LaTeX 2ε Guide d'introduction à l'édition de textes en Français

Version 1.4

28 mai 2003

Table des matières

Liste des tableaux

1

Règles fondamentales

1.1 Caractères particuliers

1.1.1 Caractères réservés

En standard, LaTeX ne comprend qu'un nombre limité de caractères. Les seuls caractères autorisés dans le fichier d'entrée sont les suivants : les lettres minuscules et majuscules, les chiffres et les 16 caractères de ponctuation suivants

```
. ; , ? : ! ’ ‘ ( ) [ ] - / * @
```

Les 10 caractères suivants, que nous appellerons *caractères réservés*, sont utilisés pour les commandes LaTeX.

```
# $ & \ % _ ^ ~ { }
```

Il est cependant possible de les faire apparaître dans le texte, à condition de les introduire par le code approprié : dans le tableau ci-dessous, les caractères de la ligne du dessus sont produits par le code de la ligne du dessous :

#	$	\	&	%	{	}
`\#`	`\$`	`$\backslash$`	`\&`	`\%`	`\{`	`\}`

Les 5 caractères

```
+ = < > |
```

sont utilisés uniquement en mode mathématique bien que les deux premiers puissent être utilisés en mode normal.

Le caractère " n'est jamais utilisé. Il est toutefois possible de mettre du texte entre “guillemets” (voilà ce que ça donne ! Le type de guillemets peut varier selon que l'on souhaite une mise en page de type française ou anglosaxonne) par l'intermédiaire des symboles `‘‘` et `’’`.

Remarque *L'environnement verbatim permet d'imprimer un texte tel qu'il est introduit et peut contenir n'importe quel caractère. Dans cet environnement, aucune commande ne pourra être utilisée (voir section 3.7.2).*

1.1.2 Les caractères accentués

À la base, LaTeX ne reconnaît pas les caractères accentués. Il faut alors les construire à l'aide de séquences de caractères reprises dans le tableau 1.1 :

Tab. 1.1 – Accents et lettres particulières

ò	`\‘o`	ó	`\’o`	ô	`\^o`	ö	`\’’o`
õ	`\~o`	ō	`\=o`	ȯ	`\.o`	ŏ	`\u o`
ǒ	`\v o`	ő	`\H o`	o͡o	`\t oo`	ǫ	`\c o`
ọ	`\d o`	o̲	`\b o`				
œ	`\oe`	Œ	`\OE`	æ	`\ae`	Æ	`\AE`
å	`\aa`	Å	`\AA`	ø	`\o`	Ø	`\O`
ł	`\l`	Ł	`\L`	¡	`!‘`	¿	`?‘`

Les caractères présents sur le clavier (é,è,ê,à,ç,...) peuvent toutefois être reconnus et interprétés si le préambule du document contient l'appel du module approprié (voir section 2.1) par la commande `\usepackage[ansinew]{inputenc}` (ou `\usepackage[latin1]{inputenc}`.)

1.1.3 Les tirets et les points

LaTeX connaît trois types de tirets, obtenus en juxtaposant un nombre variable de tirets simple ; un quatrième type est apparenté, mais n'est pas tout à fait un tiret : il s'agit du signe mathématique négatif. Les exemple suivants illustrent leur emploi :

beau-fils	`beau-fils`
pages 13–25	`pages 13--25`
Il peut venir — ou pas.	`Il peut venir --- ou pas.`
1 et −1	`$1$ et $-1$`

Par ailleurs, les points de suspension ne peuvent pas être obtenus en plaçant trois points successifs car les espaces entre les points sont plus importants dans ce cas. Pour obtenir "...", on utilise la commande `\ldots` (qui peut s'utiliser en mode texte comme en mode mathématique).

1.2 L'espacement

LaTeX suit un ensemble de règles concernant les espaces, qu'il est bon de connaître dès le départ. Afin d'éviter les doubles frappes malencontreuses, le compilateur n'insère jamais qu'une seule espace, quel que soit le nombre de blancs laissés dans une même ligne de code. De même, le retour de chariot ("enter") est considéré comme un seul espace. Pour définir un nouveau paragraphe, il faut passer une ligne dans le code.

Ces deux règles sont illustrées par l'exemple suivant, pour lequel le texte de gauche est produit par le code de la colonne de droite.

Laisser un ou plusieurs espaces entre deux mots n'a pas d'importance.
Une ligne vide définit un nouveau paragraphe.

```
Laisser un ou     plusieurs
espaces entre deux mots n'a
pas d'importance.

Une ligne vide définit un
nouveau paragraphe.
```

L'utilisation des tabulations est à éviter car celles-ci sont soumises à des règles de typographies propres nécessitant des commandes précises.

L'espacement entre les lignes est fixé par défaut ; il est cependant possible de le changer en redéfinissant la dimension

`\baselineskip=`*longueur*.

Si cette instruction est donnée dans le préambule elle concernera l'ensemble du texte ; si elle apparaît à un endroit particulier du texte, une instruction du type `\baselineskip=16pt` imposera au texte qui suit, et ce jusqu'à une nouvelle modification explicite (donc jusqu'à la fin si le rédacteur ne spécifie plus aucune valeur d'espacement), un espacement de seize points entre le haut d'une ligne et le haut de la ligne qui suit. La commande

`\linespread{`*facteur*`}`

insérée dans le préambule permet également de modifier l'interligne : la valeur par défaut de *facteur* est 1 ; pour des raisons techniques trop longues à exposer ici, un interligne "un et demi" est produit par le facteur 1.3 et un double intervalle par 1.6. Cette commande agit cependant également sur la mise en forme des tableaux et des notes de bas de pages.

Un espace horizontal simple est obtenu en laissant un — ou plusieurs — blanc. Il est toutefois possible d'ajouter des espacements horizontaux supplémentaires. La commande la plus simple pour ajouter un espace est `\␣` (un backslash suivi d'un espace). Celle-ci ajoute un espace de la valeur d'un caractère ; la commande `\,` est utilisable en mode texte et également en mode mathématique, et produit un espace de la valeur d'un demi caractère. L'insertion d'un tilde ~ entre deux mots produit un espace de la valeur d'un caractère,

mais que l'on dit insécable : les deux éléments de part et d'autre du tilde resteront toujours attachés (ce qui permet notamment d'éviter un retour de ligne au milieu de l'écriture d'un numéro de téléphone). Les commandes

`\hspace{`*esp*`}` et `\hspace*{`*esp*`}`

produisent un espace horizontal de *esp* (une longueur positive ou négative) ; dans le premier cas, cet espace n'est pas pris en compte s'il survient à une coupure de ligne, dans le second il est imposé quel que soit l'endroit où il apparaît. La commande

`\stretch{`*n*`}`

utilisée avec `\hspace` produit un espace horizontal élastique qui s'étend de manière à remplir totalement la ligne ; si deux de ces commandes sont utilisées sur une même ligne, les deux espaces qu'elles produisent s'étendent dans le rapport des facteurs n qui les caractérisent :

x x x

```
x\hspace{\stretch{1}}
x\hspace{\stretch{3}}x
```

Un espace vertical de longueur prédéterminée est inséré au moyen des commandes

`\smallskip`, `\medskip` et `\bigskip`

produisant respectivement un petit, moyen ou grand espacement ; la taille de ces espacements dépend en général du style de document, de la taille de la police et de la valeur imposée à l'interligne. Un espacement vertical de hauteur libre peut être inséré au moyen d'une des commandes

`\vspace{`*esp*`}` et `\vspace*{`*esp*`}`,

qui produisent un espace vertical de *esp* (longueur positive ou négative). Si cette commande apparaît dans un paragraphe, l'espace est passé en dessous de la ligne dans laquelle elle apparaît. L'espace disparaît dans le premier cas s'il se produit à une coupure de colonne.

Des espacements de longueur non déterminée peuvent également être utilisés afin de compléter une ligne ou une page. La commande

`\fill`

complète une page par des espacements avant de renvoyer à la page suivante.

`\hfill`

est un raccourci pour `\hspace{\fill}`. Son effet est de coller le texte qui suit le plus à droite possible, donc contre la marge de droite dans un paragraphe habituel. Si deux de ces commandes encadrent un texte sur une même ligne, celui-ci est centré. La commande

`\vfill`

est un raccourci pour `\vspace{\fill}` et agit de manière similaire. La commande

`\dotfill` (ou `\hrulefill`)

agit comme `\hfill`, au fait près qu'elle remplit le vide par des pointillés (ou une ligne pleine).

L'exemple le plus courant d'utilisation de ce genre de commandes est la création d'une page de garde : on place un premier texte qui doit apparaître au sommet de la page, ensuite `\vfill` et le titre qui devra être centré en hauteur, puis encore `\vfill` et le pied de page. La même disposition peut aussi être obtenue par l'insertion du premier texte, une commande `\vspace{\stretch{m}}`, le texte du milieu de la page, une commande `\vspace{\stretch{n}}` et le texte du bas de la page, puis finalement la commande `\pagebreak` ; le texte central sera disposé au milieu de la page si $m = n = 1$, et respectera le rapport de ces deux chiffres s'ils sont différents.

Des espaces insécables doivent parfois être utilisés, lorsque deux éléments doivent être séparés par un espace, mais doivent impérativement reste sur une même ligne ; c'est le cas d'une proposition du type "comme décrit à la page 5", où le chiffre 5 doit être séparé du mot "page", mais sur la même ligne. Pour imposer ce groupement, il suffit de remplacer l'espace par le caractère tilde (`~`) et écrire par exemple `page~5`. Une alternative consiste à grouper les termes à l'aide de la commande `\mbox{`*termes*`}` (voir 4.4).

Une attention particulière doit être portée aux espacements qui suivent une commande. En effet, LaTeX ignore un espace qui suit immédiatement la commande lors de son insertion dans le texte, puisqu'il considère que celui-ci indique la fin de la définition de la commande elle-même. Par exemple, le texte "`\LaTeX est génial.`" produira LaTeXest génial. On remarque qu'il manque un espace après LaTeX. Comme le compilateur ne compte pas le nombre d'espaces, il est inutile d'en mettre plusieurs, car le texte produit sera identique. Pour forcer un espace après une commande, il suffit d'insérer la commande `\␣` (backslash suivi directement d'un espace) juste dernière le nom de la commande. Pour produire le texte désiré, on introduit l'instruction " `\LaTeX\ est génial.`".

LaTeX ajuste les espaces d'une ligne en fonction du texte et de la largeur spécifiée. Cependant, l'espace entre un point terminant une phrase et le début de la phrase suivante est plus long que l'espace séparant deux mots. Il arrive cependant qu'un point induise le compilateur en erreur, en particulier lors de l'emploi d'abréviations telle "etc." Pour indiquer à LaTeX qu'un point ne termine pas une phrase, il suffit de placer `\␣` directement après le point (sans espace entre le point et le \). Dans le cas inverse, on indique à LaTeX qu'un point qui suit une majuscule termine une phrase par un `\@` juste avant le point. Si un point précède directement une parenthèse fermante, LaTeX ajoute également un espace supplémentaire après la parenthèse (si elle est suivie d'un espace.) Comme pour le point, on indique par un \ suivi d'un espace directement après la parenthèse que l'espace doit être normal.

La même règle doit également être utilisée pour le point d'interrogation (`?`), pour le point d'exclamation (`!`) et pour les deux points (`:`).

1.3 Les longueurs

Une *longueur* est une mesure de distance. Elle doit s'exprimer en fonction d'une unité comme le centimètre (`cm`), le millimètre (`mm`), le pouce (`in`) ou le point (`pt`) qui est une unité traditionnelle en typographie. Ces unités de mesure sont dites fixes car elles ne dépendent pas du contexte dans lequel on les utilise. Par opposition aux unités fixes, LaTeX connaît également des unités rigides dont la valeur dépend du style du document, de la police utilisée et d'autres paramètres propres au document. On utilise souvent `em` pour la largeur de la lettre "M"et `ex` pour la hauteur de la lettre "x". Il est aisé de comprendre que ces longueurs dépendent de l'endroit où on les utilise et de la taille de la police courante. Le plus souvent, on utilisera la première pour des espaces horizontaux et la deuxième pour des espaces verticaux. Il est aussi possible d'utiliser des commandes de longueur ou des multiples de celles-ci, c'est-à-dire des commandes dont la valeur est une longueur. Par exemple, `\parindent` représente la longeur du retrait de paragraphe. Si on écrit `2.5\parindent`, cela représente 2.5 fois la longueur précisée par `\parindent`.

