Sommaire

Chapitre

3

Mathématiques

Voici les noms des douze apôtres :
en tête Simon que l'on appelle Pierre [...]

L'Évangile selon Saint Matthieu Mt **10** 2.

Un des aspects pratique et rigolo[1] de LaTeX est bien sûr la génération de formules mathématiques ; elles seront naturellement belles, sans que vous n'ayez à faire quoique ce soit.[2] De plus, si vous avez un mauvais souvenir d'un certain éditeur d'équations, réjouissez vous : vous n'avez pas besoin de souris pour écrire des équations ! La génération d'équations avec LaTeX est un domaine particulièrement vaste. Nous présenterons ici les bases requises pour produire les formules « usuelles ». Ce chapitre ne constitue donc qu'une petite introduction à la manipulation des formules avec LaTeX.

Les commandes standard de LaTeX permettent de produire la plupart des équations mathématiques usuelles. Il est cependant conseillé d'utiliser les extensions de l'*American Mathematical Society* nommées amsmath et amssymb simplifiant la mise en forme dans beaucoup de situations.

1. Si, si ! Il y a même des gens qui font des formules juste pour le plaisir !
2. Ou alors juste deux ou trois petites choses...

3.1 Les deux façons d'écrire des maths

LaTeX distingue deux manières d'écrire des mathématiques. L'une consiste à insérer une formule dans le texte, comme ceci : $ax + b = c$, l'autre à écrire une ou plusieurs formules dans un environnement, par exemple :

$$\mathrm{d}U = \delta\mathcal{W} + \delta\mathcal{Q}$$

sachant que chacun de ces deux modes respectent un certain nombre de principes quant à la taille et la position des différents symboles. Voici un exemple avec les deux modes :

```
Déterminer la fonction dérivée
de $f(x)$ :
\begin{displaymath}
  f(x)=\sqrt{\frac{x-1}{x+1}}
\end{displaymath}
si elle existe.
```

Déterminer la fonction dérivée de $f(x)$:

$$f(x) = \sqrt{\frac{x-1}{x+1}}$$

si elle existe.

Cet exemple nous montre donc que l'on entre en mode mathématique « interne » grâce au symbole `$`, et que le même symbole `$` permet d'en sortir. D'autre part, on utilise ici l'environnement `displaymath` qui est le plus simple pour produire des équations. Ce dernier peut être saisi grâce aux commandes `\[` et `\]` (cf. § 3.7.1 page 55).

Nous vous présenterons au § 3.7 les différents environnements de LaTeX.

3.2 Commandes usuelles

3.2.1 Indice et exposant

Comme mentionné au § 1.4.1 page 12, `_` et `^` sont les commandes permettant de produire respectivement *indice* et *exposant*. Il est nécessaire de « grouper » les arguments entre accolades pour que ces commandes agissent sur plusieurs symboles.

`x_2`	x_2	`x_{2y}`	x_{2y}	`x_{t_0}`	x_{t_0}
`x^2`	x^2	`x^{2y}`	x^{2y}	`x_{t^0}`	x_{t^0}
		`x^{2y}_{t_0}`	$x^{2y}_{t_0}$	`x_{t^1}^{2y}`	$x^{2y}_{t^1}$

3.2.2 Fraction et racine

Voici comment produire *racines* et *fractions* :

– la commande `\frac{num}{denom}` produit une fraction formée par le numérateur num et le dénominateur denom ;

– la commande `\sqrt[n]{expr}` affiche la racine n^{e} de son argument arg.

Notons que ces deux commandes ne produisent pas le même affichage selon le mode mathématique : interne ou équation. Ainsi voici une fraction : $\frac{1}{\sin x+1}$ et une racine : $\sqrt{3x^2-1}$ et leur équivalent en mode équation :

$$\frac{1}{\sin x+1} \qquad \sqrt{3x^2-1}$$

Pour en finir avec ces deux commandes, voyons comment elles peuvent être imbriquées et l'effet que cela produit :


```
\begin{displaymath}
  \sqrt{\frac{1+\sqrt[3]{3x+1}}
              {3x+\frac{1-x}{1+x}}}
\end{displaymath}
```


$$\sqrt{\frac{1+\sqrt[3]{3x+1}}{3x+\frac{1-x}{1+x}}}$$

3.2.3 Symboles

Symboles usuels

Le tableau 3.1 page suivante donne les macros produisant une partie des symboles dont vous pourriez avoir besoin.

