+	0	−

Un exemple de tableaux de variation

```
\begin{tikzpicture}
  \tkzTabInit{$x$ / 1 , $f(x)$ / 1.5}%
    {$-\infty$, $p$, $+\infty$}

  \tkzTabVar{+/- , -/ $f(p)$ , +/ }
\end{tikzpicture}
```

x	$-\infty$	p	$+\infty$
$f(x)$			

Un exemple de tableaux de variation avec une dérivée

```
\begin{tikzpicture}
  \tkzTabInit{$x$ / 1 , $\cos(x)$ / 1 , $\sin(x)$ / 1.5}%
    {$0$, $\frac{\pi}{2}$, $\pi$}

  \tkzTabLine{ , + , z , - , }
  \tkzTabVar{-/ 0 , +/ 1 , -/ 0}
\end{tikzpicture}
```

x	0	$\frac{\pi}{2}$	π
$\cos(x)$	+	0	-
$\sin(x)$	0	1	0

10.8 Comparaison asymptotique de suites et de fonctions

10.8.1 Les notations \mathcal{O} et \mathcal{o}

Exemple d'utilisation 1

Les notations suivantes sont dues à Landau.

\mathcal{O} ou \mathcal{o}

Exemple d'utilisation 2

$\mathcal{O}(x) \neq \mathcal{o}(x)$ ou $e^{t+\mathcal{O}(t)} = e^{\mathcal{O}(t)}$

10.8.2 La notation Ω

Exemple d'utilisation 1

La notation suivante est due à Hardy et Littlewood.

<code>\bigomega{}</code>	Ω
--------------------------	----------

Exemple d'utilisation 2

Dans l'exemple suivant, $f(n) = \Omega(g(n))$ signifie : $\exists(m, n_0)$ tel que $n \geq n_0$ implique $f(n) \geq mg(n)$.

<code>f(n) = \bigomega{g(n)}</code>	$f(n) = \Omega(g(n))$
-------------------------------------	-----------------------

10.8.3 La notation Θ

Exemple d'utilisation 1

<code>\bigtheta{}</code>	Θ
--------------------------	----------

Exemple d'utilisation 2

Dans l'exemple suivant, $f(n) = \Theta(g(n))$ signifie : $\exists(m, M, n_0)$ tel que $mg(n) \leq f(n) \leq Mg(n)$ dès que $n \geq n_0$.

<code>f(n) = \bigtheta{g(n)}</code>	$f(n) = \Theta(g(n))$
-------------------------------------	-----------------------

11 Probabilité

11.1 Probabilité « simple »

Un exemple type

<code>\proba{A}</code> ou <code>\proba[P]{A}</code>	$p(A)$ ou $P(A)$
--	------------------

11.2 Probabilité conditionnelle

Un exemple type

La 1^{re} notation, qui est devenue standard, permet de comprendre l'ordre des arguments.

<code>\probacond{B}{A}</code> = <code>\probacond*{B}{A}</code> = <code>\probacond**{B}{A}</code> = <code>\dprobacond**{B}{A}</code>	$p_B(A) = p(A \mid B) = \frac{p(A \cap B)}{p(B)} = \frac{p(A \cap B)}{p(B)}$
---	--

Choisir le nom de la probabilité

$\begin{aligned} &\$ \backslash \text{probacond}[P]{B}{A} \\ &= \\ &\backslash \text{probacond}*[P]{B}{A} \\ &= \\ &\backslash \text{probacond}**[P]{B}{A} \\ &= \\ &\backslash \text{dprobacond}**[P]{B}{A} \$ \end{aligned}$	$P_B(A) = P(A \mid B) = \frac{P(A \cap B)}{P(B)} = \frac{P(A \cap B)}{P(B)}$
--	--

11.3 Espérance

Un exemple type

expval vient de *exp-ected val-ue* soit « *espérance* » en anglais.

$\$ \backslash \text{expval}\{X\} \$$	$E(X)$
---------------------------------------	--------

Choisir le nom de l'espérance

$\$ \backslash \text{expval}[E_1]\{X\} \$$	$E_1(X)$
--	----------

11.4 Arbres pondérés

Que se passe-t-il en coulisse ?

Le gros du travail est fait par le package **forest** qui utilise **TikZ**. Ceci permet de faire des choses sympatiques comme dans le 2^e exemple ci-dessous.

Exemple 1 – Le cas type

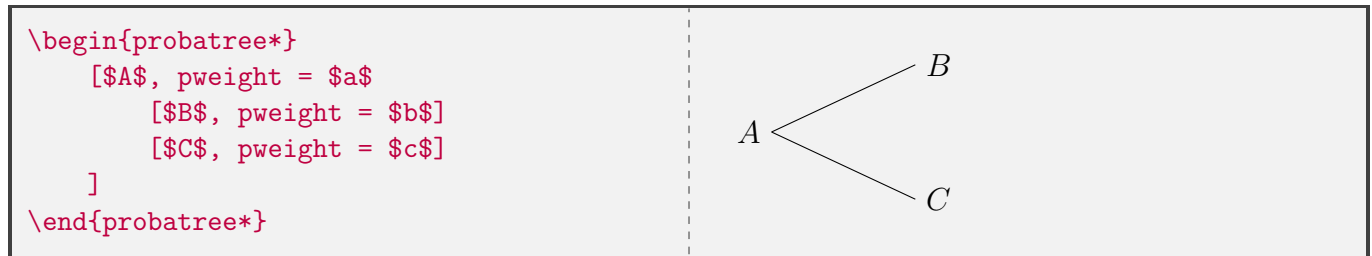
Dans le code suivant l'environnement **probatree** utilise en coulisse celui nommé **forest** du package **forest**. Des réglages spécifiques sont faits pour obtenir le résultat ci-après. À cela s'ajoutent les styles spéciaux **pweight**, **apweight** et **bpweight** qui facilitent l'écriture des pondérations sur les branches⁸.

<pre> \begin{probatree} [[A\$, pweight = \$a\$ [B\$, pweight = \$b\$] [C\$, pweight = \$c\$]] [D\$, bpweight = \$d\$ [E\$, apweight = \$e\$] [F\$, bpweight = \$f\$]]] \end{probatree} </pre>	
---	--

8. **pweight** vient de « *probability* » et « *weight* » soit « *probabilité* » et « *poids* » en anglais. Quant à **a** et **b** au début de **apweight** et **bpweight** respectivement, ils viennent de « *above* » et « *below* » soit « *dessus* » et « *dessous* » en anglais.

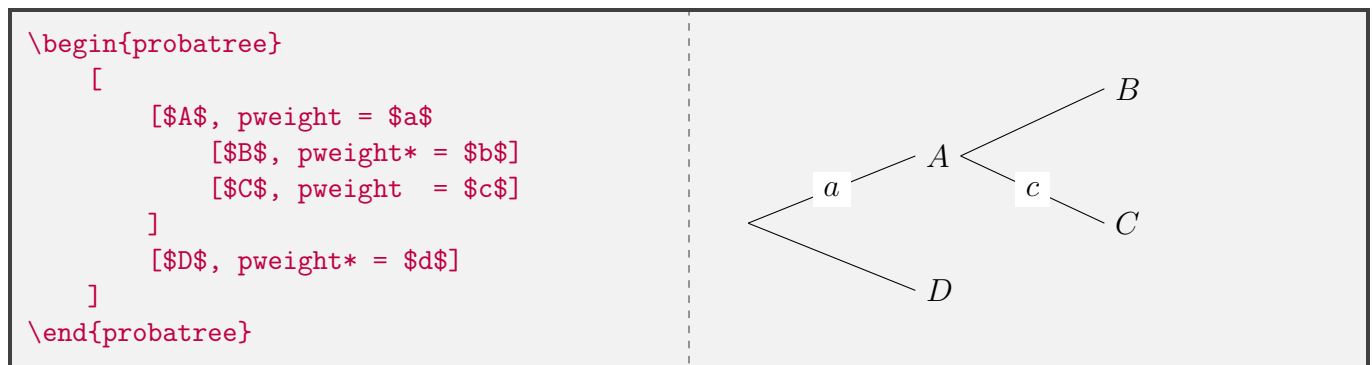
Exemple 2 – Des poids cachés partout

On peut cacher tous les poids via l'environnement étoilé `probatree*` sans avoir à retaper un arbre où les pondérations ont déjà été indiquées.