Pour (re)définir une commande de longueur, supposons vouloir attribuer une valeur de 5 millimètres à `\long`, on écrit `\long=5mm`.

2

Préambule

Cette section décrit la structure fondamentale d'un document LaTeX avant le début du texte lui-même, et en particulier l'usage des notions de classes et modules (angl. *packages*).

2.1 Classes et modules

Un document LaTeX doit impérativement commencer par une commande précisant la *classe* du document, c'est à dire le style de mise en page que l'on souhaite donner au texte. Ces classes admettent un ensemble de fonctions intégrées qui sont notées en options. Certaines fonctions ne sont pas intégrées à la définition d'une classe, mais font l'objet de modules (*packages*) séparés, pouvant eux aussi admettre certaines options[1]. Le choix de la classe et des modules additionnels s'effectue au moyen des commandes suivantes :

`\documentclass[`*options*`]{`*class*`}`
`\usepackage[`*options*`]{`*packages*`}`

La première de ces deux commandes constitue impérativement la première ligne active du document source. Les classes et les modules les plus courants sont présentés à la section suivante. Par exemple, pour spécifier la création d'un article imprimé en recto-verso, et contenant des figures au format PostScript encapsulé, on écrira :

[1] La différence entre les classes et les modules additionnels se note au niveau des fichiers sources : les classes sont définies par des fichiers de type `.cls`, alors qu'usuellement les modules admettent l'extension `.sty`.

```
\documentclass[twoside]{article}
\usepackage{epsfig}
```

Il est possible de charger plusieurs modules à l'aide d'une seule commande `\usepackage`, en séparant alors les différents modules appelés par une virgule ; par exemple :

```
\usepackage{epsfig,multicol}
```

Il est possible également que certains modules partagent les mêmes options. Ainsi on peut réaliser un article recto-verso contenant des illustrations (prises en charge par le module `graphics`) en faisant appel au gestionnaire de conversion `dvips`, ainsi qu'une prise en charge des couleurs (par le module `color`) qui appelle le même gestionnaire de conversion ; on écrit cela

```
\documentclass[twoside]{article}
\usepackage[dvips]{graphics}
\usepackage[dvips]{color}
```

ce qui peut être réduit immédiatement à :

```
\documentclass[twoside]{article}
\usepackage[dvips]{graphics,color}
```

Enfin, lorsque cela prend un sens pour eux, les modules peuvent également utiliser les options définies à la classe du document, ce qui permet d'écrire également pour l'exemple précédent :

```
\documentclass[twoside,dvips]{article}
\usepackage{graphics,color}
```

2.1.1 Classes standard

Les classes qui suivent sont les plus couramment distribuées avec LaTeX :

Tab. 2.1 – Classes de documents

article Rédaction d'un article, pour lequel la plus haute subdivision est la section.

report Rédaction d'un rapport, subdivisé en chapitres, eux-mêmes divisés en sections.

book Rédaction d'un livre ou d'un ouvrage divisé en parties, elles-mêmes divisées en chapitres, eux-mêmes en sections.

letter Mise en page d'une lettre.

slides Réalisation de transparents.

proc Classe basée sur la classe `article`, qui permet de rédiger un compte-rendu de conférence (proceedings).

2.1.2 Options de classes standard

Les options suivantes sont celles qui sont le plus souvent rencontrées pour raffiner la définition d'une classe :

Tab. 2.2 – Options essentielles pour les classes de documents

10pt définit la taille de la police de caractère principale : les valeurs possibles sont `10pt, 11pt, 12pt` et la valeur par défaut est `10pt`.

a4paper définit la taille de la feuille de papier : les valeurs possibles sont `a4paper, a5apaper, b5paper, letterpaper, executivepaper, legalpaper` et la valeur par défaut est le format standard américain `letterpaper`.

fleqn impose un alignement à gauche des équations, plutôt que de les centrer par défaut.

leqno place la numérotation des formules mathématiques à gauche plutôt qu'à droite par défaut.

titlepage, notitlepage indique si une nouvelle page doit être commencée après le titre général du document ou non. Par défaut, les classes `book` et `report` créent une page de titre, alors que la classe `article` permet au texte de continuer sur la même page que le titre principal.

twocolumn impose de formater le texte sur deux colonnes.

twoside, oneside impose respectivement une sortie en recto-verso ou en recto simple. Par défaut, les classes `article` et `report` sont définies en recto simple et la classe `book` en recto-verso.

openright, openany impose respectivement de commencer un chapitre sur une page de droite ou simplement à la page suivante. Cette option n'a pas de sens pour la classe `article` (qui ne connaît pas la notion de chapitre) ; par défaut, la classe `report` commence un chapitre sur la page vierge qui suit, alors que la classe `book` passera au besoin une page pour commencer un chapitre sur une page de droite.

2.1.3 Modules standard

Les modules présentés dans le tableau 2.3 sont usuellement fournis avec LaTeX ; les deux plus courants d'entre eux sont `latexsym` et `inputenc`. Les fichiers `.dtx` renseignés fournissent des informations additionnelles, parfois assez techniques, à propos de ces modules ; ils sont soit distribués avec ceux-ci, soit disponibles par Internet. Les modules additionnels de la table 2.4 sont généralement fournis avec les modules standards pour une configuration complète de LaTeX ; on retiendra essentiellement les modules `amslatex` et `graphics` qui gèrent respectivement l'insertion de symboles mathématiques évolués et le traitement des images et dessins.

TAB. 2.3 – Modules principaux fournis avec LaTeX

alltt Ce module fournit l'environnement `alltt`, qui agit comme l'environnement `verbatim` à l'exception des commandes `\`, `{`, et `}` qui gardent leur sens particulier de commande réservée à LaTeX. Il est décrit par le fichier `alltt.dtx`.

doc Ce module de base, décrit dans le fichier `doc.dtx`, permet l'insertion de documentation dans un programme LaTeX.

exscale Ce module fournit des versions mesurées de polices mathématiques ; il est décrit par le fichier `exscale.dtx`.

fontenc Ceci spécifie à LaTeX l'encodage de la fonte ; la description est donnée par le fichier `ltoutenc.dtx`.

graphpap Ce module définit la commande `\graphpaper` utilisée dans l'environnement `picture`.

ifthen Ceci fournit des commandes conditionnelles de type 'if...then do...otherwise do...'. Ces commandes sont décrites dans le fichier `ifthen.dtx`.

inputenc Ce module décrit l'encodage d'entrée des caractères tel que doit le considérer LaTeX ; en particulier, l'utilisation de ce module avec l'option `[ansinew]` permet à LaTeXde reconnaître les caractères accentués écrits à partir d'un clavier conventionnel 'azerty'. Le module est décrit dans `inputenc.dtx`.

latexsym Ce module contient la définition de multiples symboles qui ne sont pas contenus dans la fonte de base de LaTeX ; la description complète se trouve dans le fichier `latexsym.dtx`.

makeidx Ce module fournit les commandes nécessaires à la production d'un index.

newlfont et oldlfont Deux modules utilisés pour adapter les commandes des fontes de LaTeX 2.09 à LaTeX2e.

showidx Ce module entraîne l'impression de l'argument associé à chaque commande `\index` sur la page où il apparaît.

syntonly Ce module entraîne la réalisation d'un document sans mise en page ; il est décrit par `syntonly.dtx`.

tracefnt Ceci permet un contrôle de l'information associée aux fontes chargées par LaTeX.

Tab. 2.4 – Modules additionnels

amslatex Module fourni par l'AMS (American Mathematical Society) pour la production de formules mathématiques complexes ; ce module contient entre autres le module `amsmath`.

amsmath Ce module constitue une partie seulement de `amslatex` qui contient des éléments avancés pour l'obtention de formules mathématiques.

babel Ce module gère les mises en pages associées à de multiples langues ; le choix de la langue est donné en option de l'appel du module (ex. `\usepackage[french]{babel}` ou le même avec `[frenchb]`, qui sont d'ailleurs équivalents.)

cyrillic Ceci contient des éléments propres à la production de documents en cyrillique.

graphics Ceci contient le module `graphics` qui fournit l'information nécessaire à l'inclusion et la transformation de graphiques et d'images, en ce compris des fichiers produits par d'autres logiciels. Le module contient également l'ensemble `color` qui gère l'utilisation de couleurs.

psnfss Contient l'essentiel pour la rédaction à l'aide des fontes PostScript de type 1.

tools Collection de modules renfermant certaines mises à jours d'environnements.

2.2 Les en-têtes et pieds de pages

`\pagestyle{`*style*`}`

permet de définir un style pour les en-têtes et les pieds de pages ; les styles les plus courants sont

`plain` pour imprimer simplement le numéro de la page au milieu du pied de page ; c'est le style par défaut.

`empty` pour ne rien inclure dans les deux zones.

`headings` pour afficher le nom de la section et le numéro de page en haut de la page.

`myheadings` pour appliquer un en-tête défini par l'auteur.

D'autres commandes permettent de raffiner la mise en page ; ainsi

`\thispagestyle{`*style*`}`

permet de modifier uniquement le style de la page courante.

`\pagenumbering{`*choix*`}`

permet de définir la manière de numéroter les pages. Cette numérotation est un élément généralement prédéfini et associé à un style de page. Il existe toutefois des commandes spécifiques à l'aspect de la numérotation. La commande `\pagenumbering{`*style*`}` permet de préciser le format des nombres. Les *styles* existants sont :

`arabic` : nombres arabes,

`roman` : nombres romains minuscules,

`Roman` : nombres romains majuscules,

`alph` : lettres minuscules,

`Alph` : lettres majuscules.

Par exemple, pour obtenir un document qui commence par la numérotation romaine pour l'introduction, la table des matières,... et qui se poursuit par la numérotation arabe à partir du premier chapitre, la commande `\pagenumbering{roman}` se place dans le préambule ; en plaçant `\pagenumbering{arabic}` juste avant ou après la commande `\chapter`, on redéfinit la numérotation chiffrée au début du premier chapitre. Attention cependant qu'à chaque changement de style, la commande de numérotation réinitialise le numéro de page à 1.

Des styles de mise en page plus raffinés peuvent être obtenus à l'aide de modules additionnels, tels le module `fancyheadings` disponible avec documentation sur le serveur `http:\\www.ctan.org` ; c'est en particulier à l'aide de ce module qu'est réalisée la mise en page de ce document.

2.3 Définitions de commandes et d'environnements

Le préambule comprend un ensemble de commandes optionnelles, définies en début de document, et utilisées tout au long du document. Il existe une large gamme, permettant de définir des environnements, des théorèmes mathématiques, des commandes personnelles ou l'usage de fichiers prédéfinis. Les paragraphes qui suivent décrivent les principaux éléments susceptibles de se retrouver dans le préambule.

`\newcommand{`*nom*`}[`*narg*`]{`*définition*`}`

permet de créer une commande *nom* qui remplace l'écriture d'un texte ou d'une commande fastidieuse. Lors de la compilation, chaque occurrence de *nom* sera remplacée par *définition* ; si la commande définie admet un ou des arguments, *narg* doit être ajouté pour spécifier le nombre et ceux-ci apparaissent dans la définition de la commande précédés du caractère `#`. Par exemple :

```
\newcommand{\bdm}{\begin{displaymath}}
\newcommand{\ul}[1]{\underline{#1}}.
```

S'il s'agit de redéfinir une commande déjà connue du compilateur ou d'utiliser son nom pour une autre action que celle prévue (ce qui n'est jamais vraiment conseillé), il faut alors utiliser `\renewcommand{`*nom*`}[`*narg*`]{`*définition*`}`, qui s'utilise exactement de la même manière que l'opération précédente, la nouvelle définition prenant alors le pas sur la définition classique.

`\newenvironment{`*nom*`}[`*narg*`]{`*avant*`}{`*après*`}`

permet de définir un nouvel environnement appelé `nom` et qui peut être défini par `narg` arguments. `avant` reprend l'ensemble des commandes effectuées avant l'entrée du contenu de l'environnement, et `après` les commandes qui suivent ce contenu.

Dans le préambule :

```
\newenvironment{cbox}
{ \begin{center}
  \begin{tabular}{|c|}
  \hline
}
{ \\ \hline
  \end{tabular}
  \end{center}
}
```

Texte encadré et centré.

Dans le document :

```
\begin{cbox}
Texte encadré et centré.
\end{cbox}
```

`\renewenvironment{`*nom*`}[`*narg*`]{`*avant*`}{`*après*`}`

s'emploie exactement comme `newenvironment` mais permet de redéfinir un environnement déjà reconnu par le compilateur.