Nous avons recensé près de 450 symboles disponibles avec les packages latexsym et amssymb. Notre but n'est donc pas de les présenter ici ! Le tableau 3.1 page suivante est une sélection parmi les symboles standard. Nous avons jugé qu'ils faisaient partie des symboles les plus utiles — ce qui, malgré la présence tout à fait fortuite de l'aleph dans ce tableau, démontre que le niveau en mathématiques de l'auteur de ce document avoisine le ras des pâquerettes.

Points de suspension

On utilise couramment pour économiser de l'encre des points de suspension dans des formules. Il en existe de trois types. La commande `\dots` produit des points « posés » sur la ligne :

TAB. 3.1 – Symboles mathématiques usuels

\pm	$\pm$	\otimes	$\otimes$	\cong	$\cong$	\imath	$\imath$
\mp	$\mp$	\oslash	$\oslash$	\subset	$\subset$	\jmath	$\jmath$
\div	$\div$	\odot	$\odot$	\supset	$\supset$	\ell	ℓ
\ast	$\ast$	\leq	$\leq$	\subseteq	$\subseteq$	\aleph	$\aleph$
\times	$\times$	\geq	$\geq$	\supseteq	$\supseteq$	\nabla	∇
\bullets	$\bullet$	\equiv	$\equiv$	\in	$\in$	\\|	$\|$
\circ	$\circ$	\ll	$\ll$	\ni	$\ni$	\partial	∂
\star	$\star$	\gg	$\gg$	\emptyset	$\emptyset$	\wedge	$\wedge$
\setminus	$\setminus$	\sim	$\sim$	\forall	$\forall$	\vee	$\vee$
\oplus	$\oplus$	\simeq	$\simeq$	\infty	∞	\cup	$\cup$
\ominus	$\ominus$	\approx	$\approx$	\exists	$\exists$	\cap	$\cap$


```
$C=\{\vec{c}_0,\vec{c}_1,\dots,\vec{c}_N\}$
est l'ensemble des $N$ couleurs.
```

3.3 $C = \{\vec{c}_0, \vec{c}_1, \dots, \vec{c}_N\}$ est l'ensemble des N couleurs.

La commande `\cdots` produit des points centrés verticalement sur le signe égal :

```
$\vec{\mu}=\frac{1}{N}
(\vec{c}_0+\vec{c}_1+\cdots+\vec{c}_N)$
est la moyenne des $N$ couleurs.
```

3.4 $\vec{\mu} = \frac{1}{N}(\vec{c}_0 + \vec{c}_1 + \cdots + \vec{c}_N)$ est la moyenne des N couleurs.

Enfin les commandes `\vdots` et `\ddots` sont à utiliser essentiellement dans les matrices (cf. § 3.6 et l'exemple 3.15). Ces deux commandes produisent respectivement : $\vdots$ et $\ddots$.

Flèches

Voici un moyen simple pour mémoriser les commandes permettant de générer des flèches :

- toutes les commandes finissent par `arrow` ;
- le préfixe obligatoire `left` ou `right` indique la direction ;
- le préfixe facultatif `long` donne une version longue ;

– la première lettre de la commande mise en majuscule rend la flèche double ;
– on peut mettre des flèches aux deux extrémités en collant les deux mots `left` et `right`.

ainsi :

`\rightarrow`	donne	$\rightarrow$
`\Longleftarrow`	donne	$\Longleftarrow$
`\Leftarrow`	donne	$\Leftarrow$
`\Longleftrightarrow`	donne	$\Longleftrightarrow$

Lettres grecques

Les lettres grecques s'utilisent de la manière la plus simple qui soit : en les appelant par leur nom. Ainsi : `\alpha` donne « α » et `\pi`, « π ». Mettre une majuscule à la première lettre de la commande, donne la majuscule correspondante : `\Gamma` donne « Γ ». Attention, toutes les majuscules ne sont pas disponibles dans l'alphabet grec, on mettra par exemple α en majuscule, avec la lettre A (la commande `\Alpha` n'existe pas).