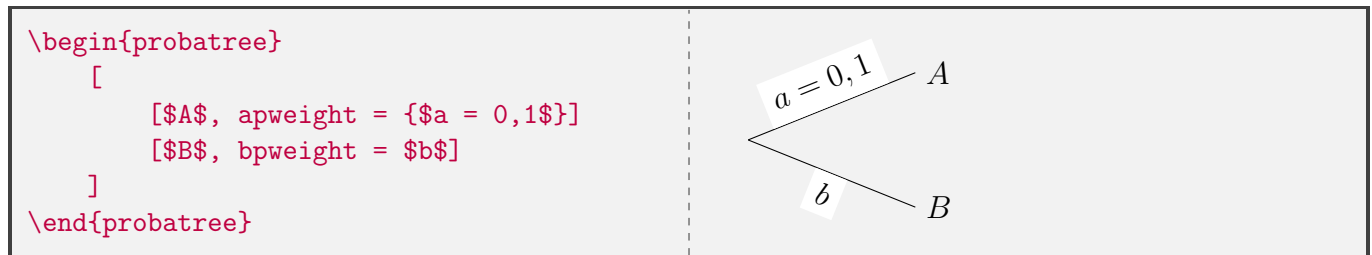
Exemple 3 – Des poids cachés localement

Pour ne cacher que certains poids, il faudra utiliser, à la main, le style `pweight*` comme dans l'exemple ci-dessous.



Exemple 4 – Un signe = et/ou une virgule dans les étiquettes

Vous ne pouvez pas utiliser directement un signe = ou une virgule dans les étiquettes des branches. L'astuce pour contourner cette limitation consiste juste à mettre le contenu de l'étiquette dans des accolades.



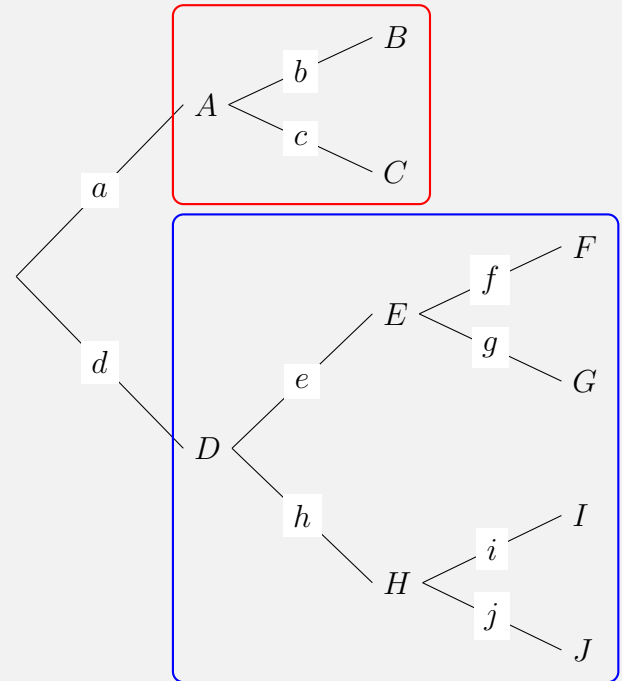
Exemple 5 – Des cadres facilement

Via la clé `frame`, il est très aisé d'encadrer un sous-arbre comme le montre l'exemple suivant. Dans l'exemple ci-après nous utilisons la bidouille `{},s sep = 1.3cm` qui évite que les cadres se superposent.

```

\begin{probatree}
[{}], s sep = 1.3cm
% Astuce pour espacer les cadres.
[$A$, pweight = $a$,
    frame = red
[$B$, pweight = $b$]
[$C$, pweight = $c$]
]
[$D$, pweight = $d$,
    frame = blue
[$E$, pweight = $e$
    [$F$, pweight = $f$]
    [$G$, pweight = $g$]
]
[$H$, pweight = $h$
    [$I$, pweight = $i$]
    [$J$, pweight = $j$]
]
]
\end{probatree}

```



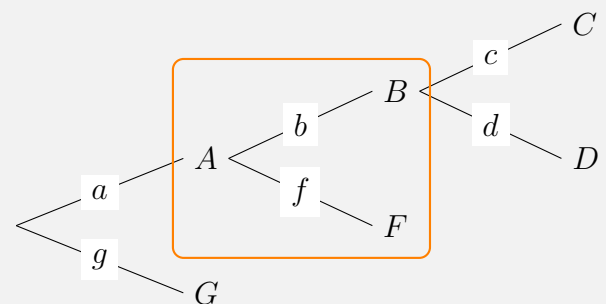
Exemple 6 – Des cadres faits à la main

En utilisant la machinerie de TiKz il est facile de décorer un arbre de probabilité comme ci-dessous où le cadre s'appuie sur trois noeuds nommés. Notons que cet exemple est tout simplement infaisable avec la clé `frame`.

```

\begin{probatree}
[
    [$A$, pweight = $a$,
        name = nA
    [$B$, pweight = $b$,
        name = nB
    [$C$, pweight = $c$]
    [$D$, pweight = $d$]
    ]
    [$F$, pweight = $f$,
        name = nF]
    ]
    [$G$, pweight = $g$]
]
\node[draw = orange,
    thick,
    rounded corners,
    fit = (nA)(nB)(nF)] {};
\end{probatree}

```



12 Arithmétique

12.1 Opérateurs de base

Pour des raisons d'expressivité des codes L^AT_EX, les opérateurs binaires `\divides`, `\notdivides` et `\modulo` ont été ajoutés comme alias respectifs de `\mid`, `\nmid` et `\bmod` qui sont proposés par le package `amssymb`.

`$10 \divides 150$` au lieu de
`$10 \mid 150$`

`$10 \notdivides 154$` au lieu de
`$10 \not\mid 154$`

`$a \equiv b \modulo p$`
`\iff`
`$p \divides (a - b)$.`

$10 \mid 150$ au lieu de $10 \mid 150$
 $10 \nmid 154$ au lieu de $10 \nmid 154$
 $a \equiv b \bmod p \iff p \mid (a - b).$

12.2 Fractions continuées

12.2.1 Fractions continuées standard

Exemple d'utilisation

Dans l'exemple suivant, la notation en ligne semble être due à Alfred Pringsheim. La notation à gauche utilise toujours le maximum d'espace pour améliorer la lisibilité.

`$ \contfrac{u_0 \mid u_1 \mid u_2 \mid \dots \mid u_n}$`
`= \contfrac*{u_0 \mid u_1 \mid u_2 \mid \dots \mid u_n}$`

$$u_0 + \frac{1}{u_1 + \frac{1}{u_2 + \frac{1}{\dots + \frac{1}{u_n}}}} = u_0 + \left| \frac{1}{u_1} \right| + \left| \frac{1}{u_2} \right| + \left| \frac{1}{\dots} \right| + \left| \frac{1}{u_n} \right|$$

12.2.2 Fractions continuées généralisées

Exemple d'utilisation

Voici comment écrire une fraction continuée généralisée.

```
$ \displaystyle
\contfracgene{a | b | c | d | e | f | \dots | y | z}
= \contfracgene*{a | b | c | d | e | f | \dots | y | z}$
```

$$a + \frac{b}{c + \frac{d}{e + \frac{f}{\dots + \frac{y}{z}}}} = a + \left\lfloor \frac{b}{c} \right\rfloor + \left\lfloor \frac{d}{e} \right\rfloor + \left\lfloor \frac{f}{\dots} \right\rfloor + \left\lfloor \frac{y}{z} \right\rfloor$$

12.2.3 Comme une fraction continuée isolée

Exemple d'utilisation

La raison d'être de la macro ci-dessous vient juste de son usage en interne.

```
$\singlecontfrac{a}{b}$
pour les fous\dots :-)
```

$\left\lfloor \frac{a}{b} \right\rfloor$ pour les fous... :-)

12.2.4 L'opérateur \mathcal{K}

Exemple d'utilisation 1

La notation suivante est proche de celle qu'utilisait Carl Friedrich Gauss.

```
$ \displaystyle
\contfracope_{k=1}^n (b_k:c_k)
= \cfrac{b_1}{c_1 + \frac{b_2}{c_2 + \frac{b_3}{\dots + \frac{b_n}{c_n}}}}
```

$$\mathcal{K}_{k=1}^n(b_k : c_k) = \frac{b_1}{c_1 + \frac{b_2}{c_2 + \frac{b_3}{\dots + \frac{b_n}{c_n}}}}$$

Remarque. La lettre \mathcal{K} vient de "kettenbruch" qui signifie "fraction continuée" en allemand.