`\newtheorem{`*nom*`}{`*titre*`}`

permet de définir des environnements mathématiques appelés `nom` qui, lorsqu'ils sont utilisés, appellent l'affichage du titre `titre`, de la numérotation des éléments de la même structure ; le texte à l'intérieur de l'environnement étant imprimé en italique.
Ex. `\newtheorem{thrm}{Théorème}`. La section 5.1.3 spécifie plus en détails l'utilisation de ces structures et les options dont elles disposent.

2.4 Découpage du fichier d'entrée

Lorsqu'on rédige un long document, il n'est pas intéressant de travailler dans un seul fichier. En effet, le temps de sauvegarde devient plus long et la navigation n'y est pas toujours aisée, surtout que l'on travaille dans un éditeur de textes.

Une alternative est de morceler le texte, par exemple en plaçant chaque chapitre dans un fichier différent, de même pour la bibliographie et l'introduction. On crée alors un fichier principal qui contiendra la structure générale du document avec son préambule. À la suite de celui-ci seront introduites les commandes d'inclusion des fichiers de chapitres et éventuellement des commandes pour la tables des matières, la numérotation des pages,... . LaTeX propose deux manières d'insérer le contenu des sous-fichiers dans le fichier principal.

`\input`[*file*]

importe le texte du fichier *file* dans le fichier qui contient la commande, exactement comme si ce texte faisait partie de ce document. Cette commande peut être utilisée n'importe où dans le document et peut même apparaître dans un fichier appelé par une de ces deux commandes.

`\include`[*file*]

importe le texte du fichier *file* dans le fichier qui contient la commande, exactement comme si ce texte faisait partie de ce document, au fait près qu'il insère un saut de page avant et après ce texte. Par ailleurs, cette commande ne peut ni apparaître dans le préambule, ni dans un document inclus par la commande elle-même.

Une autre commande est associée à cette dernière pour la compilation de fichiers en cours de modification, afin de gagner du temps sans perdre les références et la numérotation :

`\includeonly`[*file-list*]

conditionne les inclusions de la commande `\include`. C'est-à-dire que si le fichier *file* d'une des commandes `\include` utilisées est un des noms de la liste *file-list* (liste de noms séparés par des virgules), le fichier est compilé par LaTeX alors que s'il n'apparaît pas dans la liste, les références et les entrées de tables récapitulatives sont conservées mais le fichier n'est pas compilé. Il y a alors un gain de temps appréciable lors de la compilation. La commande `\includeonly` ne peut apparaître que dans le préambule du document principal.

3

Mise en page fondamentale

3.1 Structure du document

3.1.1 Génération d'un titre

Il existe une fonction qui permet la mise en page automatique du titre principal d'un document : cette fonction tient en deux étapes. Dans un premier temps, les éléments que l'on peut trouver dans le titre sont définis au moyen des commandes

`\title{`*titre*`}`, `\author{`*auteur*`}` et `\date{`*date*`}`.

Ces commandes peuvent être regroupées dans le préambule afin d'être facilement retrouvables. Dans un deuxième temps, la commande

`\maketitle`

est insérée juste après le début du document : le compilateur se charge automatiquement de prendre les informations des commandes définies au préalable et de les mettre en page.

S'il y a plusieurs auteurs, leurs noms seront séparés dans la définition `\author` par l'instruction `\and`. Pour afficher la date du jour dans `\date`, il suffit d'utiliser l'instruction `\today` (cette commande peut également être utilisée en dehors de la définition de la date) ; enfin si la date s'affiche automatiquement et qu'on souhaite s'en débarrasser, il suffit d'utiliser la commande vide : `\date{}`.

3.1.2 Subdivisions du texte

Lorsque l'on rédige un document, il faut le subdiviser en parties logiques numérotées et correctement espacées. La méthode de subdivision d'un document dépend de son style. Il existe sept possibilités de subdivisions mais elles ne sont pas toutes utilisables avec certains styles.

Les sept commandes permettant de créer les subdivisions sont

```
\part          \subsection          \paragraph
\chapter       \subsubsection       \subparagraph
\section
```

Toutes ces commandes doivent avoir un argument qui est le titre de la subdivistion que l'on désire créer. Par exemple, pour créer cette section, nous avons introduit la commande `\subsection{Subdivision du texte}`.

Ces commandes introduisent la hiérarchie du texte, que le compilateur peut extraire pour réaliser automatiquement la table des matières (voir section 6.1). En cas de titre trop long, on peut ajouter en argument optionnel un autre titre, qui apparaîtra dans la table des matières, comme par exemple
`\section[Les finances du FNRS]{Les finances du Fonds National pour la Recherche Scientifique}`.

Enfin, une subdivision étoilée * produit un titre qui n'est pas numéroté et n'apparaît pas dans la table des matières : c'est le cas si l'on remplace `\chapter{Introduction}` par `\chapter*{Introduction}`.

L'effet de ces commandes n'est pas toujours identique dans tous les styles. Par exemple, dans un style, une commande provoque un saut de page et dans l'autre pas (voir notamment les options de classes à la section 2.1). Dans la classe "livre" (book), on peut trouver toutes ces subdivisions. Dans un article par contre, il ne peut y avoir de partie ni de chapitre, mais toutes les autres commandes sont disponibles. Dans une lettre, il ne peut y avoir aucune séparation.

Trois commandes supplémentaires permettent de subdiviser un document de type `book` en trois parties (par exemple le prologue, le texte et l'épilogue) : ces trois parties sont délimitées par les instructions `\frontmatter`, `\mainmatter` et `\backmatter`, qui agissent sur la mise en page des titres et sur la numérotation.

Par défaut, LaTeX numérote les chapitres en indiquant le mot "Chapitre" suivi du numéro et du nom du chapitre à la ligne suivante[1]. Ensuite, les sections sont numérotées en faisant référence au chapitre dans lequel elles se trouvent et le nom suit directement le numéro. Par exemple, la deuxième section du troisième chapitre sera numérotée "3.2". Un principe similaire est appliqué pour les sous-sections mais ensuite, il n'y a plus de numérotation. On dira que la profondeur de numérotation est de 2 (car deux numéros peuvent suivre le numéro de chapitre).

[1] Par défaut, on verra apparaître "Chapter" et le numéro ; c'est en chargeant la partie francophone de "Babel" que parties et chapitres seront libellés en français.

Il est possible de modifier la profondeur de numérotation de sorte de LaTeX pratique une numérotation plus en profondeur. Cela est obtenu en redéfinissant le compteur *secnumdepth* au moyen de la commande

`\setcounter{secnumdepth}{`*num*`}`

si on désire une numérotation jusqu'à la profondeur *num*, par exemple 3 pour la numérotation jusqu'aux sous sous-sections.

Dans un livre ou un rapport, il est encore possible d'ajouter des annexes en fin de document. On insère alors la commande

`\appendix`

après la fin du dernier chapitre, et avant le début des annexes. chacune d'entre elle est alors introduite comme un chapitre, mais dans ce cas la commande `\chapter` provoquera non plus l'affichage du mot *Chapitre* avec un chiffre, mais bien le mot *Annexe* avec une lettre. La suite des numérotations est bien entendu adaptée.

Connaissant les éléments de base qui permettent de réaliser un document, il est à présent possible d'aborder le document lui-même, et en particulier sa mise en forme. LaTeX est en effet un outil qui génère sa propre mise en page, mais il est possible de forcer certains paramètres à une valeur définie par l'utilisateur.

3.2 Taille des caractères

La taille des caractères est en premier lieu définie par les options de style spécifiées au début d'un document. En plus de cela, il existe 10 tailles que l'on peut spécifier par les commandes du tableau 3.1, qui s'utilisent selon la syntaxe de l'exemple suivant : `{\large` *texte* `}`

Tab. 3.1 – Tailles des caractères

Bon	`\tiny`	Bon	`\large`
Bon	`\scriptsize`	Bon	`\Large`
Bon	`\footnotesize`	Bon	`\LARGE`
Bon	`\small`	Bon	`\huge`
Bon	`\normalsize`	Bon	`\Huge`

Dans le document, le changement de taille de caractères est relatif à la taille par défaut spécifiée dans les options de style du document. De ce fait, un changement de cette taille générale n'affecte en rien les autres changements. Si par exemple, dans un paragraphe, on désire noter en petit une remarque entre parenthèses, on place alors ce texte entre des accolades avec l'instruction `\small` au début. Si la taille par défaut est de 12 points, la taille de ce texte sera de 11 points alors que si la taille par défaut est 11, elle sera de 10.

3.3 Polices de caractères

La police par défaut de LaTeX est la police “roman”. D'autres polices, reprises dans le tableau suivant, sont également disponibles pour mettre en évidence certaines parties de texte. La définition est la même que pour le choix de la taille : des accolades avec la commande en début à l'intérieur de celles-ci. À gauche, on trouve le résultat de l'instruction placée à droite.

Tab. 3.2 – Aspect de la police de caractères

This is a bold style.	{\bf This is a bold style.}
This is an italic style.	{\it This is an italic style.}
This is an emphasized style.	{\em This is an emph. style.}
This is a sans serif style.	{\sf This is a sans serif style.}
This is a slanted style.	{\sl This is a slanted style.}
This is a Small Caps style.	{\sc This is a Small Caps style.}
`This is a typewriter style.`	{\tt This is a typewriter style.}
This is a roman style.	{\rm This is a roman style.}

Dans le cas où la police italique (ou emphasized) ne porte que sur une partie d'un mot, il faut ajouter `\/` avant de repasser en mode normal. Ainsi *bon*jour et *bon*jour sont différents et obtenus en tapant respectivement `{\em bon}jour` et `{\em bon\/}jour`. La différence se marque dans l'espacement entre le n et le j.

Par ailleurs, deux instructions du même changement de police ont un effet d'annulation. Ainsi l'instruction

```
{\em Un {\em texte accentué} dans un texte accentué}
```

produit le texte “*Un* texte accentué *dans un texte accentué*”.

Il est également possible de souligner du texte, mais la syntaxe est quelque peu différente : “<u>This is an underlined style.</u>” est produit par `\underline{This is an underlined style.}` Dans cet ordre d'idées, les dernières versions du compilateur LaTeX ont introduit une méthode alternative pour imposer les aspects de polices, calquée sur les commandes générales, c'est-à-dire à l'aide d'une commande et du texte auquel elle s'applique en argument. Cela donne

`\textbf{`*texte*`}`

Le suffixe -bf peut alors être remplacé par tous les éléments présentés dans le tableau 3.2, à l'exception de la mise en emphase qui s'obtient par la commande `\emph{`*texte*`}`.

Enfin, pour obtenir un style d'écriture large et gras, il faut utiliser la combinaison de commandes `\large\bf` et non l'inverse.

3.4 Alignement du texte

Dans la plupart des ouvrages, le texte est justifié, c'est-à-dire qu'une ligne de texte débute à la marge de gauche et se termine à la marge de droite, les espaces entre les mots étant ajustés en fonction de cet impératif. Il en va de même pour une texte mis en page avec LaTeX si aucune information supplémentaire ne lui est communiquée. Pour certains extraits cependant, on peut choisir d'aligner le texte à gauche ou à droite, ou de le centrer sur la largeur de la page. Ces trois types d'alignements sont définis par les environnements `center`, `flushleft` et `flushright`. Puisque ce sont des environnements, la syntaxe générale est

`\begin{`*envtype*`}`
texte
`\end{`*envtype*`}`

où `envtype` est un des trois environnements précités :

`center` est utilisé pour produire une ou plusieurs lignes de texte centré sur la page. La commande `\\` permet de passer à la ligne. Les paragraphes sont également permis mais ne sont pas indentés ;

`flushleft` produit un texte aligné à la marge gauche et non justifié à droite ;

`flushright` produit un texte aligné à la marge de droite et non justifié à gauche.

Ce texte est aligné à gauche et donc non justifié à droite.

```
\begin{flushleft}
Ce texte est aligné à gauche
et donc non justifié à droite.
\end{flushleft}
```

Ce texte est centré sur la colonne de ce tableau : il n'est justifié ni à gauche, ni à droite.

```
\begin{center}
Ce texte est centré sur
la colonne de ce tableau:\\
il n'est justifié ni à
gauche, ni à droite.
\end{center}
```

Ce texte est aligné à droite et donc non justifié à gauche.

```
\begin{flushright}
Ce texte est aligné à droite
et donc non justifié à gauche.
\end{flushright}
```

Les commandes

`\centering`, `\raggedleft` et `\raggedright`

modifient les paramètres de mise en page de LaTeX et s'utilise *à l'intérieur d'un environnement* pour produire un effet similaire aux trois environnements définis ci-dessus. Le placement d'une de ces commandes dans l'environnement principal (document) change donc la mise en page de la suite du document tout entier.