L'ensemble des réels

Une question « cruciale » que se posent les rédacteurs potentiels de documents scientifiques est : « Comment peut/doit-on écrire le 'R' de l'ensemble des réels ? ». Les avis sont partagés à ce sujet. Historiquement il semble qu'initialement, dans les ouvrages de mathématiques, le symbole des réels était typographié en gras (« Soit $x \in \mathbf{R}$ ») et que les professeurs pour reprendre ces notations sur un tableau avec une craie avaient recours à l'artifice de repasser plusieurs fois sur la lettre « R » ; cette pratique pénible aurait évoluée vers l'écriture « bien connue » : « Soit $x \in \mathbb{R}$ ». Il y a donc les adeptes du $\mathbf{R}$, du $\mathbb{R}$, etc. Pour choisir par soi-même, voir les packages :

– bbm qui propose la commande `\mathbbm{R}` produisant $\mathbb{R}$, la commande `\mathbbmss{R}` produisant $\mathbb{R}$, etc.
– bbold qui propose la commande `\mathbbm{R}` produisant $\mathbb{R}$, etc.
– amssymb qui propose les commandes `\mathbb{R}` produisant $\mathbb{R}$ ainsi que `\mathbf{R}` produisant $\mathbf{R}$

3.3 Fonctions

3.3.1 Fonctions standards

Lorsqu'on veut produire des fonctions mathématiques classiques (logarithmes, trigonométrie,...), il faut utiliser les fonctions de LaTeX prévues à cet effet. Voici un exemple pour vous en convaincre.

```
$\sin^2x + \cos^2 x=1$
```

3.5

$$\sin^2 x + \cos^2 x = 1$$

Et sans les fonctions LaTeX :

```
$sin^2x + cos^2x=1$
```

3.6

$$sin^2x + cos^2x = 1$$

La différence réside dans le fait que LaTeX traite la chaîne `cos` comme une suite de variable (donc produites en italiques) alors que la fonction `\cos` produit « cos » en roman. Une autre différence importante est le placement d'éventuels indices (cf. l'exemple de la fonction `\max` ci-dessous). Parmi les fonctions mathématiques standard de LaTeX, on trouvera :

- toutes les fonctions trigonométriques : `\sin`, `\cos` et `\tan`. En rajoutant `arc` devant, vous aurez les réciproques, et `h` derrière vous obtiendrez les versions hyperboliques.
- les logarithmes népérien et décimal définis respectivement par les fonctions `\ln` et `\log`.
- les fonctions `\sup`, `\inf`, `\max`, `\min`, et `\arg` qui vous permettront de générer des formules de ce genre :

```
\begin{displaymath}
  T=\arg \max_{t<0} f(t)
\end{displaymath}
```

3.7

$$T = \arg\max_{t<0} f(t)$$

Notez l'utilisation de l'opérateur indice `_` et le placement résultant avec la commande `\max`.

3.3.2 Intégrales, sommes et autres limites

LaTeX utilise une syntaxe simple pour produire *intégrales*, *sommes*, etc. La syntaxe est la suivante :

\op_{inf}^{sup}

où op est l'un des opérateurs `sum`, `prod`, `int` ou `lim` et inf et sup sont les bornes inférieure et supérieure de la somme ou de l'intégrale. Ainsi on peut donc écrire :

```
Somme des termes d'une suite
géométrique :
\begin{displaymath}
  \sum_{i=0}^{n}q^i=\frac{1-q^{n+1}}{1-q}
\end{displaymath}
```