Exemple d'utilisation 2

```
$ \displaystyle
u_0 + \contfracope_{k=1}^n (1:u_k)
= \contfrac{u_0 | u_1 | u_2 | \dots | u_n}$
```

$$u_0 + \mathcal{K}_{k=1}^n(1 : u_k) = u_0 + \frac{1}{u_1 + \frac{1}{u_2 + \frac{1}{\dots + \frac{1}{u_n}}}}$$

13 Algèbre

13.1 Polynômes, séries formelles et compagnie

13.1.1 Polynômes et fractions polynômiales

Exemple d'utilisation 1 : Polynômes

$\text{\texttt{\$}\setpoly{\RR}\{X\}}$ ou $\text{\texttt{\$}\setpoly{\RR}\{X \mid Y \mid Z\}}$	$\mathbb{R}[X]$ ou $\mathbb{R}[X; Y; Z]$
---	--

Exemple d'utilisation 2 : Fractions polynômiales

$\text{\texttt{\$}\setpolyfrac{\QQ}\{T\}}$ ou $\text{\texttt{\$}\setpolyfrac{\QQ}\{S_1 \mid S_2 \mid \dots \mid S_k\}}$	$\mathbb{Q}(T)$ ou $\mathbb{Q}(S_1; S_2; \dots; S_k)$
--	---

13.1.2 Séries formelles et leurs corps de fractions

Exemple d'utilisation 1 : Séries formelles

$\text{\texttt{\$}\setserie{\CC}\{X\}}$ ou $\text{\texttt{\$}\setserie{\CC}\{T \mid O \mid P\}}$	$\mathbb{C}[[X]]$ ou $\mathbb{C}[[T; O; P]]$
---	--

Exemple d'utilisation 2 : Corps des fractions de séries formelles

$\text{\texttt{\$}\setseriefrac{\ZZ}\{X\}}$ ou $\text{\texttt{\$}\setseriefrac{\ZZ}\{Z \mid T \mid O \mid P\}}$	$\mathbb{Z}((X))$ ou $\mathbb{Z}((Z; T; O; P))$
--	---

13.1.3 Polynômes de Laurent et séries formelles de Laurent

Exemple d'utilisation 1 : Polynômes de Laurent

Ci-dessous, la notation $\mathbb{R}\{X_1; X_2\}$ n'est pas standard.

$\text{\texttt{\$}\setpolylaurent{\RR}\{X\}}$ $= \text{\texttt{\$}\setpoly{\RR}\{X \mid X^{-1}\}}$ $\text{\texttt{\$}\setpolylaurent{\RR}\{X_1 \mid X_2\}}$ $= \text{\texttt{\$}\setpoly{\RR}\{X_1 \mid X_1^{-1} \mid X_2 \mid X_2^{-1}\}}$	$\mathbb{R}\{X\} = \mathbb{R}[X; X^{-1}]$ $\mathbb{R}\{X_1; X_2\} = \mathbb{R}[X_1; X_1^{-1}; X_2; X_2^{-1}]$
--	--

Exemple d'utilisation 2 : Séries formelles de Laurent

Ci-dessous, la notation $\mathbb{Q}\{\{X_1; X_2\}\}$ n'est pas standard.

```

 $\setseries{QQ}{X}$ 
 $= \setseries{QQ}{X \mid X^{-1}}$ 

```

```

 $\setseries{QQ}{X_1 \mid X_2}$ 
 $= \setseries{QQ}{X_1 \mid X_1^{-1} \mid X_2 \mid X_2^{-1}}$ 

```

$$\mathbb{Q}\{\{X\}\} = \mathbb{Q}[[X; X^{-1}]]$$

$$\mathbb{Q}\{\{X_1; X_2\}\} = \mathbb{Q}[[X_1; X_1^{-1}; X_2; X_2^{-1}]]$$

13.2 Matrices

Tout le boulot ou presque est fait par l'excellent package `nicematrix` auquel on impose l'option `transparent` avec l'ajout d'une macro « *maison* » à but pédagogique. Veuillez vous reporter à la documentation de `nicematrix` pour savoir comment s'y prendre en général.

13.2.1 Calculs expliqués des déterminants 2×2

Dans l'exemple suivant, le préfixe `c` est pour `c`-alculer et `two` est pour « *deux* » en anglais.

```

 $\cdettwo{a}{b}{c}{d}$ 

```

$$\begin{vmatrix} a & b \\ c & d \end{vmatrix}$$

13.2.2 Quelques exemples pour bien démarrer

Exemple 1 (cf. la documentation de `nicematrix`)

```

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

```

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

Exemple 2

```

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

```

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

Exemple 2

```

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

```

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

Exemple 3 (cf. la documentation de nicematrix)

```

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

\tikz[remember picture,
  overlay]
\draw[red] (mymatrix-2-2) circle (2.5mm);

```

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

Exemple 4 (cf. la documentation de nicematrix)

```

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

```

$$\left(\begin{array}{cccc:c} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \end{array} \right)$$

Exemple 5 proposé par l’auteur de nicematrix suite à une discussion par mail

```
% Besoin du package ‘‘ifthen’’.
\newcommand\aij{%
  a_{\arabic{iRow}\arabic{jCol}}%
}

$\begin{bNiceArray}%
  {*{5}}>{%
    \ifthenelse{\value{iRow}>0}{\aij}{}%
  }C}[
  first-col,
  first-row,
  code-for-first-row
    = \mathbf{\arabic{jCol}},
  code-for-first-col
    = \mathbf{\arabic{iRow}}
  ]

  & & & & \\
  & & & & \\
  & & & & \\
\end{bNiceArray}$
```

$$\begin{matrix} & \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} \\ \mathbf{1} & a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ \mathbf{2} & a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \end{matrix}$$

Exemple 6 avec des calculs automatiques

```
\newcounter{cntaij}
\newcommand\aij{%
  \setcounter{cntaij}{\value{iRow}}%
  \addtocounter{cntaij}{\value{jCol}}%
  \addtocounter{cntaij}{-1}%
  \arabic{cntaij}%
}
Si $a_{ij} = i + j - 1$ alors

$(a_{ij})_{1 \leq i \leq 3, 1 \leq j \leq 5}$

=

\begin{bNiceArray}{*{5}}>{\aij}C}
  & & & & \\
  & & & & \\
  & & & & \\
\end{bNiceArray}$
```

Si $a_{ij} = i + j - 1$ alors

$$(a_{ij})_{1 \leq i \leq 3, 1 \leq j \leq 5} = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 6 \\ 3 & 4 & 5 & 6 & 7 \end{bmatrix}$$

14 Historique

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

2020-06-08 Nouvelle version mineure 0.7.0-beta.