3.5 Les paragraphes

Il convient de rappeler ici qu'un retour de chariot (*enter*) constitue pour le compilateur un espace et non un début de paragraphe. Le passage au paragraphe suivant s'effectue donc en laissant dans le fichier de code une ligne entièrement vide.

La définition des paragraphes est propre au compilateur, ce qui signifie qu'il définit lui-même le retrait en début de paragraphe ainsi que l'espacement entre les paragraphes. En particulier, signalons que par défaut, LaTeX n'indente pas les paragraphes (i.e. il n'insère pas de retrait au début), conformément à la mise en page anglosaxonne et en opposition aux conventions francophones. Pour modifier cette caractéristique, on peut charger dans le préambule la partie francophone du module "babel" (`\usepackage[french]{babel}`)[2], ce qui adapte également d'autres caractéristiques telles que les énumérations. Il est encore possible de modifier uniquement la définition du retrait de paragraphe, en la rendant effective au moyen des lignes

```
\let\@afterindentfalse\@afterindenttrue
\@afterindenttrue
```

incluses dans le préambule. Le plus simple afin d'éviter l'insertion de ces lignes au début de tous les documents est de les placer dans un fichier de style réservé à cela `indent.sty` ou propre à l'utilisateur (ex. `mystyle.sty`).

Bien que ces valeurs soient définies par le compilateur, il est encore possible de redéfinir l'espace additionnel entre les paragraphes et la longueur du retrait. Il suffit de donner respectivement aux dimensions `\parskip` et `\parindent` une nouvelle valeur, en ajoutant dans le préambule l'instruction `\parskip=5pt` ou `\setlength{\parskip}{5pt}` par exemple.

Il est encore possible de forcer l'indentation ou d'éviter une indentation par défaut au moyen des commandes respectives

`\indent` et `\noindent`.

Cette dernière est notamment intéressante lorsqu'en mise en page francophone, on souhaite bénéficier de l'espacement par défaut après l'écriture d'une formule mathématique en évitant le retrait.

Finalement, l'extension `french` de `babel` permet encore de commencer un paragraphe par une lettrine, au moyen de la commande

`\lettrine{`*texte*`}`.

3.6 Les coupures

Lors de la compilation, LaTeX décide seul des césures de mots, des sauts de lignes et de pages, afin de produire un résultat optimal. Toutefois les règles sur lesquelles il reposent sont adaptées à une mise en page anglosaxonne et sont

[2] Il existe également dans certains modules une commande nommée "french", c'est la raison pour laquelle un deuxième module babel a été développé pour le français, appelé par `frenchb` au lieu de `french`. Il possède exactement les mêmes caractéristiques.

parfois mal adaptées à la mise en page d'un texte en français, même lors de l'utilisation du module Babel. Il existe dès lors des commandes permettant de forcer certaines coupures.

Césure d'un mot

Les instructions suivantes permettent d'agir sur la découpe d'un mot lorsque le compilateur ne sait pas prendre de décision ou choisit un découpage en conflit avec les règles du français :

`\-` Utilisée dans un mot, cette commande suggère à LaTeX de couper le mot à cet endroit si nécessaire ; elle n'a donc pas d'effet *coercitif*. Lorsque plusieurs commandes se trouvent dans un même mot, le compilateur peut choisir l'endroit où il coupera le mot. Ainsi l'écriture `envi\-ron\-ne\-ment` indique que, si le mot "environnement" doit être coupé, il le sera obligatoirement en un des trois endroits spécifiés ; le seul trait d'union qui apparaîtra sera celui de la coupure.

`\fussy` et `\sloppy` définissent la manière dont LaTeX réagit lorsqu'il se trouve face à un mot dépassant dans la marge de droite et qu'il ne peut couper. La valeur par défaut est `\fussy` : si le texte est trop long pour tenir sur la ligne et qu'une coupure ne peut être effectuée en fonction des règles du logiciel, il y a génération d'un avertissement (dans le fichier *name.log*) et le texte est laissé sur la ligne, dépassant ainsi dans la marge de droite. Avec la commande `\sloppy`, le texte est mis à la ligne (un avertissement est également produit dans le fichier d'information). Ces commandes s'appliquent jusqu'à la fin de l'environnement courant. L'action de l'instruction `\sloppy` peut également être restreinte à un paragraphe, que l'on enferme alors dans un environnement `sloppypar` : en indiquant

```
\begin{sloppypar}
pars
\end{sloppypar};
```

le(s) paragraphe(s) *pars* est (sont) alors mis en page en respectant l'option `\sloppy`.

`\hyphenation{`*mots*`}` déclare une liste de mots pouvant être coupés. Les coupures autorisées sont indiquées par des tirets (-). Cette déclaration est globale (valable dans tout le document à partir du point où elle se situe) et donc, se trouve généralement dans le préambule afin de prendre cours dans l'ensemble du document courant.
Par exemple, `\hyphenation{envi-ron-ne-ment}` autorise la coupure du mot *environnement* à trois endroits uniquement. Si le mot utilisé avec cette commande ne contient aucun signe de césure, alors il ne sera jamais coupé. Signalons encore que pour cette instruction le compilateur n'est pas sensible à la casse des caractères.

`\mbox{...}` L'utilisation d'une boîte permet aussi d'imposer qu'un mot ou un ensemble de caractères (par exemple un numéro de téléphone) *ne soit pas* coupé.

3.6.1 Sauts de lignes

Rappelons que les retours chariots dans le fichier d'entrée sont interprétés comme des espaces lors de la compilation d'un texte et une ligne vide de tout caractère indique un nouveau paragraphe. Il est cependant possible de forcer un retour à la marge à l'intérieur d'un paragraphe de deux manières différentes.

`\linebreak[num]` et `\nolinebreak[num]` encourage (ou décourage) une coupure de ligne à l'endroit où se trouve la commande. La ligne est alors justifiée à droite. L'argument optionnel `num` est un nombre de 0 à 4 (4 par défaut) qui caractérise la liberté que l'utilisateur laisse au compilateur dans l'interprétation de la commande, allant de la liberté d'ignorer l'instruction (0) à l'obligation de l'appliquer (4). Un avertissement (`underfull hbox`) est produit dans le fichier d'information si trop d'espace entre les mots résulte de cette commande.

`\\[len]` et `\newline` provoquent une coupure de ligne sans justification de la ligne sur laquelle cette coupure se produit et sans retrait de paragraphe pour la nouvelle ligne. Le texte est aligné à gauche. Le paramètre optionnel `len` permet d'ajouter un espace vertical supplémentaire entre les lignes. Ces commandes ne peuvent se trouver en fin de paragraphe sinon elles provoquent un avertissement.

3.6.2 Colonnes et pages

Tout comme pour les lignes, LaTeX dispose au mieux le texte sur la hauteur de la page (ou de la colonne si le texte est imprimé sur deux colonnes par page). Cependant, les règles qu'il suit peuvent le conduire à terminer la page à un endroit inopportun ; il est alors nécessaire de le corriger manuellement en insérant un saut de page ou lui interdisant une coupure à un endroit particulier.

Les commandes qui sont décrites ci-dessous ont souvent un comportement différent suivant le style du document ainsi qu'en fonction de ses options de présentation. Elles ont aussi une influence conséquente sur la place des éléments flottants (tables et figures), qui peuvent être reportés après un saut de page si on n'y prend garde.

`\pagebreak[num]` et `\nopagebreak[num]` suggère ou interdit respectivement le passage à une nouvelle page (ou colonne). Le paramètre optionnel `num` (nombre de 0 à 4, par défaut 4) caractérise la liberté laissée au compilateur, allant de la liberté d'ignorer l'instruction (0) à l'obligation de l'appliquer (4). Lorsqu'elles sont utilisées dans un paragraphe, elles s'appliquent à la fin de la ligne dans laquelle elles apparaissent.

`\clearpage` oblige LaTeX à afficher tous les tableaux et figures qui ne le sont pas encore et à débuter une nouvelle page.

`\cleardoublepage` agit comme `\clearpage` et en plus, dans le cas d'un texte sur deux faces, force le passage à une page de droite.

`\newpage` force le passage à une nouvelle page mais ne force pas l'affichage des tableaux et figures (qui peuvent donc être projetés également à la page ou la colonne suivante).

`\enlargethispage{`*size*`}` permet de forcer LaTeXà agrandir la zone effective sur une page afin de contenir une ligne en plus ou en moins. L'argument *size* indique la longueur d'allongement (positive) ou de réduction (négative) de la page.

3.7 Environnements particuliers

3.7.1 Citations et poèmes

LaTeX propose trois environnements pour les citations dans lesquels les marges gauche et droite sont déplacées de la même distance. Un poème est mis en page d'une manière similaire à celles-ci.

`quote` est utilisé pour de brèves citations, séparées par un ligne vide dans le texte d'entrée. Il n'y a pas de retrait de paragraphe et un espace vertical supplémentaire est ajouté entre les paragraphes.

Ceci constitue le texte de la première phrase.

Ceci est la deuxième phrase.

```
\begin{quote}
Ceci constitue le texte
de la première phrase.

Ceci est la deuxième phrase.
\end{quote}
```

`quotation` est utilisé pour des citations de plus d'un paragraphe, séparés par une ligne vide également. Il y a un retrait de paragraphe et l'espace normal entre les paragraphes est utilisé.

`verse` est utilisé pour un poème. Les lignes d'une même strophe sont séparées par `\\` et les strophes sont déterminées par une ou plusieurs lignes vides.

> Les gens qui n'ont plus rien à faire
> Se suivent dans la rue comme
> Des wagons de chemin de fer.
>
> Mais pour quoi faire ?

```
\begin{verse}
Les gens qui n'ont plus rien à faire\\
Se suivent dans la rue comme\\
Des wagons de chemin de fer.

Mais pour quoi faire?\\
\end{verse}
```

`poetry` Cet environnement est similaire à `verse`. Une nouvelle strophe s'obtient en laissant une ligne vide. Une nouvelle ligne s'obtient par un `\\`.

3.7.2 Texte littéral

L'environnement `verbatim` restitue en sortie le texte entré dans le fichier source *name.tex* en respectant tous les caractères (même réservés) ainsi que les espaces et les fins de ligne. La police utilisée est de type *typewriter*.

Cet environnement commence une nouvelle ligne et passe à la ligne lorsqu'il se termine avec ajout d'un espace vertical. Il est cependant possible de produire quelques caractères affichés exactement comme introduits mais qui ne se trouvent pas sur une ligne séparée. La commande à utiliser est

`\verb`

et sa syntaxe est un peu particulière. Cette commande prend un argument (le texte à afficher) délimité par une paire de caractères identiques (à l'exception de l'espace, de `*` et d'une lettre). Ainsi les commandes `\verb-$\é &-` et `\verb+$\é &+` produisent le même texte qui est `$\é &`.

Il existe également l'environnement `verbatim*` et la commande `\verb*`. Elles agissent comme `verbatim` et `\verb` au fait près qu'un espace produit le caractère ␣. Ainsi `\verb*-$\é &-` produit `$\é␣&`.

Il y a deux restrictions à l'utilisation de cet environnement et de cette commande :

(1) ils ne peuvent apparaître dans un argument d'une commande (donc notamment pas dans une note de bas de page !) mais bien dans un autre environnement,

(2) il ne peut y avoir aucun espace entre la commande `\verb` ou `\verb*` et son argument.