3.8

Somme des termes d'une suite géométrique :

$$\sum_{i=0}^{n} q^i = \frac{1-q^{n+1}}{1-q}$$

Le produit $\prod$ s'utilise de manière analogue avec la commande `\prod`. Un exemple avec une intégrale, en veux-tu en voilà :

```
On définit le logarithme
népérien de $x>0$ comme suit :
\begin{displaymath}
  \ln(x)=\int_{1}^{x}\frac{1}{t}\,\mathrm{dt}
\end{displaymath}
```

3.9

On définit le logarithme népérien de $x > 0$ comme suit :

$$\ln(x) = \int_1^x \frac{1}{t}\,\mathrm{dt}$$

La commande `\,` permet d'insérer un léger blanc avant le « dt » (cf. § 3.5.1). Si vous êtes plutôt *curviligne*, vous pouvez utiliser `\oint` qui donne : $\oint$. Bon, je vous donne juste un exemple avec une limite mais c'est bien parce que c'est vous :

```
$f(x)$ admet une limite $\ell$ en $x_0$ :
\begin{displaymath}
  \lim_{x\rightarrow x_0}f(x)=\ell
\end{displaymath}
```

3.10

$f(x)$ admet une limite ℓ en x_0 :

$$\lim_{x \to x_0} f(x) = \ell$$

J'espère que vous avez apprécié le beau ℓ ; pour se fixer les idées sur les deux modes mathématiques, voici les mêmes formules mais incrustées dans le texte. Donc d'abord la sommation : $\sum_{i=0}^{n} q^i = \frac{1-q^{n+1}}{1-q}$, ensuite l'intégrale : $\int_1^x \frac{1}{t} dt = \ln(x)$, et enfin la limite : $\lim_{x \to x_0} f(x) = \ell$.

3.4 Des symboles les uns sur les autres

3.4.1 L'opérateur `not`

L'opérateur `\not` permet de produire la « négation » d'une relation :

```
Soit $x \not\in I$ un réel...
```

3.11 Soit $x \not\in I$ un réel...

le résultat est donc un « slash » sur le symbole suivant. **Attention**, cet opérateur n'est pas très performant : `$\not\longrightarrow$` donne : $\not\longrightarrow$, mais est satisfaisant pour les symboles d'une largeur raisonnable.

3.4.2 Accents

Il est souvent utile,[3] d'accentuer les symboles en guise de notation particulière. Voici les accents disponibles :

`\hat{x}`	$\hat{x}$	`\check{x}`	$\check{x}$	`\breve{x}`	$\breve{x}$
`\acute{x}`	$\acute{x}$	`\grave{x}`	$\grave{x}$	`\tilde{x}`	$\tilde{x}$
`\bar{x}`	$\bar{x}$	`\dot{x}`	$\dot{x}$	`\ddot{x}`	$\ddot{x}$

3.4.3 Vecteurs

Il existe deux[4] façons d'obtenir un vecteur :
- `\vec` pour les petits symboles car `\vec` est une commande d'accentuation ;
- `\overrightarrow` dans les autres cas.

```
Soit $\overrightarrow{A\!B}$ défini
dans la base $(\vec{\imath},\vec{\jmath})$.
```

3.12 Soit $\overrightarrow{A\!B}$ défini dans la base $(\vec{\imath},\vec{\jmath})$.

Notez que `$\vec{A\!B}$` aurait donné : $\vec{A\!B}$ (voir aussi le paragraphe 3.5.1 pour la signification de la commande `\!`). Remarquez également les commandes `\imath` et `\jmath` qui fournissent les lettres « i » et « j » sans point : $\imath$ et $\jmath$.

3. En réalité les mathématiciens dignes de ce nom raffolent de ce genre de petits chapeaux au dessus des symboles ; certains en superposent même deux, voire trois...

4. Le package esvect d'Eddie Saudrais permet de produire des vecteurs avec des flèches mieux dessinées que celles proposées ici.