- En analyse, il y a eu les changements suivants.
 - Pour éviter des conflits avec d'autres packages les renommages suivants ont été faits.
 - `\ch` et `\ach` sont devenus `\fch` et `\afch` où `f` est pour `f-rech`.
 - `\sh` et `\ash` sont devenus `\fsh` et `\afsh`.
 - `\th` et `\ath` sont devenus `\fth` et `\afth`.
 - Les macros `\acosh`, `\asinh` et `\atanh` été ajoutées.
 - `\derpar` et `\derpar*` servent à rédiger des dérivées avec des parenthèses extensibles. En coulisse, `\derpow` et `\derpow*` sont appelées. Pour utiliser des parenthèses non extensibles, on passera par `\sderpar` et `\sderpar*` où `s` est pour `s-mall`.
 - Ajout de `\stdint` pour rendre public l'opérateur intégral proposé par défaut par L^AT_EX.
- En géométrie, il y a eu une suppression et trois ajouts.
 - La macro `\pts` a été supprimée car sans signification sémantique puisqu'un point peut être nommé avec deux lettres.
 - Les macros `\gline` et `\pgline` servent à indiquer des droites.
 - La macro `\hgline`, avec `h` pour `h-alf`, est pour les demi-droites.
 - La macro `\segment` est utile pour les segments.
- En logique, voici les améliorations apportées.
 - Pour les inégalités, on peut maintenant utiliser le décorateur `plot`.
 - Ajout des versions négatives des opérateurs logiques verticaux.
- Pour les probabilités, il y a eu les modifications ci-après.
 - Ajout de `\proba` pour écrire des probabilités.
 - Le comportement de `\probacond` a été modifié pour le rendre plus logique.

2019-10-21 Nouvelle version sous-mineure 0.6.3-beta.

- Pour les intervalles, `\CSinterval` a été déplacée dans le package `lyalgo` disponible à l'adresse <https://github.com/bc-latex/ly-algo>.
- En logique, il y a eu les modifications suivantes.
 - `\eqdef**` a été supprimé. Voir la macro `\Store*` du package `lyalgo`.
 - Différentes versions de l'opérateur \exists via `\existssone` et `\existmulti` avec leurs versions négatives `\nexistsone` et `\nexistmulti`.
 - Deux nouvelles macros `\eqplot` et `\eqappli` pour indiquer une équation de courbe et l'application d'une identité à des variables. Ceci s'accompagne de l'ajout des macros `\textopplot` et `\textopappli`.

- Ajout des formes négatives `\niff`, `\nimplies` et `\nliesimp`.
- Les décorations `cons`, `appli` et `choice` sont utilisables avec les opérateurs `\iff`, `\implies` et `\liesimp` et leurs formes négatives.
- Une macro `\textoptest` a été ajoutée afin de rendre personnalisable tous les textes décorant les symboles.
- En analyse, il y a eu les renommages suivants.
 - `\hypergeo` est devenu `\seqhypergeo`.
 - `\suprageo` est devenu `\seqsuprageo`.
- En géométrie, `\notparallel` est devenu `\nparallel`.

2019-10-14 Nouvelle version sous-mineure 0.6.2-beta.

- En algèbre, il y a eu les renommages ci-dessous qui avaient été oubliés.
 - `\polyset` est devenu `\setpoly`.
 - `\polyfracset` est devenu `\setpolyfrac`.
 - `\serieset` est devenu `\setserie`.
 - `\seriefracset` est devenu `\setseriefrac`.
 - `\polylaurentset` est devenu `\setpolylaurent`.
 - `\serielaurentset` est devenu `\setserielaurent`.

2019-10-13 Nouvelle version sous-mineure 0.6.1-beta.

- En logique, la macro `\explain` possède maintenant un argument optionnel pour indiquer l'espacement avant le symbole. Ceci s'accompagne de la suppression des macros obsolètes `\explain*` et `\textexplainspacebefore`.
- En probabilité, voici ce qui a évolué.
 - Les macros `\probacond` et `\probacond*` n'ont plus d'argument optionnel. Pour obtenir l'écriture fractionnaire, il faut utiliser `\probacond**` ou `\dprobacond**`.
 - Les environnements `probatree` et `probatree*` ont trois nouvelles clés. La clé `frame` permet d'encadrer un sous-arbre, et les clés `apweight` et `bpweight` permettent d'écrire des poids dessus/dessous une branche.
- Pour les ensembles, il y a eu les renommages suivants par souci de cohérence.
 - `\algeset` est devenu `\setalge`.
 - `\geoset` est devenu `\setgeo`.
 - `\geneset` est devenu `\setgene`.
 - `\probacond` est devenu `\setproba`.
 - `\specialset` est devenu `\setspecial`.

2019-10-10 Nouvelle version mineure 0.6.0-beta.

- Des nouveaux outils spécifiques aux probabilités.
 - Les macros `\probacond` et `\probacond*` servent à écrire des probabilités conditionnelles.
 - Les environnements `probatree` et `probatree*` simplifient la production d'arbres probabilistes pondérés ou non.

- En géométrie, la macro `\notparallel` a été rajoutée.
- Un nouveau type d'intervalle pour l'informatique théorique via la macro `\CSinterval` afin d'obtenir quelque chose comme $a..b$.
- En logique, il y a deux nouvelles macros sémantiques `\neqid` et `\eqchoice`.

2019-09-27 Nouvelle version mineure 0.5.0-beta.

- Ajout des macros `\dsum` et `\dprod` qui sont vis à vis de `\sum` et `\prod` des équivalents de `\dfrac` pour `\frac`.
- En arithmétique, ajout des opérateurs `\divides`, `\notdivides` et `\modulo`.
- En géométrie, une nouvelle macro et un opérateur modifié.
 - `\pts` permet d'indiquer plusieurs points.
 - `\parallel` utilise des obliques pour symboliser le parallélisme au lieu de barres verticales.
- En logique, il y a les nouveautés suivantes.
 - La version doublement étoilée `\eqdef**` donne une deuxième écriture symbolique d'un symbole égal de type définition (*cette notation vient du langage B*).
 - Ajout de `\liesimp` comme alias de `\Longleftarrow`.
 - Les macros `\vimplies`, `\viff` et `\vliesimp` sont des versions verticales de `\implies`, `\iff` et `\liesimp`.
 - Comme pour les égalités, il existe les macros `\impliestest`, `\iffhyp` ... etc.

2019-09-06 Nouvelle version mineure 0.4.0-beta.

- Dans « *Logique et fondements* », différents types de signes d'inéquation et de non égalité pour des cas de test, d'hypothèse faite et de condition à vérifier.
- Intégration du package `tkz-tab` pour rédiger des tableaux de variations et de signes.
- Intégration du package `nicematrix` pour écrire des matrices.

15 Toutes les fiches techniques

15.1 Deux séparateurs d'arguments par défaut

`\lymathsep <macro>` (Sans argument)
`\lymathsubsep <macro>` (Sans argument)

15.2 Quelques gestions d'espaces

15.2.1 Fractions

`\frac <macro>` (2 Arguments)
`\dfrac <macro>` (2 Arguments)
`\stdfrac <macro>` (2 Arguments)
`\stddfraction <macro>` (2 Arguments)

- Argument 1: le numérateur.
- Argument 2: le dénominateur.

15.2.2 Racines n-ièmes

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

- Option: l'indice à indiquer pour une racine n-ième.
- Argument: le radicande, c'est à dire ce qui sera écrit sous le radical.

15.2.3 Sommes et produits

Les macros suivantes sans argument ont un comportement spécifique vis à vis des mises en index et en exposant.