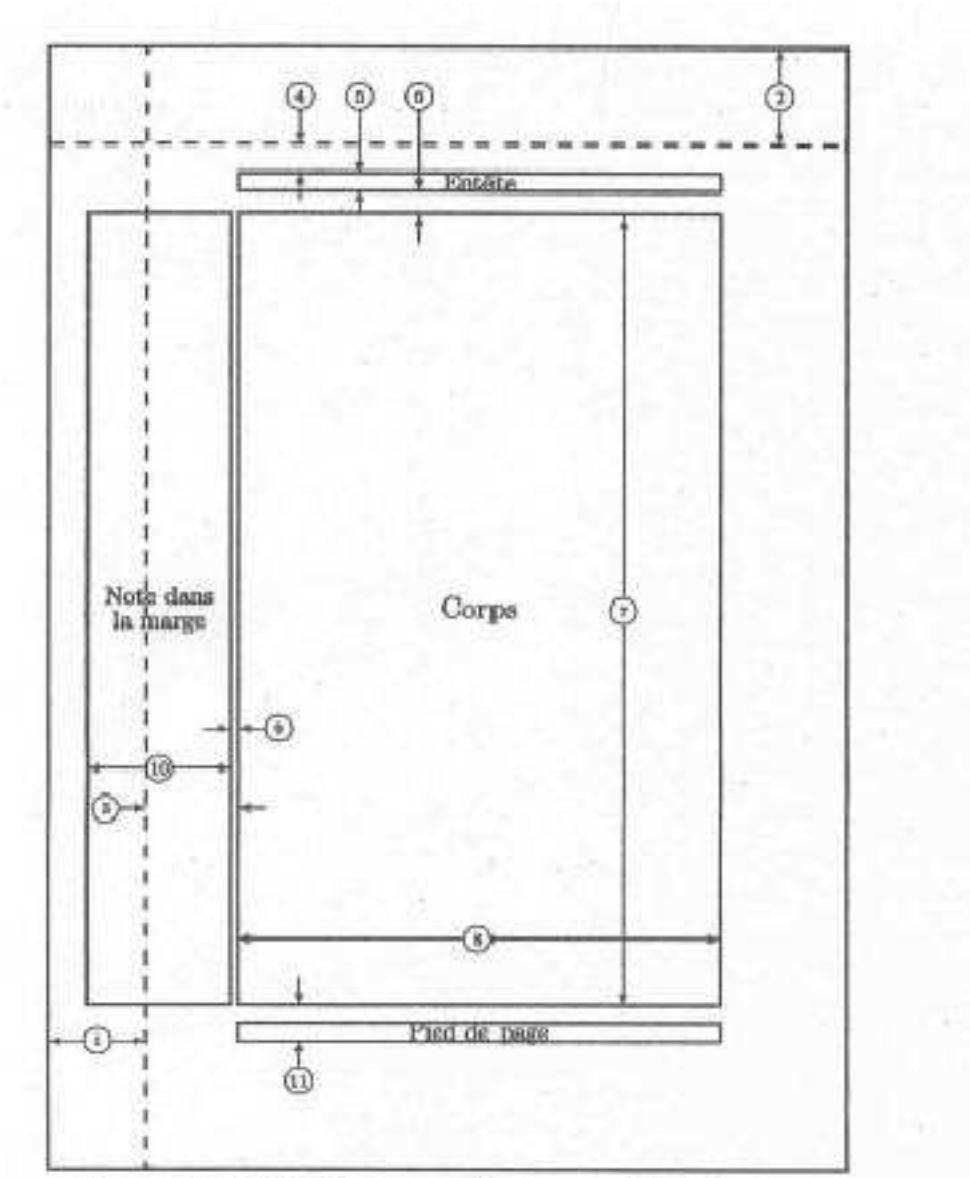
3.8 Disposition d'une page

Bien que LaTeX calcule automatiquement les marges en fonction de la taille de papier spécifiée dans les paramètres de `\documentclass`, l'utilisateur peut vouloir modifier certaines valeurs. Ces modifications sont introduites par les deux commandes

`\setlength{`*paramètre*`}{`*longueur*`}`
`\addtolength{`*paramètre*`}{`*longueur*`}`

La première détermine une valeur fixe au *paramètre* considéré, alors que la seconde opère en relatif, en ajoutant une longueur (éventuellement négative) à la valeur par défaut ; en pratique, la seconde est plus aisée à manipuler.

Les différents paramètres sur lesquels il est possible d'agir sont définis à la figure 3.1.



1 un pouce + \hoffset
2 un pouce + \voffset
3 \evensidemargin = 70pt
4 \topmargin = 22pt
5 \headheight = 13pt
6 \headsep = 19pt
7 \textheight = 595pt
8 \textwidth = 360pt
9 \marginparsep = 7pt
10 \marginparwidth = 106pt
11 \footskip = 27pt
\marginparpush = 5pt (non affiché)
\hoffset = 0pt
\voffset = 0pt
\paperwidth = 597pt
\paperheight = 845pt

FIG. 3.1 – Paramètres de la disposition d'une page

4 Eléments additionnels dans le texte

4.1 Références numérotées

4.1.1 Références croisées

Dans un document structuré, il est parfois intéressant d'éviter une répétition en référant le lecteur à un élément cité ailleurs, que ce soit un numéro de page où se trouve une certaine information, un numéro de théorème, d'équation, un tableau, une figure, ou encore une subdivision du document (chapitre, section,...). Ces relations intratextuelles sont appelées *références croisées*.

Le principe est simple : à chaque élément numéroté qui pourra être rappelé on associe une *étiquette* avec un nom particulier ; c'est ce nom qui sera rappelé pour faire apparaître le numéro de l'élément auquel on réfère.

L'étiquette est définie par l'ajout de la commande

`\label{`*crossref*`}`[1]

juste à côté de l'élément numéroté. Cette commande ne produit aucun texte dans le document, mais lors de la compilation, LaTeX inscrit dans son fichier *name.aux* le nom `crossref` de l'étiquette et le numéro auquel elle se réfère.

À l'endroit où l'on souhaite faire réapparaître ce numéro, on insérera la commande

`\ref{`*crossref* `}`

et le compilateur remplacera automatiquement cette commande par le numéro correspondant.

[1] Le nom de l'étiquette `crossref` sera judicieusement choisi de manière à faciliter le rappel ultérieur.

L'appel d'une étiquette par la commande `\ref{ }` peut apparaître dans le texte avant ou après la définition de l'étiquette par `\label{ }`. Il est nécessaire de compiler deux fois le document ; ainsi lors de la première fois LaTeX génère un fichier d'information qui contient toutes les références et lors de la seconde le compilateur insère les références appropriées.

Une étiquette peut être utilisée à l'intérieur d'un environnement portant un numéro (les théorèmes, les équations,...). Il faut être prudent lors de l'emploi d'une telle référence dans un tableau d'équations (voir également `\eqnarray`) : dans ce cas l'étiquette sera insérée après l'écriture de l'équation et avant l'instruction de retour de ligne. Lorsqu'elle est utilisée avec la commande `\caption` dans la description d'une figure ou un tableau, la commande `\label` doit se trouver après cette instruction. Enfin, pour référer à un numéro de chapitre ou de section, la commande de définition de l'étiquette suivra immédiatement la définition de la subdivision.

Enfin, exactement comme pour le numéro, il est possible d'insérer la page associée à une étiquette dans le texte au moyen de la commande

`\pageref{`*crossref*`}`

lors de la compilation cette commande sera remplacée par le numéro de page à laquelle apparaît l'étiquette appelée `crossref`.

4.1.2 Références bibliographiques

Une partie importante d'un document est sa bibliographie, dont la mise en page doit être rigoureuse et claire. Il existe avec LaTeX un environnement spécialement conçu pour cela (voir section 6.4), qui regroupe l'ensemble des éléments bibliographiques dans une annexe numérotée : au sein de celle-ci, chaque élément est introduit au moyen de la commande

`\bibitem{`*biblioref*`}`[2].

La référence intratextuelle à des éléments bibliographiques peut respecter plusieurs conventions au niveau de la numérotation : la plus simple est obtenue par l'utilisation de la commande

`\cite[`*texte*`]{`*biblioref*`}`

qui produit l'affichage de l'étiquette associée à `\bibitem{biblioref}`. Si l'argument optionnel *texte* est présent, il est ajouté comme commentaire à cette référence.

[2] Comme dans le cas des références croisées, le nom `biblioref` sera choisi de manière explicite, par exemple à partir du nom de l'auteur.

4.1.3 Notes de bas de pages

L'insertion d'une note de bas de page attachée à un mot est réalisée au moyen de la commande

`\footnote{`*texte* `}`

disposée juste après ce mot : la numérotation au moyen de lettres minuscules sera automatiquement réalisée, et le *texte* de la note sera inscrit au bas de la page en cours.

Les notes de bas de page[a] sont faciles à réaliser.

[a] Voici la note.

```
Les notes de bas de
page\footnote{Voici la
note.} sont faciles à
réaliser.
```

4.2 Les listes et énumérations

LaTeX propose trois environnements permettant la structuration d'éléments au sein d'une liste. Ceux-ci se construisent sur le modèle général

`\begin{`*envliste*`}`
`\item[`*opt*`]` *text*
...
`\end{`*envliste*`}`

où *envliste* détermine le type de liste généré ; chaque élément *text* de la liste est introduit par l'instruction `\item` qui peut comporter une ou des options *opt* agissant sur la mise en page. LaTeX autorise qu'un élément d'une liste soit également une liste. Un élément d'une liste insérée dans le document est de profondeur 1. Si un élément d'une liste est encore une liste, les éléments de cette deuxième liste auront une profondeur 2 et ainsi de suite.

4.2.1 Liste de points

Une simple liste est produite par l'environnement `itemize`. Par défaut, celui-ci précède chaque nouvel élément par un symbole dépendant du style et de sa profondeur. Le choix des symboles associés aux listes dépend notamment du style de mise en page : ainsi par défaut, LaTeX utilise la mise en page anglosaxonne pour laquelle le premier symbole utilisé est un gros point (•) ; en utilisant la mise en page francophone de `babel`, le premier symbole utilisé est un tiret (-).

Un élément d'une liste peut également être une liste : un retrait à droite supplémentaire est alors imposé à chaque niveau de profondeur. La mise en page anglosaxonne impose par ailleurs un changement de symbole à chaque niveau, ce qui ne se retrouve pas dans `babel`.

Le choix du symbole peut être imposé par l'argument optionnel. Ce changement peut n'être imposé qu'à un seul élément dans une liste : le symbole modifié

est alors aligné avec les autres symboles de même niveau et peut même déborder dans la marge de gauche sans provoquer d'erreur.

Voici une liste :
– Un élément,
– this is an English list :
  • One item,
  • another one.
* le troisième est introduit par un autre symbole.

```
Voici une liste:
\begin{itemize}
\item Un élément,
\item this is an English list:
\begin{itemize}
\item One item,
\item another one.
\end{itemize}
\item[*]le troisième est
introduit par un autre
symbole.
\end{itemize}
```

4.2.2 Énumération

L'utilisation de l'environnement `enumerate` permet de lister des éléments suivant des symboles incrémentés, que ce soit des chiffres ou des lettres. S'il y a plusieurs niveaux d'énumération, chaque profondeur sera identifiée par un type de symbole différent : par exemple, un nombre pour la profondeur 1 et une lettre pour la profondeur 2.

Si un argument optionnel est ajouté pour un des éléments, il remplace encore une fois le numéro, mais ne provoque pas l'incrémentation du compteur correspondant : l'élément de la liste qui suivra directement cet élément porteur d'un argument adoptera la numérotation de l'élément qui précède celui-ci, augmentée d'une unité.

1. Un élément,
2. (a) un sous-élément,
   (b) et un autre.

* le troisième est introduit par un autre symbole,

3. et le quatrième suit le numéro du deuxième.

```
\begin{enumerate}
\item Un élément,
\item \begin{enumerate}
\item un sous-élément,
\item et un autre.
\end{enumerate}
\item[*]le troisième est
introduit par un autre
symbole,
\item et le quatrième
suit le numéro du
deuxième.
\end{enumerate}
```

4.2.3 Description

Une description se présente comme les deux précédents environnements mais il n'y a pas de symbole qui identifie un nouvel élément. Si aucun argument optionnel n'est fourni, le texte est simplement placé légèrement en retrait. Si l'environnement est utilisé pour décrire plusieurs termes, chaque terme sera introduit comme argument optionnel de la commande `\item` : ce terme sera alors placé en gras avec un retrait avant la ligne de texte, et servira d'identificateur de ligne.

mot 1 : la définition 1 ;
mot 2 : la définition 2.

```
\begin{description}
\item[mot 1]: la définition 1;
\item[mot 2]: la définition 2.
\end{description}
```

4.3 Objets flottants : tableaux et figures

Une particularité de LaTeX est le traitement des figures et tableaux , considérés par le compilateur comme des *objets flottants* : cela signifie que la place prise par ces objets ne sera pas systématiquement celle à laquelle ils ont été définis, mais ils constituent une entité que le compilateur place librement à l'endroit qu'il juge le plus opportun. Cette liberté peut sembler étrange, mais permet en réalité de rentabiliser au mieux l'espace de mise en page. En effet, supposons que l'on désire placer un tableau qui demande les trois-quarts d'une page sur une page qui contient déjà une moitié de texte, il y aura un blanc d'une demi page avant de voir apparaître ledit tableau à la page suivante. Avec le principe des éléments flottants, le tableau sera placé au début de la page qui suit mais du texte sera inséré sur la demi page qui aurait été vide (pour autant qu'il y ait du texte qui suit le tableau). Il existe cependant certaines options auxquelles l'utilisateur peut recourir pour forcer l'insertion de ces entités à un endroit imposé. Les deux éléments sont insérés par des structures semblables, mais ils apparaissent sous des noms différents et des numérotations distinctes :

- les tableaux (`table`) sont libellés par Tab. et
- les figures (`figure`) sont libellées par Fig..