3.4.4 Commande `stackrel`

La commande `\stackrel` permet de poser deux symboles l'un sur l'autre :

`\stackrel{`symb$_1$`}{`symb$_2$`}`

met le symb$_1$ sur symb$_2$. Par exemple :

```
x\stackrel{f}{\longmapsto}y
```

donne : $x \stackrel{f}{\longmapsto} y$.

3.5 Deux principes importants

Pour bien comprendre la manière dont LaTeX génère les formules, il faut saisir les deux principes suivants :

Espaces : LaTeX ignore les espaces entre les symboles mathématiques ; ainsi : `$x+1$` produira la même formule que `$x + 1$`. C'est LaTeX qui insère les espaces à l'endroit qu'il juge le plus judicieux ;

Texte[5] **:** tout groupe de symboles est considéré comme un groupe de variables ou fonctions ; ainsi `$x=t avec t>0$` produira $x = tavect > 0$ et non ce que vous espériez : $x = t$ avec $t > 0$.

Une fois ces deux principes acquis, voyons comment on peut faire avec.

3.5.1 Espaces en mode mathématique

Tout d'abord, sachez que LaTeX fait un choix d'espacement qui est en général correct. Cependant le jour où vous aurez à jouer l'expert de mouche, les commandes du tableau 3.2 vous permettront d'insérer un ou des espaces dans des formules. Dans ce tableau, on montre l'effet des commandes d'espacement entre deux symboles □.

Pour ce qui concerne les mouches, sachez que l'auteur de ce manuel a sournoisement inséré un certain nombre d'espacements au numérateur du calcul de

5. L'insertion de texte dans une formule ne devient un problème que dans un environnement de la famille `displaymath`, puisque vous pouvez toujours écrire « `$x=t$ avec $t>0$` », bien sûr !

TAB. 3.2 – Espacement en mode mathématique

`\!` □□	*(rien)* □□	`\,` □□	`\:` □□
`\;` □□	`\␣` □□	`\quad` □□	`\qquad` □□

la somme des termes de la suite géométrique (§ 3.3.2 page 49), pour aligner les deux q de la fraction. Voici ce que donnait la formule par défaut :

$$\sum_{i=0}^{n} q^i = \frac{1-q^{n+1}}{1-q}$$

ε

et voyons si les histoires de q vous donnent le sens de l'observation.

3.5.2 Texte en mode mathématique

Le moyen le plus simple d'insérer du texte dans une formule est de de le mettre « en boîte » et d'insérer quelques espaces :

```
Soient les suites $(u_n)$ et $(v_n)$ :
\begin{displaymath}
  u_n=\ln n\quad
  \mbox{et}\quad v_n=(1+\frac{1}{n})^n
  \label{ex-maths-suite}
\end{displaymath}
```

3.13

Soient les suites (u_n) et (v_n) :

$$u_n = \ln n \quad \text{et} \quad v_n = (1+\frac{1}{n})^n$$

Vous trouverez des détails sur la commande `\mbox` à la section 4.4.1 page 74. Si vous avez pensé à mettre en route le package amsmath vous serez en mesure d'utiliser la commande `\text` en lieu et place de `\mbox`.

3.6 Array : simple et efficace

L'environnement `array` est un environnement qui vous permettra de produire la grande majorité de vos formules. Comme son nom l'indique il range des objets en ligne et colonne. En fait c'est le pendant de l'environnement `tabular` du mode texte. Et comme `tabular`, `array` ne passe pas à la ligne.

3.6.1 Comment ça marche

La syntaxe rappelle celle de `tabular` :

`\begin{array}[vpos]{format} ... \end{array}`

où format précise pour chaque colonne l'alignement : `c` pour centré, `l` pour aligné à gauche et `r` pour aligné à droite ; l'argument optionnel vpos spécifie quant à lui le positionnement vertical du tableau. Comme dans les tableaux, on notera l'utilisation des commandes :

- `&` comme séparateur de colonne ;
- `\\` pour passer à la ligne.

```
Soit $A=\begin{array}{rc}
  -1 & 1 \\
   3 & 4
\end{array}$ la matrice ...
```

3.14

Soit $A = \begin{array}{rc} -1 & 1 \\ 3 & 4 \end{array}$ la matrice ...