`\dprod <macro>` (Sans argument)
`\dsum <macro>` (Sans argument)

15.3 Logique et fondements

15.3.1 Les textes pour les opérateurs de « comparaison algébrique » et de logique

`\textopappli <macro>` (Sans argument)
`\textopchoice <macro>` (Sans argument)
`\textopcond <macro>` (Sans argument)
`\textopcons <macro>` (Sans argument)
`\textopdef <macro>` (Sans argument)
`\textophyp <macro>` (Sans argument)
`\textopid <macro>` (Sans argument)
`\textopplot <macro>` (Sans argument)
`\textoptest <macro>` (Sans argument)

15.3.2 Les opérateurs de « comparaison algébrique »

```
\eqappli <macro> (Sans argument)
\eqchoice <macro> (Sans argument)
\eqcond <macro> (Sans argument)
\eqcons <macro> (Sans argument)
\eqdef <macro> (Sans argument)
\eqdef* <macro> (Sans argument)
\eqhyp <macro> (Sans argument)
\eqid <macro> (Sans argument)
\eqid* <macro> (Sans argument)
\eqplot <macro> (Sans argument)
\eqtest <macro> (Sans argument)
```

```
\neqappli <macro> (Sans argument)
\neqchoice <macro> (Sans argument)
\neqcond <macro> (Sans argument)
\neqcons <macro> (Sans argument)
\neqhyp <macro> (Sans argument)
\neqid <macro> (Sans argument)
\neqtest <macro> (Sans argument)
```

```
\lappli <macro> (Sans argument)
\lchoice <macro> (Sans argument)
\lcond <macro> (Sans argument)
\lcons <macro> (Sans argument)
\lhyp <macro> (Sans argument)
\lplot <macro> (Sans argument)
\ltest <macro> (Sans argument)
```

```
\gappli <macro> (Sans argument)
\gchoice <macro> (Sans argument)
\gcond <macro> (Sans argument)
\gcons <macro> (Sans argument)
\ghyp <macro> (Sans argument)
\gplot <macro> (Sans argument)
\gtest <macro> (Sans argument)
```

```
\leqappli <macro> (Sans argument)
\leqchoice <macro> (Sans argument)
\leqcond <macro> (Sans argument)
\leqcons <macro> (Sans argument)
\leqhyp <macro> (Sans argument)
\leqplot <macro> (Sans argument)
\leqtest <macro> (Sans argument)
```

15.3.3 Les opérateurs de logique

`\iff <macro> (Sans argument)`
`\iffappli <macro> (Sans argument)`
`\iffchoice <macro> (Sans argument)`
`\iffcond <macro> (Sans argument)`
`\iffcons <macro> (Sans argument)`
`\iffhyp <macro> (Sans argument)`
`\ifftest <macro> (Sans argument)`

`\notiff <macro> (Sans argument)`
`\notiffappli <macro> (Sans argument)`
`\notiffchoice <macro> (Sans argument)`
`\notiffcond <macro> (Sans argument)`
`\notiffcons <macro> (Sans argument)`
`\notiffhyp <macro> (Sans argument)`
`\notifftest <macro> (Sans argument)`

`\implies <macro> (Sans argument)`
`\impliesappli <macro> (Sans argument)`
`\implieschoice <macro> (Sans argument)`
`\impliescond <macro> (Sans argument)`
`\impliescons <macro> (Sans argument)`
`\implieshyp <macro> (Sans argument)`
`\impliestest <macro> (Sans argument)`

`\notimplies <macro> (Sans argument)`
`\notimpliesappli <macro> (Sans argument)`
`\notimplieschoice <macro> (Sans argument)`
`\notimpliescond <macro> (Sans argument)`
`\notimpliescons <macro> (Sans argument)`
`\notimplieshyp <macro> (Sans argument)`
`\notimpliestest <macro> (Sans argument)`

`\liesimp <macro> (Sans argument)`
`\liesimpappli <macro> (Sans argument)`
`\liesimpchoice <macro> (Sans argument)`
`\liesimpcond <macro> (Sans argument)`
`\liesimpcons <macro> (Sans argument)`
`\liesimphyp <macro> (Sans argument)`
`\liesimptest <macro> (Sans argument)`

`\notliesimp <macro> (Sans argument)`
`\notliesimpappli <macro> (Sans argument)`
`\notliesimpchoice <macro> (Sans argument)`
`\notliesimpcond <macro> (Sans argument)`
`\notliesimpcons <macro> (Sans argument)`

`\notliesimphyp <macro>` (Sans argument)
`\notliesimptest <macro>` (Sans argument)

15.3.4 Les opérateurs de logique « verticaux »

`\viff <macro>` (Sans argument)
`\notviff <macro>` (Sans argument)
`\vimplies <macro>` (Sans argument)
`\notvimplies <macro>` (Sans argument)
`\vliesimp <macro>` (Sans argument)
`\notvliesimp <macro>` (Sans argument)

15.3.5 Les textes pour détailler un raisonnement – Version pour le lycée et après

`\textexplainleft <macro>` (Sans argument)
`\textexplainright <macro>` (Sans argument)
`\textexplainspacein <macro>` (Sans argument)

15.3.6 Détailler un raisonnement – Version pour le lycée et après

`explain <env>` [1 Option]

— Option: le symbole à utiliser dans l’environnement, la valeur par défaut étant = .

`\explnext <macro>` [1 Option] (1 Argument) où `expl = expl-ain`

— Option: le symbole à utiliser pour une explication, la valeur par défaut étant celle du symbole de l’environnement `explain` où `\explnext` est utilisé.

— Argument: le texte de l’explication qui peut être vide si aucune explication n’est à afficher.

`\explnext* <macro>` [1 Option] (2 Arguments) où `expl = expl-ain`

— Option: le symbole à utiliser pour une explication, la valeur par défaut étant celle du symbole de l’environnement `explain` où `\explnext` est utilisé.

— Argument 1: le texte de l’explication pour la 1^{re} ligne. Ce texte peut être vide (*voir l’environnement `aexplain` pour la raison de ceci*).

— Argument 2: le texte de l’explication pour la 2^e ligne. Ce texte peut être vide (*voir l’environnement `aexplain` pour la raison de ceci*).

15.3.7 Détailler un raisonnement – Version pour les collégiens

Voir les macros `\textexplainleft`, `\textexplainright`, `\textexplainspacein`, `\explnext` ainsi que `\explnext*` dans les sections 15.3.5 et 15.3.6.

`aexplain <env>` [1 Option] où `a = a-rrow`
`aexplain* <env>` [1 Option] où `a = a-rrow`

— Option: le symbole à utiliser dans l’environnement, la valeur par défaut étant = .

`\explsymb <macro>` (Sans argument)

ATTENTION! Cette macro n'est utilisable qu'au début de l'environnement *aexplain**.

15.3.8 Des versions alternatives du quantificateur \exists

`\existmulti <macro>` (1 Argument)

`\nexistmulti <macro>` (1 Argument)

— Argument 1 : une écriture mathématique servant à préciser la portée du quantificateur.

`\existsone <macro>` (Sans argument)

`\nexistsone <macro>` (Sans argument)

15.4 Ensembles et applications

15.4.1 Ensembles versus accolades

`\setgene <macro>` (1 Argument)

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

— Argument : la définition de l'ensemble.

15.4.2 Ensembles pour la géométrie

`\setgeo <macro>` (1 Argument)

— Argument : un seul caractère ASCII indiquant un ensemble géométrique.

`\setgeo* <macro>` (2 Arguments)

— Argument 1 : un seul caractère ASCII indiquant \mathcal{U} dans le nom \mathcal{U}_d d'un ensemble géométrique.

— Argument 2 : un texte donnant d dans le nom \mathcal{U}_d d'un ensemble géométrique.

15.4.3 Ensembles probabilistes

`\setproba <macro>` (1 Argument)

— Argument : un seul caractère ASCII majuscule indiquant un ensemble probabiliste.

`\setproba* <macro>` (2 Arguments)

— Argument 1 : un seul caractère ASCII majuscule indiquant \mathcal{U} dans le nom \mathcal{U}_d d'un ensemble probabiliste.

— Argument 2 : un texte donnant d dans le nom \mathcal{U}_d d'un ensemble probabiliste.

15.4.4 Ensembles pour l'algèbre générale

`\setalge <macro>` (1 Argument)

— Argument : soit l'une des lettres h et k , soit un seul caractère ASCII majuscule indiquant un ensemble de type anneau ou corps.

`\setalge* <macro> (2 Arguments)`

— **Argument 1**: un seul caractère ASCII indiquant \mathbb{U} dans le nom \mathbb{U}_d d’un ensemble de type anneau ou corps.

— **Argument 2**: un texte donnant d dans le nom \mathbb{U}_d d’un ensemble de type anneau ou corps.