L'insertion des éléments flottants s'effectue par la séquence

`\begin{`*env*`}[`*loc*`]`
contenu
`\end{`*env*`}`

où

env est le nom de l'environnement (`table` ou `figure`),

loc est une séquence optionnelle d'au plus 4 caractères qui sont `! h t b p`, désignant l'emplacement désiré :

- `h (here)` emplacement à l'endroit d'insertion dans le texte si cela est possible,
- `t (top)` en haut d'une page,

`b (bottom)` au bas de la page,

`p (page of floats)` sur une page séparée ne contenant que des éléments flottants.

`!` force à respecter ces paramètres même si la mise en page de LaTeX s'en trouve perturbée.

Si plusieurs caractères sont employés, le compilateur les considérera dans leur ordre d'apparition : ainsi avec `\begin{figure}[!htp]`, il va s'acharner (`!`) à placer la figure à l'endroit où est écrit ce code (`h`), s'il n'y parvient pas il essaiera de la placer au sommet d'une page (`t`) et si cela impose encore trop de contraintes il la placera sur une page séparée avec d'autres figures (`p`).

contenu est l'élément à placer.

Il peut être intéressant de s'assurer, avant de commencer un chapitre ou une nouvelle section, que les objets flottants de la subdivision précédente ont trouvé une place. Les commandes

`\clearpage` et `\cleardoublepage`

réalisent cela, en plaçant tous les objets en attente avant de commencer une nouvelle page ; la commande `\cleardoublepage` impose en plus de commencer ce qui suit sur une page de droite (voir également section 3.6.2).

Pour obtenir une légende ainsi qu'un numéro associés à l'élément, il faut utiliser la commande

`\caption[`*entree*`]{`*texte*`}`

à l'intérieur de l'environnement, soit au dessus ou en dessous de la commande de définition de l'objet en lui-même selon que l'on souhaite l'affichage de cette légende au dessus ou en dessous de celui-ci. Plusieurs commandes `\caption` peuvent apparaître dans un même environnement. Le texte optionnel *entree* sert d'entrée pour la liste des figures ou tableaux (voir 6.2) ; en l'absence de celui-ci, c'est le texte de la légende `texte` lui-même qui sera repris dans cette liste.

4.3.1 Les figures

LaTeX permet l'insertion dans un document de figures enregistrées dans le format Postscript encapsulé (`picture.eps`). Il est nécessaire dans ce cas de vérifier que l'imprimante accepte ce format. La reconnaissance des commandes qui vont être décrite nécessite le chargement du module `graphicx` à la compilation avec le pilote de conversion approprié : le programme le plus répandu est `dvips` ; le module et le pilote approprié font partie d'un ensemble d'extensions graphiques globalement appelées par le module `graphics` par l'insertion dans le préambule de l'instruction

`\usepackage{graphics}`.

L'insertion d'une figure dans le document est réalisée au moyen de l'environnement `figure`. Au sein de cet environnement, il faut ensuite ordonner le chargement de la figure ; on peut alors donner une légende à l'image.
La syntaxe complète d'une insertion de figure est la suivante :

```
\begin{figure}[loc]
\includegraphics[forme]{fichier}
\caption[lib]{texte}
\end{figure}
```

où

forme est un argument optionnel qui spécifie la taille de la figure et son orientation : cet argument prend les valeurs suivantes :

height définit la hauteur de la figure ;

width définit la largeur de la figure ;

angle impose un angle de rotation (positif dans le sens des aiguilles d'une montre).

Si on ne définit qu'une seule de ses dimension, l'autre est ajustée de manière à conserver la proportion ;

fichier est le nom du fichier ; si le répertoire dans lequel il est stocké est différent du répertoire courant, il faut spécifier le chemin complet ou écrire `../` autant de fois qu'il faut remonter d'un niveau dans l'arborescence.

Les autres paramètres sont explicités dans la description des commandes qu'ils précisent. L'exemple suivant permettra de visualiser l'emploi de ces commandes. Il s'agit d'insérer impérativement (`[!]`) à l'endroit spécifié (`[h]`) une figure dont le nom est `image.eps` en imposant sa taille, une rotation d'un quart de tour, centrée sur la page (environnement `center`) et en lui donnant une légende :

```
\begin{figure}[h!]
\begin{center}
\includegraphics[angle=90,width=10cm]{image.eps}
\caption{Voilà l'image.}
\end{center}
\end{figure}
```

4.3.2 Les tableaux

L'obtention d'une table s'obtient à l'aide de l'environnement `table` ; à l'intérieur de celui-ci on réalise alors la mise en forme du tableau proprement dit avec l'un des deux environnements `tabular` et `array`. Ceux-ci s'utilisent de manière similaire : la différence entre les deux réside dans le fait que le mode texte est utilisé pour les éléments des cellules dans un `tabular` alors que c'est le mode mathématique pour un `array`. Par conséquent, `array` est essentiellement à l'intérieur d'un environnement mathématique (voir 5.1) utilisé pour créer des matrices et `tabular` pour créer des tableaux insérés dans le texte. Signalons encore qu'un tableau peut être inséré dans le texte à l'extérieur

d'un environnement `table`, mais il ne constitue plus alors un objet flottant : utilisé seul, il n'imposera pas de retour de ligne et s'insérera à la suite du texte sur la ligne courante en respectant l'alignement du document ; pour une mise en page agréable il conviendra alors le plus souvent d'imposer un retour à la ligne `\\` avant celui-ci, et de l'inclure dans un environnement centré (`\begin{center}` ... `\end{center}`).

Un tableau est considéré par LaTeX comme une boîte (voir section 4.4) et donc doit être plus petit qu'une page.

La plupart des notions et des paramètres qui sont décrits ci-dessous sont communs aux deux environnements : ce qui est inscrit pour `tabular` reste donc valable pour `array` sauf mention explicite du contraire.

La syntaxe d'un tableau est

`\begin{tabular}[`*pos*`]{`*cols*`}`
lignes
`\end{tabular}`

où

pos est un argument optionnel qui spécifie la position en hauteur du tableau sur la ligne ; par défaut, le tableau est centré, un `t` aligne le dessus avec la ligne et un `b`, le dessous. C'est un paramètre courant pour les boites.

cols précise le format des colonnes. Chaque colonne doit être définie par un caractère qui spécifie la manière dont le texte est aligné ; entre ces caractères, il est possible de spécifier des éléments de remplissage ou de séparation :

l Une colonne où les éléments sont alignés à gauche.

r Une colonne où les éléments sont alignés à droite.

c Une colonne où les éléments sont centrés.

| Une ligne verticale entre deux colonnes.

@{*texte***}** Insère une colonne dans laquelle le *texte* est insrit dans toutes les lignes. Le *texte* est en mode texte ou mathématique suivant que l'on est dans un `tabular` ou un `array`. Cette commande annule l'espace normal entre les deux colonnes où elle se trouve. Cette option permet notamment d'écrire des tableaux de chiffres centrée sur le point décimal, comme en atteste l'exemple suivant :

```
\begin{tabular}{r @{.} l}
3   & 141592 \\
65  & 215    \\
245 & 1      \\
\end{tabular}
```

3.141592
65.215
245.1

p{*larg***}** produit une colonne de largeur *larg* où le texte est comme dans une commande `\parbox`.

{nbr***}{***cols***}** est équivalent à *nbr* copies de *cols* qui peut lui même contenir une *–expression.

lignes est une séquence de lignes qui remplissent le tableau. Chaque ligne est une séquence d'éléments séparés par `&` et il doit y avoir le même nombre d'éléments que de colonnes spécifiées par *cols*. Il ne peut pas y avoir un `&` après le dernier élément d'une ligne, mais le passage à la ligne suivante est indiqué par `\\`. Chaque élément est compilé comme s'il se trouvait entre accolades, et donc l'effet d'une macrocommande (de changement de caractères par exemple) se limite à cet élément.

Les commandes suivantes peuvent apparaître dans les lignes.

`\multicolumn`{*num*}{*col*}{*texte*} applique le *texte* dans une cellule réalisée par la fusion de *num* cellules dans la lignes où elle se trouve. Si *num*=1, alors la commande sert uniquement à changer la position de cet élément dans la cellule. *col* indique la position du texte dans la cellule fusionnée : ce paramètre peut contenir exactement les mêmes caractères que *cols* mais ne peut contenir qu'une seule colonne.

`\vline` Lorsqu'elle est utilisée à l'intérieur d'une colonne, elle produit une ligne verticale de la hauteur de la cellule.

Les instructions suivantes peuvent venir entre les lignes pour produire des lignes horizontales.

`\hline` produit une ligne horizontale sur toute la largeur du tableau.

`\cline{i - j}` produit une ligne horizontale de la colonne `i` à la colonne `j` incluses.

Ces commandes ne doivent pas être suivies de l'instruction de retour à la ligne `\\` car ce retour est automatique : si cette instruction est ajoutée, une ligne est passée.

La largeur des colonnes est automatiquement ajustée à la largeur du plus large élément de cette colonne. On peut inclure un tableau dans un tableau (récursivement).

L'exemple suivant reprend la plupart des commandes décrites :

Fruit	Vente		
	1995	1996	1997
Poire	6.5	7.8	7.2
Pomme	15	16	15.5
Prune	7.8	9	9.2

```
\begin{tabular}{|l|c|c|c|}
\hline
 & \multicolumn{3}{c|}{Vente}\\
\cline{2-4}
\multicolumn{1}{|c|}
{\raisebox{6pt}[0pt][0pt]{Fruit}}
  & 1995 & 1996 & 1997 \\
\hline
Poire & 6.5 & 7.8 & 7.2 \\
Pomme & 15 & 16 &15.5 \\
Prune & 7.8 & 9 & 9.2 \\
\hline
\end{tabular}
```

4.3.3 Paramètres additionnels

Si on utilise abondamment les éléments flottants, il se peut que le compilateur en regroupe librement plusieurs en début de page. Le nombre d'éléments et la proportion d'espace occupé par ceux-ci sont contrôlés par LaTeX au moyen des compteurs `\topnumber` et `\topfraction` respectivement. Par exemple, pour autoriser les éléments flottants à occuper au maximum 75% de la page, on insère la ligne suivante dans le préambule : `\renewcommand{\topfraction}{0.75}`. Les compteurs `\bottomnumber` et `\bottomfraction` permettent également de contrôler les espaces flottants dans le bas de la page.

Il existe aussi une commande permettant d'annuler les paramètres pour une page particulière. Plus précisément, il est possible de supprimer le placement d'éléments flottant sur une page au moyen de la commande

`\suppressfloats[`*pos*`]`

où *pos* est un argument optionnel qui vaut soit `b` soit `t`. Attention que l'utilisation d'un ! dans les arguments d'un élément flottant est prioritaire sur cette commande.

4.4 Les boîtes

Il est possible de définir certains éléments comme des objets que LaTeX ne peut diviser pour les placer correctement sur une ligne ou une page : ces élément sont créés en les enfermant dans des *boîtes*. Il existe trois sortes de boîtes suivant le mode de compilation du texte qui se trouve à l'intérieur. Lors de la compilation, LaTeX prend le style en cours pour le texte contenu dans une boite sauf si on se trouve en mode mathématique, auquel cas il prend le style qui était en cours juste avant. Bien entendu, en tant qu'objet indéformable, une boîte ne peut avoir des dimensions supérieures à celle de la page !

4.4.1 Le mode LR

Les commandes ci-dessous traitent le texte qui se trouve à l'intérieur en mode LR c'est-à-dire en mode texte où il ne peut y avoir de paragraphe. Ainsi, placer une telle commande en mode mathématique permet d'insérer quelques mots en mode textuel. Remarquons que, comme c'est une boîte dans laquelle les règles de définition des paragraphes ne s'appliquent pas, le texte doit tenir sur une seule ligne ou sur le restant de la ligne qui le contient.