Voici un exemple utilisant les points de suspensions :

```
\begin{displaymath} A=\left[\begin{array}{ccc}
    a_{00} & \dots & a_{0n}\\
    \vdots & \ddots & \vdots\\
    a_{n0} & \dots & a_{nn}
  \end{array}\right]\end{displaymath}
```

3.15

$$A = \left[\begin{array}{ccc} a_{00} & \dots & a_{0n} \\ \vdots & \ddots & \vdots \\ a_{n0} & \dots & a_{nn} \end{array}\right]$$

3.6.2 Array et les délimiteurs

On utilise couramment l'environnement `array` pour produire des matrices. Il faut alors avoir recours à des *délimiteurs*. Ces délimiteurs sont de la famille des parenthèses et permettent d'englober un objet mathématique entre crochets, accolades, etc. La syntaxe est la suivante :

`\left`delim$_1$ mobjet `\right`delim$_2$

où delim$_1$ et delim$_2$ sont deux délimiteurs et mobjet un objet mathématique.

Parmi les délimiteurs, voici les plus usités :

`(` et `)`	(Π)	`[` et `]`	$[\Pi]$
`\{` et `\}`	$\{\Pi\}$	`\lfloor` et `\rfloor`	$\lfloor\Pi\rfloor$
`\lceil` et `\rceil`	$\lceil\Pi\rceil$	`\langle` et `\rangle`	$\langle\Pi\rangle$
`|`	$\|\Pi\|$	`\\|`	$\|\|\Pi\|\|$

L'intérêt des délimiteurs est qu'ils s'adaptent automatiquement à la taille des objets qu'ils entourent :

```
soit  $I=
\left[\begin{array}{cc}
       1&0 \\ 0&1
      \end{array}\right]$ la matrice identité.
```

3.16 soit $I = \left[\begin{array}{cc} 1&0 \\ 0&1 \end{array}\right]$ la matrice identité.

On peut également reprendre l'exemple 3.13 page 52 avec des délimiteurs pour ajuster la taille des parenthèses :

```
Soient les suites $(u_n)$ et $(v_n)$ :
\begin{displaymath}
  u_n=\ln n\quad\mbox{et}
  \quad v_n=\left(1+\frac{1}{n}\right)^n
\end{displaymath}
```

3.17 Soient les suites (u_n) et (v_n) :

$$u_n = \ln n \quad \text{et} \quad v_n = \left(1+\frac{1}{n}\right)^n$$

Il doit toujours y avoir une commande `\right` pour une commande `\left`. Cependant, il n'est pas nécessaire d'avoir les mêmes symboles à droite et à gauche.

Voici un exemple où on utilise la commande `\right.` pour spécifier que l'on n'utilise pas de symbole à droite :

```
soit $ S_i=\left\{\begin{array}{rl}
    -1  & \mbox{si $i$ est pair}    \\
     1  & \mbox{sinon.}\end{array}\right.$
```

3.18 soit $S_i = \left\{\begin{array}{rl} -1 & \text{si } i \text{ est pair} \\ 1 & \text{sinon.} \end{array}\right.$

3.6.3 Pour vous simplifier la vie...

Le package amsmath permet de saisir plus simplement les matrices avec notamment deux environnements : `pmatrix` (p pour parenthèse) et `bmatrix` (b pour *bracket*).

```
\begin{displaymath}
  \bar{\bar{\sigma}}=\begin{bmatrix}
    \sigma_{11} & \sigma_{12} \\
    \sigma_{21} & \sigma_{22} \\
  \end{bmatrix}
\end{displaymath}
```

3.19

$$\bar{\bar{\sigma}} = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix}$$

3.7 Équations et environnements

Nous présenterons dans ce paragraphe trois environnements standard de LaTeX permettant de produire des formules.