15.4.5 Ensembles classiques

`\NN <macro> (Sans argument)`

`\NNs <macro> (Sans argument)`

`\PP <macro> (Sans argument)`

`\ZZ <macro> (Sans argument)`

`\ZZn <macro> (Sans argument)`

`\ZZp <macro> (Sans argument)`

`\ZZs <macro> (Sans argument)`

`\ZZsn <macro> (Sans argument)`

`\ZZsp <macro> (Sans argument)`

`\DD <macro> (Sans argument)`

`\DDn <macro> (Sans argument)`

`\DDp <macro> (Sans argument)`

`\DDs <macro> (Sans argument)`

`\DDsn <macro> (Sans argument)`

`\DDsp <macro> (Sans argument)`

`\QQ <macro> (Sans argument)`

`\QQn <macro> (Sans argument)`

`\QQp <macro> (Sans argument)`

`\QQs <macro> (Sans argument)`

`\QQsn <macro> (Sans argument)`

`\QQsp <macro> (Sans argument)`

`\RR <macro> (Sans argument)`

`\RRn <macro> (Sans argument)`

`\RRp <macro> (Sans argument)`

`\RRs <macro> (Sans argument)`

`\RRsn <macro> (Sans argument)`

`\RRsp <macro> (Sans argument)`

`\CC <macro> (Sans argument)`

`\CCs <macro> (Sans argument)`

`\HH <macro> (Sans argument)`

`\HHs <macro>` (Sans argument)

`\00 <macro>` (Sans argument)

`\00s <macro>` (Sans argument)

15.4.6 Des suffixes à la carte

`\setspecial <macro>` (2 Arguments)

`\setspecial* <macro>` (2 Arguments)

— Argument 1: l'ensemble à "suffixer".

— Argument 2: l'un des suffixes `n`, `p`, `s`, `sn` ou `sp`.

15.4.7 Intervalles réels - Notation française (?)

Pour toutes les macros ci-dessous, la version non étoilée produit des délimiteurs qui s'étirent si besoin verticalement, tandis que la version étoilée ne le fait pas.

`\intervalC0 <macro>` (2 Arguments)

`\intervalC0* <macro>` (2 Arguments)

— Argument 1: borne inférieure a de l'intervalle $[a; b[$.

— Argument 2: borne supérieure b de l'intervalle $[a; b[$.

`\intervalC <macro>` (2 Arguments)

`\intervalC* <macro>` (2 Arguments)

— Argument 1: borne inférieure a de l'intervalle $[a; b]$.

— Argument 2: borne supérieure b de l'intervalle $[a; b]$.

`\interval0 <macro>` (2 Arguments)

`\interval0* <macro>` (2 Arguments)

— Argument 1: borne inférieure a de l'intervalle $]a; b[$.

— Argument 2: borne supérieure b de l'intervalle $]a; b[$.

`\interval0C <macro>` (2 Arguments)

`\interval0C* <macro>` (2 Arguments)

— Argument 1: borne inférieure a de l'intervalle $]a; b]$.

— Argument 2: borne supérieure b de l'intervalle $]a; b]$.

15.4.8 Intervalles réels - Notation américaine

Pour toutes les macros ci-dessous, la version non étoilée produit des délimiteurs qui s'étirent si besoin verticalement, tandis que la version étoilée ne le fait pas.

`\intervalCP <macro> (2 Arguments)`
`\intervalCP* <macro> (2 Arguments)`

- Argument 1: borne inférieure a de l'intervalle $[a; b)$.
 - Argument 2: borne supérieure b de l'intervalle $[a; b)$.
-

`\intervalP <macro> (2 Arguments)`
`\intervalP* <macro> (2 Arguments)`

- Argument 1: borne inférieure a de l'intervalle $(a; b)$.
 - Argument 2: borne supérieure b de l'intervalle $(a; b)$.
-

`\intervalPC <macro> (2 Arguments)`
`\intervalPC* <macro> (2 Arguments)`

- Argument 1: borne inférieure a de l'intervalle $(a; b]$.
- Argument 2: borne supérieure b de l'intervalle $(a; b]$.

15.4.9 Intervalles discrets d'entiers

Pour toutes les macros ci-dessous, la version non étoilée produit des délimiteurs qui s'étirent si besoin verticalement, tandis que la version étoilée ne le fait pas.

`\ZintervalC0 <macro> (2 Arguments)`
`\ZintervalC0* <macro> (2 Arguments)`

- Argument 1: borne inférieure a de l'intervalle $\llbracket a; b \rrbracket$.
 - Argument 2: borne supérieure b de l'intervalle $\llbracket a; b \rrbracket$.
-

`\ZintervalC <macro> (2 Arguments)`
`\ZintervalC* <macro> (2 Arguments)`

- Argument 1: borne inférieure a de l'intervalle $\llbracket a; b \rrbracket$.
 - Argument 2: borne supérieure b de l'intervalle $\llbracket a; b \rrbracket$.
-

`\Zinterval0 <macro> (2 Arguments)`
`\Zinterval0* <macro> (2 Arguments)`

- Argument 1: borne inférieure a de l'intervalle $\llbracket a; b \rrbracket$.
 - Argument 2: borne supérieure b de l'intervalle $\llbracket a; b \rrbracket$.
-

`\Zinterval0C <macro> (2 Arguments)`
`\Zinterval0C* <macro> (2 Arguments)`

- Argument 1: borne inférieure a de l'intervalle $\llbracket a; b \rrbracket$.
- Argument 2: borne supérieure b de l'intervalle $\llbracket a; b \rrbracket$.

15.4.10 Unions et intersections

`\dcap <macro>` (Sans argument)
`\dcup <macro>` (Sans argument)
`\dsqcup <macro>` (Sans argument)

15.4.11 Cardinal, image et compagnie

`\card <macro>` (Sans argument)
`\card* <macro>` (Sans argument)

`\dom <macro>` (Sans argument)
`\codom <macro>` (Sans argument)
`\im <macro>` (Sans argument)
`\ker <macro>` (Sans argument)

15.4.12 Application totale, partielle, injective, surjective et/ou bijective

<code>\to <macro></code> (Sans argument)	<code>\pto <macro></code> (Sans argument)
<code>\onetoone <macro></code> (Sans argument)	<code>\ponetoone <macro></code> (Sans argument)
<code>\onto <macro></code> (Sans argument)	<code>\ponto <macro></code> (Sans argument)
<code>\biject <macro></code> (Sans argument)	<code>\pbiject <macro></code> (Sans argument)

15.5 Géométrie

15.5.1 Points

`\pt <macro>` (1 Argument)

— Argument : un texte donnant le nom d'un point.

`\pt* <macro>` (2 Arguments)

— Argument 1 : un texte indiquant UP dans le nom UP_{down} d'un point.

— Argument 2 : un texte indiquant *down* dans le nom UP_{down} d'un point.

15.5.2 Lignes

`\gline <macro>` [1 Option] (2 Arguments) où $g = g\text{-eometry}$
`\pgline <macro>` [1 Option] (2 Arguments) où $p = p\text{-oint}$ et $g = g\text{-eometry}$

— Option : pour indiquer les parenthèses ou crochets à utiliser, les valeurs possibles étant 0, valeur par défaut, C, CO et OC.

— Argument 1 : le 1^{er} point géométrique.

— Argument 2 : le 2^e point géométrique.

`\hgline <macro>` (2 Arguments) où $h = h\text{-alf}$ et $g = g\text{-eometry}$
`\phgline <macro>` (2 Arguments) où $p = p\text{-oint}$, $h = h\text{-alf}$ et $g = g\text{-eometry}$
`\segment <macro>` (2 Arguments)
`\psegment <macro>` (2 Arguments) où $p = p\text{-oint}$

- Argument 1: le 1^{er} point géométrique.
- Argument 2: le 2^e point géométrique.

15.5.3 Droites parallèles ou non

`\parallel <macro>` (Sans argument)
`\nparallel <macro>` (Sans argument)
`\stdparallel <macro>` (Sans argument)
`\stdnparallel <macro>` (Sans argument)

15.5.4 Vecteurs

`\vect <macro>` (1 Argument)
 — Argument: un texte donnant le nom d'un vecteur.

`\vect* <macro>` (2 Arguments)

- Argument 1: un texte indiquant *up* dans le nom $\overrightarrow{up}_{down}$ d'un vecteur.
- Argument 2: un texte indiquant *down* dans le nom $\overrightarrow{up}_{down}$ d'un vecteur.