`\makebox` permet de créer une boîte suivant la syntaxe

`\makebox{`*len*`}[`*pos*`]{`*texte*`}` où

len spécifie la longueur de la boîte et doit être une unité de mesure (inférieure à la largeur de la page). Par exemple `10mm`.

pos précise la position du texte au sein de la boîte, le défaut étant centré, la lettre `l` l'aligne à gauche et `r` à droite.

texte est le contenu de la boîte qui sera compilé en mode textuel.

`\mbox` est identique à la commande précédente au fait près qu'elle ne nécessite pas d'argument de longueur et de position : elle prend la longueur du texte qu'elle contient.

`\framebox` fonctionne comme `\makebox` mais insère un cadre autour de la boite.

`\fbox` est un raccourci de `\framebox` fonctionnant de la même manière que `\mbox` pour `\makebox`.

`\raisebox` est une boîte utilisée avec la syntaxe

`\raisebox{`*raise*`}[`*above*`][`*below*`]{`*texte*`}`

et qui permet de décaler verticalement l'alignement du texte. Le texte ne repose plus sur la ligne courante, mais s'en écarte vers le haut (par exemple en exposant) ou vers le bas (définissant par exemple un indice).

raise spécifie le décalage vertical du texte par rapport à la ligne.

above est un paramètre optionnel qui définit la taille que le compilateur considère comme étant au-dessus de la ligne même si effectivement ce n'est pas la réalité.

below est également optionnel, et fait de même mais en dessous de la ligne.

texte est le texte à écrire à cet endroit.

4.4.2 Les paragraphes en boîtes

Il est également permis de créer une boîte dans laquelle le texte est compilé en mode paragraphe exactement comme dans le texte normal, c'est à dire en respectant les justifications de lignes et les espacements prédéfinis. Deux commandes permettent d'introduire de telles boîtes :

`\parbox` s'utilise suivant la syntaxe

`\parbox[`*pos*`]{`*width*`}{`*texte*`}`

et crée une boîte pour laquelle

width spécifie la largeur de la boîte, inférieure à la largeur de la page.

texte est le contenu de la boite qui sera compilé en mode paragraphe.

pos permet de préciser l'alignement du texte par rapport à la ligne de texte dans laquelle la boîte se trouve. Par défaut, le texte est centré, mais si *pos*=`t`, la ligne supérieure du bloc est alignée avec la ligne courante du texte et si *pos*=`b`, la ligne inférieure est alignée avec le texte qui suivra.

Remarquons que dans cette boite, il est possible de créer des paragraphes mais ils ne seront pas indentés comme dans le reste du texte. En effet, LaTeX change la valeur du paramètre `\parindent` pour la mettre à zéro. Si on désire une indentation, il faut attribuer une nouvelle valeur à cette longueur.

`minipage` est le second moyen de créer une boite compilée en mode paragraphe. Il s'agit d'un environnement, qui comporte un argument obligatoire et un facultatif, et dont la syntaxe est :

`\begin{minipage}[`*pos*`]{`*width*`}`
texte
`\end{minipage}`

Les arguments ont la même signification que dans la boîte `\parbox`.

La différence entre les deux est l'utilisation des notes de bas de page. Dans une *minipage*, les notes de bas de page apparaissent dans le bloc de texte produit par l'environnement. De plus, dans cet environnement, on peut utiliser tous les autres environnements comme `itemize`,... ce qui n'est pas le cas dans une commande `\parbox`.

4.4.3 Les boites noires

LaTeX permet de créer des rectangles complètement noir, au moyen de la commande

`\rule[`*raise*`]{`*width*`}{`*height*`}`

où

raise spécifie le décalage vertical par rapport à la ligne : une valeur positive le place au dessus de la ligne courante alors qu'une négative le place en dessous ;

width est la largeur du rectangle ;

height définit sa hauteur.

4.4.4 Boîtes de sauvegarde

La définition de commandes (`\newcommand`, voir section 2.3) permet de gagner du temps lors de la rédaction du document en raccourcissant le texte qu'il

est nécessaire de taper. Cependant, à la compilation, ce texte doit chaque fois être recompilé ce qui peut occasionner une perte de temps spécialement si le texte est compliqué. C'est pourquoi LaTeX permet de sauvegarder des boîtes, c'est-à-dire non seulement le contenu mais également la manière de le présenter. Il le considère donc comme un seul caractère.

La création d'une commande associée à une boîte de sauvegarde respecte la syntaxe suivante :

`\newsavebox{`*name*`}`
`\savebox{`*name*`}[`*length*`]{`*texte*`}`

La première ligne permet de déclarer une variable *name* qui référera à une boîte de sauvetage. Le nom doit être un nom de commande non encore connu de LaTeX. La seconde introduit la définition même de la boîte, pour laquelle

name est le nom de la variable référant à la boîte,

length est la longueur de la boîte.

texte est le contenu de la boîte que l'on désire mémoriser.

Cette commande a un effet local uniquement c'est-à-dire que si on l'utilise dans un environnement, la valeur mémorisée n'est valable que dans celui-ci et ceux inclus dans celui-ci — ce qui permet éventuellement de redéfinir localement la boîte associée au nom *name*.

La commande `\sbox` est identique à `\savebox` au fait près qu'elle n'a pas d'argument optionnel précisant la longueur.

Après sauvegarde d'une boîte dans une variable *name*, on peut l'utiliser au moyen de la commande `\usebox{`*name*`}`.

4.5 Dessins

Un dessin créé avec LaTeX est une boîte (voir 4.4) dans laquelle on place des lignes, des flèches, des cercles et des mots. Tous les éléments d'un dessins doivent être positionnés à l'aide de leurs coordonnées. La définition d'un repère nécessite donc la définition d'une origine et d'une *unité de longueur*. Pour spécifier l'unité de longueur d'un dessin, on définit avant celui-ci la longueur `\unitlength`, par défaut fixée à 1 point.

Remarque *Il est conseillé de prendre une unité assez petite comme le millimètre (et pas le centimètre) ou le point. Ceci évite de devoir spécifier des valeurs décimales pour tous les positionnements.*

Pour créer un dessin, on utilise l'environnement `picture` qui crée une boîte dont la taille est spécifiée par le premier argument et l'unité de longueur. Les éléments de dessin sont placés dans cette boîte par rapport à une origine qui, par défaut, est le coin inférieur gauche. Si on spécifie un deuxième argument (optionnel mais placé entre parenthèses), celui-ci correspond à la coordonnée du coin inférieur gauche, changeant de ce fait l'origine de place.

La syntaxe est alors

`\begin{picture}(`*Hor, Vert*`)(`*orHor, orVert*`)`
instructions de dessin
`\end{picture}`

où

- `(`*Hor*`,` *Vert*`)` est la taille du dessin,
- `(`*orHor*`,` *orVert*`)` est optionnel et correspond à la coordonnée du coin inférieur gauche de l'image,
- *instructions de dessin* sont des instructions de positionnement d'objets dans le dessin.

Par exemple, l'instruction

`\begin{picture}(100,200)(10,20)`

crée un dessin dans une boîte dont la taille horizontale est 100 fois l'unité et la taille verticale 200 fois celle-ci. Le coin inférieur gauche est situé à la coordonnée (10,20) ce qui signifie que le coin supérieur droit est de coordonnée (110,220).

Remarque *L'argument optionnel est très intéressant lorsque l'on modifie un dessin précédemment réalisé. Il permet en effet de déplacer en une seule opération tous les éléments qui se trouvent dans le dessin.*

Il est important de noter que la taille de l'image indique simplement à LaTeX la taille de la boîte à créer et n'influence en rien la taille réelle de l'image qui est fixée par les éléments de dessin qui y sont placés. Il se peut donc que le dessin déborde de la boîte si on y positionne des éléments trop grands ou trop près du bord. Il faut dès lors faire attention à un texte qui serait placé à proximité d'un dessin : spécifier une taille de dessin trop petite risque de faire se chevaucher le dessin et le texte alors qu'une taille trop grande laisserait trop d'espace.

4.5.1 Les instructions de dessin

A l'intérieur de l'environnement `picture`, deux commandes permettent le positionnement d'objets : ce sont les instructions `\put` et `\multiput`.

Placement d'un objet : les éléments d'une image sont positionnés par l'instruction

`\put(`*Hor*`,` *Vert*`){`*objet*`}`

où `(`*Hor*`,` *Vert*`)` est la coordonnée à laquelle placer l'*objet*.

Répétition d'objets : LaTeX propose une commande permettant de répéter plusieurs fois un objet avec un décalage spécifié. Sa syntaxe est

`\multiput(`*Hor*`,` *Vert*`)(`*dHor*`,` *dVert*`){`*nbr*`}{`*objet*`}`

où

- `(`*Hor*`,` *Vert*`)` est la coordonnée de positionnement du premier objet, exactement comme pour la commande `\put`,

- `(`*dHor*`,`*dVert*`)` est le déplacement du point d'origine entre deux copies de l'objet, c'est-à-dire que le second objet est positionné en `(`*Hor*`+`*dHor*`,`*Vert*`+`*dVert*`)`, le troisième en `(`*Hor*`+2*`*dHor*`,`*Vert*`+2*`*dVert*`)` et ainsi de suite,
- *nbr* est le nombre de copies à placer,
- *objet* est l'élément de dessin à créer *nbr* fois.

4.5.2 Les objets de dessin

Les objets que l'on peut placer dans un dessin sont assez nombreux. Les principaux sont présentés ici, chaque fois accompagnés de la commande de position `\put` pour illustrer l'instruction complète.

Texte : le premier objet qu'il est possible de placer dans un dessin est un texte. La syntaxe est

`\put(`*Hor*`,`*Vert*`){`*texte*`}`

où le point de positionnement du *texte* est le coin inférieur gauche de celui-ci. Par exemple

```
\put(5,2.5){Le chien}
```

place le texte *Le chien* à la coordonnée (5,2.5), c'est-à-dire que le coin inférieur gauche du texte est à cette coordonnée.

Boîte : comme dans un texte (voir 4.4), il est possible d'ajouter une boîte au moyen des instructions `\makebox`, `\framebox` et `\dashbox`. Ces commandes ont une forme spéciale dans l'environnement de dessin car le premier argument est une coordonnée qui spécifie la largeur et la hauteur de la boîte (alors qu'en mode texte on ne peut spécifier que la largeur). La syntaxe d'insertion d'une boîte dans un dessin est donc

`\put(`*Hor*`,`*Vert*`){\makebox(`*bHor*`,`*bVert*`)[`*pos*`]{`*texte*`}}`

où
- `(`*Hor*`,`*Vert*`)` est la coordonnée à laquelle placer le coin inférieur gauche de la boîte,
- `(`*bHor*`,`*bVert*`)` est la taille de la boîte,
- *pos* est optionnel et comporte un ou deux caractères précisant la position du texte au sein de la boîte, le défaut étant centré, la lettre `l` l'aligne à gauche, `r` à droite, `t` au-dessus et `b` en-dessous,
- *texte* est le texte à placer dans la boîte.

Par exemple

```
\put(2,3.1){\makebox(2,1)[br]{Le chien}}
```

place *Le chien* dans une boîte de deux unités en longueur et d'une unité en hauteur à la position (2,3.1), le texte étant placé en bas et aligné à droite dans la boîte.

La syntaxe de la commande `\framebox` est identique et le résultat est une boîte encadrée.

Remarque *Il est tout à fait possible de placer une boîte de taille (0,0) dans une image et en voir le texte malgré cette taille. En effet, LaTeX place le texte quelque soit la taille de la boîte. Cette possibilité est très utilisée pour placer du texte centré sur un point particulier ou pour superposer du texte sur un autre.*

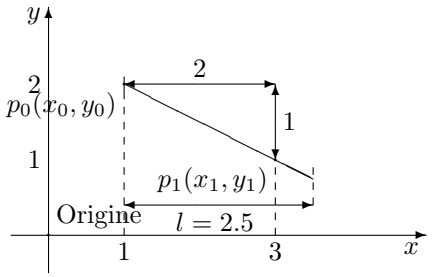
La commande `\dashbox` est similaire à la commande `\framebox` au fait près qu'elle place un rectangle avec le bord en pointillés. Pour spécifier la valeur de celui-ci, c'est un premier argument entre accolades que l'on utilise. Par exemple

```
\put(2,3.1){\dashbox{.5}(2,1)[br]{Le chien}}
```

place un rectangle avec le bord en pointillés d'une longueur de 0.5 unités.

Ligne ou trait : la manière de définir une ligne est assez complexe car il ne suffit pas de spécifier un point de départ et un point d'arrivée mais un point de départ, une inclinaison sous la forme d'une coordonnée et une longueur. Certaines restrictions sont imposées à l'inclinaison car LaTeX utilise une police de texte particulière pour les dessins, celle-ci ne contenant qu'un nombre limité de modèle de ligne.