3.7.1 L'environnement `displaymath`

Vous l'avez compris, si vous avez lu jusqu'ici, `displaymath` affiche une formule centrée, interrompant le paragraphe. Un raccourci agréable de :

```
\begin{displaymath}...\end{displaymath}
```

est : `\[...\]`. Ainsi :

```
Distance colorimétrique :\[
\Delta E=\sqrt{
  \Delta L^{*2}+ \Delta a^{*2}+\Delta b^{*2}}
\]
```

3.20

Distance colorimétrique :

$$\Delta E = \sqrt{\Delta L^{*2} + \Delta a^{*2} + \Delta b^{*2}}$$

3.7.2 L'environnement `equation`

L'environnement `equation` est l'équivalent du précédent, sauf qu'il numérote la formule.

```
À retenir : si $a>0$ et $b>0$,\begin{equation}
  \ln(ab)=\ln(a)+\ln(b)
\end{equation}
```

3.21

À retenir : si $a > 0$ et $b > 0$,

$$\ln(ab) = \ln(a) + \ln(b) \tag{3.1}$$

L'option de classe de document `leqno` met le numéro des équations à gauche. Et l'option `fleqn` aligne les équations à gauche, au lieu de les centrer.

3.7.3 Formules multi-lignes

Dans une précédente édition, nous finissions la présentation des environnements standard par l'environnement `eqnarray` qui permet de produire des formules de plusieurs lignes. **Sachez que c'est mal**. Il existe d'ailleurs des écrits à ce sujet (lire par exemple [5] ou [6]), vous expliquant comment produire des documents « propres ». Prenez bien conscience qu'utiliser `eqnarray` (et bien d'autres choses) **est un péché**, et que si vous cédez malgré tout à la tentation, l'inquisition vous retrouvera un jour ou l'autre par l'intermédiaire d'un moteur de recherche. Aucune confession ou indulgence ne pourra vous sortir de ce mauvais pas, vous êtes prévenus.

Nous vous présentons donc ici l'environnement `align` du package amsmath :

- \\ passe à la ligne ;
- chaque ligne est numérotée sauf si la commande `\nonumber` est présente dans la ligne ;
- on procède à l'alignement avec deux [6] opérateurs `&`.

```
\begin{align}
  (a+b)^2 &=  (a+b)(a+b)\nonumber\\
          &=  a^2+b^2+2ab
\end{align}
```

3.22

$$\begin{aligned}(a+b)^2 &= (a+b)(a+b)\\ &= a^2+b^2+2ab\end{aligned} \tag{3.2}$$

Il existe une forme « étoilée » de l'environnement : `align*` où aucune des lignes n'est numérotée. Si vous voulez faire référence à certaines lignes d'un `align`, il vous faudra poser autant de `\label` nécessaires sur chaque ligne correspondante.

Pour faire numéroter une équation s'étalant sur plusieurs lignes on peut utiliser l'environnement `split` (lui aussi fourni avec amsmath) :

```
\begin{equation}
  \begin{split}
    (a+b)^2 &=  (a+b)(a+b)\\
    &=  a^2+b^2+2ab
  \end{split}
\end{equation}
```

3.23

$$\begin{aligned}(a+b)^2 &= (a+b)(a+b)\\ &= a^2+b^2+2ab\end{aligned} \tag{3.3}$$

6. Puisqu'il y a trois colonnes.

3.8 Changer le style en mode mathématique

3.8.1 Fontes

LaTeX fournit plusieurs commandes permettant de changer de fontes dans les modes mathématiques. Par défaut tout symbole ou suite de caractères (autre qu'une fonction) est produit en italique dans le document final. Or dans certains cas, il est utile de pouvoir forcer le style de fonte. Voici comment réaliser un tel exploit :