15.5.5 Norme

`\norm <macro>` (1 Argument)
`\norm* <macro>` (1 Argument)
 — Argument: le vecteur sur lequel appliquer la norme.

15.5.6 Produit scalaire – Écriture minimaliste

`\dotprod <macro>` (2 Arguments)
 — Argument 1: le 1^{er} vecteur qu'il faut taper via la macro `\vect`.
 — Argument 2: le 2^e vecteur qu'il faut taper via la macro `\vect`.

`\vdotprod <macro>` (2 Arguments) où $v = v\text{-ector}$

- Argument 1: le nom du 1^{er} vecteur sans utiliser la macro `\vect`.
- Argument 2: le nom du 2^e vecteur sans utiliser la macro `\vect`.

15.5.7 Produit scalaire – Écriture « à la physicienne »

`\adotprod <macro>` (2 Arguments) où $a = a\text{-ngle}$
`\adotprod* <macro>` (2 Arguments)
 — Argument 1: le 1^{er} vecteur qu'il faut taper via la macro `\vect`.
 — Argument 2: le 2^e vecteur qu'il faut taper via la macro `\vect`.

`\vadotprod <macro>` (2 Arguments) où $a = a\text{-ngle}$ et $v = v\text{-ector}$

- Argument 1: le nom du 1^{er} vecteur sans utiliser la macro `\vect`.
- Argument 2: le nom du 2^e vecteur sans utiliser la macro `\vect`.

15.5.8 Produit vectoriel

`\crossprod <macro> (2 Arguments)`

— Argument 1: le 1^{er} vecteur qu'il faut taper via la macro `\vect`.

— Argument 2: le 2^e vecteur qu'il faut taper via la macro `\vect`.

`\vcrossprod <macro> (2 Arguments)` où `v = v-ector`

— Argument 1: le nom du 1^{er} vecteur sans utiliser la macro `\vect`.

— Argument 2: le nom du 2^e vecteur sans utiliser la macro `\vect`.

`\ccrossprod <macro> (8 Arguments)` où `c = c-ordinates`

— Argument 1: le 1^{er} vecteur qu'il faut taper via la macro `\vect`.

— Argument 2: la 1^{re} coordonnée du 1^{er} vecteur.

— Argument 3: la 2^e coordonnée du 1^{er} vecteur.

— Argument 4: la 3^e coordonnée du 1^{er} vecteur.

— Argument 5: le 2^e vecteur qu'il faut taper via la macro `\vect`.

— Argument 6: la 1^{re} coordonnée du 2^e vecteur.

— Argument 7: la 2^e coordonnée du 2^e vecteur.

— Argument 8: la 3^e coordonnée du 2^e vecteur.

`\vccrossprod <macro> (8 Arguments)` où `c = c-ordinates` et `v = v-ector`

— Argument 1: le 1^{er} vecteur sans utiliser la macro `\vect`.

— Argument 2: la 1^{re} coordonnée du 1^{er} vecteur.

— Argument 3: la 2^e coordonnée du 1^{er} vecteur.

— Argument 4: la 3^e coordonnée du 1^{er} vecteur.

— Argument 5: le 2^e vecteur sans utiliser la macro `\vect`.

— Argument 6: la 1^{re} coordonnée du 2^e vecteur.

— Argument 7: la 2^e coordonnée du 2^e vecteur.

— Argument 8: la 3^e coordonnée du 2^e vecteur.

15.5.9 Plan – Déterminant de deux vecteurs

`\cdetplane <macro> (6 Arguments)` où `c = c-ordinates`

`\cdetplane* <macro> (6 Arguments)` où `c = c-ordinates`

— Argument 1: le 1^{er} vecteur qu'il faut taper via la macro `\vect`.

— Argument 2: la 1^{re} coordonnée du 1^{er} vecteur.

— Argument 3: la 2^e coordonnée du 1^{er} vecteur.

— Argument 4: le 2^e vecteur qu'il faut taper via la macro `\vect`.

— Argument 5: la 1^{re} coordonnée du 2^e vecteur.

— Argument 6: la 2^e coordonnée du 2^e vecteur.

`\vcdetplane <macro> (6 Arguments)` où `c` = c-ordinates et `v` = v-ector
`\vcdetplane* <macro> (6 Arguments)` où `c` = c-ordinates et `v` = v-ector

— Argument 1: le 1^{er} vecteur sans utiliser la macro `\vect`.

— Argument 2: la 1^{re} coordonnée du 1^{er} vecteur.

— Argument 3: la 2^e coordonnée du 1^{er} vecteur.

— Argument 4: le 2^e vecteur sans utiliser la macro `\vect`.

— Argument 5: la 1^{re} coordonnée du 2^e vecteur.

— Argument 6: la 2^e coordonnée du 2^e vecteur.

15.5.10 Coordonnées

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

— Argument: l'argument est une suite de "morceaux" séparés par des barres `|`, chaque morceau étant une coordonnée. Il peut n'y avoir qu'un seul morceau.

`\vcoord <macro> (1 Argument)` où `v` = v-ertical
`\vcoord* <macro> (1 Argument)` pour des crochets à la place de parenthèses

— Argument: l'argument est une suite de "morceaux" séparés par des barres `|`, chaque morceau étant une coordonnée. Il peut n'y avoir qu'un seul morceau.

15.5.11 Nommer un repère

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

— Argument: l'argument est une suite de "morceaux" séparés par des barres `|`.

- Le premier morceau est l'origine du repère.
- Les morceaux suivants sont des points ou des vecteurs qui "définissent" chaque axe.

`\paxes <macro> (1 Argument)` où `p` = p-oint

— Argument: l'argument est une suite de "morceaux" séparés par des barres `|`.

- Le premier morceau est le nom de l'origine du repère sur laquelle la macro-commande `\pt` sera automatiquement appliquée.
- Viennent ensuite les noms des points "définissant" chaque axe. Pour chacun de ces points la macro-commande `\pt` sera automatiquement appliquée.

`\vaxes <macro> (1 Argument)` où `v` = v-ector

— Argument: l'argument est une suite de "morceaux" séparés par des barres `|`.

- Le premier morceau est l'origine du repère.
- Viennent ensuite les noms des vecteurs "définissant" chaque axe. Pour chacun de ces vecteurs la macro-commande `\vect` sera automatiquement appliquée.

`\pvaxes <macro> (3 Arguments)` où $pv = p + v$

— Argument: l'argument est une suite de "morceaux" séparés par des barres |.

- Le premier morceau est le nom de l'origine du repère sur laquelle la macro-commande `\pt` sera automatiquement appliquée.
- Viennent ensuite les noms des vecteurs "définissant" chaque axe. Pour chacun de ces vecteurs la macro-commande `\vect` sera automatiquement appliquée.

15.5.12 Arcs circulaires

`\circarc <macro> (1 Argument)` où $\text{circ} = \text{circ-ular}$

— Argument: un texte donnant le nom d'un arc circulaire.

`\circarc* <macro> (2 Arguments)` où $\text{circ} = \text{circ-ular}$

— Argument 1: un texte indiquant *up* dans le nom \widehat{up}_{down} d'un arc circulaire.

— Argument 2: un texte indiquant *down* dans le nom \widehat{up}_{down} d'un arc circulaire.

15.5.13 Angles géométriques « intérieurs »

`\anglein <macro> (1 Argument)`

— Argument: un texte donnant le nom d'un angle intérieur.

`\anglein* <macro> (2 Arguments)`

— Argument 1: un texte indiquant *up* dans le nom \widehat{up}_{down} d'un angle intérieur.

— Argument 2: un texte indiquant *down* dans le nom \widehat{up}_{down} d'un angle intérieur.

15.5.14 Angles orientés de vecteurs

`\angleorient <macro> (2 Arguments)`

`\hangleorient <macro> (2 Arguments)` où $h = h\text{-at}$

— Argument 1: le premier vecteur qu'il faut taper via la macro `\vect`.

— Argument 2: le second vecteur qu'il faut taper via la macro `\vect`.