`Soit $\mathit{A\in\Phi}$`	Soit $\mathit{A\in\Phi}$
`Soit $\mathrm{A\in\Phi}$`	Soit $\mathrm{A\in\Phi}$
`Soit $\mathbf{A\in\Phi}$`	Soit $\mathbf{A\in\Phi}$
`Soit $\mathsf{A\in\Phi}$`	Soit $\mathsf{A\in\Phi}$
`Soit $\mathtt{A\in\Phi}$`	Soit $\mathtt{A\in\Phi}$
`Soit $\mathcal{A}\in\Phi$`	Soit $\mathcal{A}\in\Phi$

La commande `\mathcal` doit prendre exclusivement des lettres majuscules latines comme argument. Dans le cas contraire, les résultats seront farfelus. Par exemple, la séquence :

```
\mathcal{abcd\Gamma}
```

donne $\mathcal{abcd\Gamma}$.

3.8.2 Taille des symboles

LaTeX distingue quatre *styles* d'écriture des formules. Ces modes sont utilisés suivant la « situation » dans laquelle se trouve LaTeX lorsqu'il produit une partie d'une formule :

texte pour une formule insérée dans le texte courant ;

equation pour une formule sous forme d'*équation* ;

indice pour l'écriture des indices ;

sous-indice pour les indices d'indices

chacun de ces modes peut être enclenché explicitement par l'utilisateur grâce aux déclarations suivantes :

- `\textstyle` pour le mode texte ;
- `\displaystyle` pour le mode équation ;
- `\scriptstyle` pour le mode indice ;

– `\scriptscriptstyle` pour le mode indice d'indice

Voici deux exemples illustrant comment forcer le mode *texte* en mode *équation* et inversement :

```
deux produits : $\prod_{1}^{n}f_i$
et $\displaystyle\prod_{1}^{n}f_i$

et inversement :
\[ \prod_{1}^{n}f_i
\mbox{ et }\textstyle\prod_{1}^{n}f_i \]
```

3.24

deux produits : $\prod_1^n f_i$ et $\displaystyle\prod_1^n f_i$

et inversement :

$$\prod_1^n f_i \text{ et } \textstyle\prod_1^n f_i$$

3.8.3 Créer de nouveaux opérateurs

Imaginez que vous ayez besoin de créer un opérateur spécial nommé « burps ». Il suffira de procédé comme suit :

```
\newcommand{\burps}{%
  \mathop{\textrm{burps}}}
$x=\burps_i f(i)$
```

3.25

$x = \operatorname{burps}_i f(i)$

Un autre exemple, pour franciser la fonction « arcsinus » (produisant par défaut arcsin), on pourra écrire :

```
$\theta = \arcsin x$
\renewcommand{\arcsin}{%
  \mathop{\textrm{Arcsin}}\nolimits}
$\theta = \arcsin x$
```

3.26

$\theta = \arcsin x \quad \theta = \operatorname{Arcsin} x$

La commande `\nolimits` indique que l'opérateur concerné ne fera pas usage d'arguments en indice ou exposant comme le font les opérateurs `\lim`, `\int`, etc. En outre les deux exemples précédents utilisent les commandes `\newcommand` et `\renewcommand` dont il est question au paragraphe 4.5 page 82.

Enfin une autre voie possible si vous avez pris soin de charger le package amsmath est de déclarer dans le préambule :

```
\DeclareMathOperator*{\vlunch}{vlunch}
\DeclareMathOperator{\zirgl}{Zirgl}
```

```
\[x=\vlunch_i f(\theta)\]
où $\theta = \zirgl y$
```

3.27

$$x = \operatorname*{vlunch}_i f(\theta)$$

où $\theta = \operatorname{Zirgl} y$

Conclusion

Ce chapitre présente les fonctions de base pour produire des formules. Ces commandes suffisent pour la plupart des documents scientifiques. Si vous êtes amenés à rédiger des documents truffés de formules complexes, il est possible que les seules macros de LaTeX ne suffisent plus. C'est pourquoi la célèbre *American Mathematical Society* a conçu pour vous un package nommé $\mathcal{AMS}$TeX (mise en route : `\usepackage{amsmath}`) capable de générer des formules particulièrement « tordues. »