`\vangleorient <macro> (2 Arguments)` où $v = v\text{-ector}$

`\hvangleorient <macro> (2 Arguments)` où $h = h\text{-at}$ et $v = v\text{-ector}$

— Argument 1: le nom du premier vecteur sans utiliser la macro `\vect`.

— Argument 2: le nom du second vecteur sans utiliser la macro `\vect`.

15.6 Analyse

15.6.1 Constantes classiques

<code>\ggamma <macro> (Sans argument)</code>	<code>\ii <macro> (Sans argument)</code>
<code>\ppi <macro> (Sans argument)</code>	<code>\jj <macro> (Sans argument)</code>
<code>\ttauu <macro> (Sans argument)</code>	<code>\kk <macro> (Sans argument)</code>
<code>\ee <macro> (Sans argument)</code>	

15.6.2 Constantes latines personnelles

`\ct <macro> (1 Argument)`

— Argument: un texte utilisant l’alphabet latin. `\abs <macro> (1 Argument)`

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

— Argument: l’expression à laquelle on applique la fonction valeur absolue.

15.6.3 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 $f = f\text{-rench}$

`\fsh <macro> (Sans argument)` où $f = f\text{-rench}$

`\fth <macro> (Sans argument)` où $f = f\text{-rench}$

15.6.4 Avec un paramètre

`\expb <macro> (1 Argument)`

— Argument: la base de l’exponentielle

`\logb <macro> (1 Argument)`

— Argument: la base du logarithme

15.6.5 Des suites spéciales

`\seqplus <macro> (2 Arguments)`

— Argument 1: l'exposant à droite.

— Argument 2: l'indice à droite.

`\seqhypergeo <macro> (2 Arguments)`

— Argument 1: l'indice à gauche.

— Argument 2: l'indice à droite.

`\seqsuprager <macro> (4 Arguments)`

— Argument 1: l'indice à gauche.

— Argument 2: l'indice à droite.

— Argument 3: l'exposant à droite.

— Argument 4: l'exposant à gauche.

15.6.6 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 .

15.6.7 Dérivation totale

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

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

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

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

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

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

— Option: utilisée, cette option sera l'exposant de dérivation mis entre des parenthèses pour la version non étoilée, et le nombre de primes pour la version étoilée.

— Argument: la fonction à différencier.

`\derfrac <macro> [1 Option] (2 Arguments)`

`\derfrac* <macro> [1 Option] (2 Arguments)`

`\dersub <macro> [1 Option] (2 Arguments)`

— Option: utilisée, cette option sera l'exposant de dérivation.

— Argument 1: la fonction à dériver.

— Argument 2: la variable.

15.6.8 Dérivation partielle

`\partialfrac <macro> [1 Option] (2 Arguments)`

`\partialfrac* <macro> [1 Option] (2 Arguments)`

— **Option**: utilisée, cette option sera l'exposant total de dérivation mis en exposant de ∂ .

— **Argument 1**: la fonction à dériver partiellement.

— **Argument 2**: les variables utilisées pour la dérivation partielle en utilisant la syntaxe suivante : par exemple, `x | y^3 | ...` indique de dériver suivant x une fois, puis suivant y trois fois... etc.

`\partialsub <macro> (2 Arguments)`

`\partialprime <macro> (2 Arguments)`

— **Argument 1**: la fonction à dériver partiellement.

— **Argument 2**: les variables utilisées pour la dérivation partielle en utilisant la syntaxe suivante : par exemple, `x | y^3 | ...` indique de dériver suivant x une fois, puis suivant y trois fois... etc.

15.6.9 L'opérateur crochet – 1^{ère} version

`\hook <macro> (3 Arguments)`

`\hook* <macro> (3 Arguments)`

— **Argument 1**: le contenu entre les crochets.

— **Argument 2**: la borne inférieure affichée en indice.

— **Argument 3**: la borne supérieure affichée en exposant.

15.6.10 L'opérateur crochet – 2^{nde} version

`\vhook <macro> (3 Arguments)` où `v = v-ertical`

`\vhook* <macro> (3 Arguments)` où `v = v-ertical`

— **Argument 1**: le contenu avant le trait vertical.

— **Argument 2**: la borne inférieure affichée en indice.

— **Argument 3**: la borne supérieure affichée en exposant.

15.6.11 L'opérateur d'intégration standard

`\stdint <macro> (Sans argument)`

15.6.12 Les notations \mathcal{O} et \mathcal{o}

`\bigO <macro> (1 Argument)`

`\smallO <macro> (1 Argument)`

— **Argument**: si l'argument est vide, il est ignoré, sinon il est mis entre des parenthèses après \mathcal{O} ou \mathcal{o} .

15.6.13 La notation Ω

`\bigomega <macro> (1 Argument)`

— **Argument**: si l'argument est vide, il est ignoré, sinon il est mis entre des parenthèses après Ω .

15.6.14 La notation Θ

`\bigtheta <macro> (1 Argument)`

— Argument: si l'argument est vide, il est ignoré, sinon il est mis entre des parenthèses après Θ .

15.7 Probabilité

15.7.1 Probabilité « simple »

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

— Option: le nom de la probabilité.

— Argument: l'ensemble dont on veut calculer la probabilité.

15.7.2 Probabilité conditionnelle

`\probacond <macro> [1 Option] (2 Arguments)`

`\probacond* <macro> [1 Option] (2 Arguments)`

`\probacond** <macro> [1 Option] (2 Arguments)`

`\dprobacond** <macro> [1 Option] (2 Arguments)`

— Option: le nom de la probabilité.

— Argument 1: l'ensemble qui donne la condition.

— Argument 2: l'ensemble dont on veut calculer la probabilité.

15.7.3 Espérance

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

— Option: le nom de la fonction espérance.

— Argument: la variable aléatoire dont on veut calculer l'espérance.

15.7.4 Arbres pondérés

`probatree <env>`

`probatree* <env>`

— Contenu: un arbre codé en utilisant la syntaxe supportée par le package `forest`.

— Option "*pweight*": pour écrire un poids sur le milieu d'une branche.

— Option "*apweight*": pour écrire un poids au-dessus le milieu d'une branche.

— Option "*bpweight*": pour écrire un poids en-dessous du milieu d'une branche.

— Option "*frame*": pour encadrer un sous-arbre depuis un noeud vers toutes les feuilles de celui-ci.

15.8 Arithmétique

15.8.1 Opérateurs de base

`\divides <macro> (Sans argument)`

`\notdivides <macro> (Sans argument)`

`\modulo <macro> (Sans argument)`

15.8.2 Fractions continuées standard

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

— Argument: tous les éléments de la fraction continuée séparés par des |.

15.8.3 Fractions continuées généralisées

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

— Argument: tous les éléments de la fraction continuée généralisée séparés par des |.

15.8.4 Comme une fraction continuée isolée

`\singlecontfrac <macro> (2 Arguments)`

— Argument 1: le pseudo numérateur.

— Argument 2: le pseudo dénominateur.

15.8.5 L'opérateur \mathcal{K}

La macro suivante sans argument a un comportement spécifique vis à vis des mises en index et en exposant.

`\contfracope <macro> (Sans argument)`

15.9 Algèbre

15.9.1 Polynômes, séries formelles et compagnie

`\setpoly <macro> (2 Arguments)`
`\setpolyfrac <macro> (2 Arguments)`
`\setserie <macro> (2 Arguments)`
`\setseriefrac <macro> (2 Arguments)`
`\setpolylaurent <macro> (2 Arguments)`
`\setserielaurent <macro> (2 Arguments)`

— Argument 1: l'ensemble auquel les coefficients appartiennent.

— Argument 2: cet argument est une suite de "morceaux" séparés par des barres |, chaque morceau étant une variable formelle.

15.9.2 Calculs expliqués des déterminants 2×2

`\cdettwo <macro> (4 Arguments)` où `c = c-alculate`

— Argument 1: l'entrée à la position (1,1)

— Argument 2: l'entrée à la position (1,2)

— Argument 3: l'entrée à la position (2,1)

— Argument 4: l'entrée à la position (2,2)