Pour en expliquer le fonctionnement, considérons le graphique ci-dessous. Nous désirons placer une ligne qui débute au point p_0 qui est de coordonnée (x_0, y_0) (soit (1,2) dans notre exemple). L'inclinaison de la ligne est spécifiée par une coordonnée, c'est-à-dire un couple (x, y) qui ne peut pas être nulle (soit (2,-1) dans notre exemple). Considérons le point p_1 qui est de coordonnée $(x_1, y_1) = (x_0+x, y_0+y)$ (soit (3,1) dans notre exemple). Ces deux points définissent une droite (représentée en pointillés) sur laquelle se situera la ligne. Deux cas se présentent en fonction de la valeur de x. Si $x \neq 0$, alors la ligne débute au point p_0 et se trace sur la droite décrite par p_0 et p_1, en direction de p_1 et jusqu'au point d'abscisse $x_0 + l$ où l représente la longueur du trait souhaité (valeur strictement positive). Si $x = 0$, le trait est vertical et part de p_0 en direction de p_1 jusqu'au point d'ordonnée $y_0 + l$ où l est la longueur du trait (valeur strictement positive).



La syntaxe de l'instruction pour tracer une ligne est

`\put(`*Hor*`,`*Vert*`){\line(`x`,`y`){`l`}}`

où

- `(`*Hor*`,` *Vert*`)` est la coordonnée du point de départ de la ligne,
- `(`x`,`y`)` est l'inclinaison de la ligne qui ne peut être nulle ; comme il n'y a qu'un nombre limité de traits que LaTeX peut tracer, quelques restrictions sont imposées à l'inclinaison : x et y doivent être des entiers entre $+6$ et -6 et ne pas avoir de diviseur commun plus grand que 1 (c'est-à-dire qu'au lieu d'indiquer par exemple $(2, 4)$ vous devez spécifier $(1, 2)$),
- l est la longueur du tait, c'est-à-dire un nombre strictement positif qui peut contenir des décimales.

Flèche : de la même manière qu'il est possible de tracer une ligne, il est possible de placer une flèche dans un dessin. Une flèche est identique à une ligne terminée par un triangle plein. La syntaxe est

`\put(`*Hor*`,` *Vert*`){\vector(`x`,`y`){`l`}}`

où les arguments sont identiques à ceux nécessaires pour une ligne. Une restriction est cependant ajoutée à l'inclinaison car les valeurs limites sont $+4$ et -4.

Pile : une pile dans un dessin de LaTeX est un ensemble de lignes de texte. Le point de référence pour le placement de ces lignes de texte est le coin inférieur gauche de la boîte formée par ces lignes. La syntaxe est

`\put(`*Hor*`,` *Vert*`){\shortstack[`*pos*`]{`*lignes*`}}`

où
- `(`*Hor*`,` *Vert*`)` est la coordonnée de positionnement du coin inférieur gauche des lignes de texte,
- *pos* est un caractère précisant la position horizontale du texte au sein de la boîte, le défaut étant centré, la lettre `l` l'aligne à gauche et `r` à droite,
- *lignes* est une suite de lignes de caractères, deux lignes étant séparées par une double barre oblique inverse `\\`.

Cercle et disque : il est possible de tracer des cercles (c'est-à-dire le contour uniquement) et des disques (c'est-à-dire un cercle plein) au moyen de deux commandes similaires. Celles-ci ne demandent que le rayon du cercle à tracer. La syntaxe pour un cercle et un disque est donc respectivement

`\put(`*Hor*`,` *Vert*`){\circle{`*rayon*`}}`
`\put(`*Hor*`,` *Vert*`){\circle*{`*rayon*`}}`

où les arguments ne demandent aucune explication supplémentaire. Encore une fois, certaines restrictions peuvent intervenir sur le rayon en fonction des polices installées sur l'ordinateur.

Ovale et coin arrondi : dans LaTeX, un *ovale* est un rectangle dont les coins ont été remplacés par des quarts de cercle les plus grands possibles. Le point de placement d'un ovale est le centre du rectangle sur lequel il se base. Un argument optionnel à la commande d'un ovale permet de ne dessiner qu'une moitié ou un coin, ce qui permet donc de créer des quarts de cercle.

La syntaxe pour un ovale est

`\put(`*Hor*`,`*Vert*`){\oval(`x`,`y`)[`*partie*`]}`

où
- `(`*Hor*`,`*Vert*`)` est la coordonnée de positionnement du centre de l'ovale,
- *(x,y)* sont les dimensions du rectangle sur lequel est basé la construction de l'ovale,
- *partie* est un ou deux caractères précisant la portion de l'ovale à tracer, le défaut est tout, la lettre `l` ne place que la moitié gauche, `r` la droite, `t` la supérieure et `b` l'inférieure.

Pour ne tracer qu'un quart de cercle, il suffit d'utiliser une combinaison de deux lettres et de prendre pour les dimensions $x = y$.

Encadrement : il est possible de placer un encadrement autour d'un objet dans un dessin. Ce cadre est obtenu en plaçant l'objet comme argument obligatoire de la commande `\frame`. Celle-ci n'insère pas d'espace supplémentaire et prend la taille exacte de l'objet qu'elle encadre.

On écrit

`\put(`*Hor*`,`*Vert*`){\frame{`*objet*`}}`

où
- `(`*Hor*`,`*Vert*`)` définit le positionnement de l'objet,
- *objet* est l'élément de dessin à placer.

5

Formules mathématiques

Un des atouts majeurs du traitement de textes LaTeX est la facilité qu'il propose pour l'écriture des mathématiques. L'ensemble des symboles sont en effet décrits par des commandes au nom évocateur, qui permettent à l'utilisateur de garder les doigts sur le clavier et de rédiger des formules avec une étonnante rapidité.

Des formules mathématiques peuvent être placées à l'intérieur du texte ou sur un ligne séparée. Plusieurs environnements sont disponibles pour introduire de telles formules. On dit que LaTeX est en mode mathématique (à l'opposé du mode texte ou LR mode) s'il est à l'intérieur d'un environnement mathématique.

Une remarque fondamentale doit être formulée d'emblée : en mode mathématique, *le compilateur ignore les espaces introduits dans le texte.* Les espaces introduits par l'utilisateur dans le texte d'entrée sont cependant indispensables pour séparer les éléments, notamment les commandes ; de plus ils permettent d'aérer le texte d'entrée pour des raisons de clarté. L'insertion d'espaces dans le texte compilé autres que ceux imposés par le compilateur est possible à l'aide de commandes particulières (voir la section 5.2).

Remarque *Les sections qui suivent présenteront des commandes spécifiques au mode mathématique à l'aide de nombreux exemples. Ceux-ci seront présentés dans la colonne de gauche de tableaux, la colonne de droite illustrant le code propre à l'exemple.*

5.1 Environnements mathématiques

5.1.1 Equations isolées

Pour placer une formule dans un texte, sans saut de ligne ni modification de l'alignement, on utilise l'environnement `math` en écrivant les éléments mathématiques à l'intérieur des commandes

`\begin{math}...\end{math}`.

Ces dernières sont cependant rarement utilisées, puisque les raccourcis

`\(...\)` et `$...$`

produisent le même résultat. À titre d'exemple :

Voilà 100 m^3 d'eau `Voilà $100\ m^3$ d'eau`

Pour créer une formule non numérotée sur une ligne séparée, on utilise l'environnement `displaymath` en écrivant les éléments mathématiques à l'intérieur des commandes

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

Le résultat équivalent est produit par les raccourcis

`\[...\]` et `$$...$$` :

Ajoutez a au carré et b au carré pour obtenir c au carré ; mathématiquement on note

$$a^2+b^2=c^2.$$

Voilà !

```
Ajoutez $a$ au carré et $b$
au carré pour obtenir $c$ au
carré ; mathématiquement on
note $$ a^2+b^2=c^2.$$ Voilà !
```

Pour créer une formule numérotée sur une ligne séparée, on utilise l'environnement `equation` qui ne possède pas de raccourci :

Ajoutez a au carré et b au carré pour obtenir c au carré ; mathématiquement on note

$$a^2+b^2=c^2. \tag{5.1}$$

Voilà !

```
Ajoutez $a$ au carré et $b$
au carré pour obtenir $c$ au
carré ; mathématiquement on
note
\begin{equation}
a^2+b^2=c^2.
\end{equation}
Voilà !
```

Il est possible de faire référence au numéro de cette équation au moyen des instructions `\label` et `\ref` décrits à la section 4.1.1.

La mise en forme des formules diffère suivant que la formule se trouve dans un texte ou sur une ligne séparée. Cependant il est possible de forcer le style que l'on désire. Le style correspondant à une formule dans un texte est `\textstyle` et celui d'une formule d'aspect mathématique `\displaystyle`. Il suffit, pour forcer un style, de précéder la formule par la commande adéquate. La différence entre les deux est clairement mise en évidence par l'écriture d'une fraction, simplement dans le premier exemple et en imposant la mise en forme mathématique dans le second, à l'aide de la commande `\displaystyle` :

La citerne est vide aux $\frac{2}{3}$

```
La citerne est vide aux
$\frac{2}{3}$
```

La citerne est vide aux $\displaystyle\frac{2}{3}$

```
La citerne est vide aux
$\displaystyle\frac{2}{3}$
```

5.1.2 Systèmes d'équations

Pour écrire des systèmes d'équations numérotées, on utilise l'environnement `eqnarray` : cet environnement engendre l'écriture de chaque équation à l'intérieur d'un tableau, les équations étant alignées sur le bord droit du tableau. Cet environnement est également utilisé pour écrire une équation trop longue sur plusieurs lignes : en effet dans ce cas LaTeX ne prend pas l'initiative de la découper.

Il est possible de réaliser ces systèmes d'équations sans numérotation au moyen de l'environnement `\eqnarray*`, qui vérifie les mêmes propriétés que le précédent.

Au sein de l'un de ces deux environnements, le passage à la ligne est produit par la commande `\\`. Pour supprimer la numérotation d'une ligne déterminée, on utilise la commande `\nonumber` avant d'imposer le passage à la ligne suivante. Il est encore possible de forcer l'alignement des lignes en imposant dans chaque ligne un repère : ces repères sont entourés par les symboles `&...&` et sont alors alignés l'un au-dessus de l'autre. Enfin, lorsqu'une équation est rédigée sur deux lignes, il peut être commode d'aligner la première ligne totalement à gauche au moyen de la commande `\lefteqn{...}` ; les symboles `& &` débuteront alors chacune des autres lignes.

Exemples :

– Système de relations numérotées :

$$a^2 + b^2 = c^2 \quad (5.2)$$
$$\sqrt{a} = \frac{d}{2} \quad (5.3)$$

```
\begin{eqnarray}
a^2+b^2=c^2\\
\sqrt{a}=\frac{d}{2}
\end{eqnarray}
```

– Equation rédigée sur deux lignes avec alignement de la première ligne à gauche :

$$a^2 + b^2 + c^2 + \\ +d^2 + e^2 + f^2 = \frac{z}{2} \quad (5.4)$$

```
\begin{eqnarray}
\lefteqn{a^2+b^2+c^2+}
\nonumber \\
& & +d^2+e^2+f^2
=\frac{z}{2}
\end{eqnarray}
```

– Alignement sur un symbole de référence :

$$\begin{aligned} f(x) &= \cos x \\ f'(x) &= -\sin x \end{aligned}$$

```
\begin{eqnarray*}
f(x) & = & \cos x \\
f'(x) & = & -\sin x
\end{eqnarray*}
```

5.1.3 Théorèmes, propositions, lemmes

L'écriture des lemmes, propositions mathématiques ou des théorèmes nécessite l'emploi d'environnements particuliers préalablement définis dans le préambule (voir 2.3). Ce préambule contiendra des déclarations du type

`\newtheorem{`*nom*`}{`*titre*`}`,

définissant des environnements mathématiques appelés `nom` qui, lorsqu'ils sont utilisés, appellent l'affichage du titre `titre`, de la numérotation des éléments de la même structure ; le texte à l'intérieur de l'environnement étant imprimé en italique.
Ex. `\newtheorem{thrm}{Théorème}`.

L'utilisation de la structure dans le texte comportera les lignes suivantes :

```
\begin{thrm}
Texte du théorème.
\end{thrm}
```

et la compilation de ces lignes produira :

Théorème 1 *Texte du théorème.*

Il est possible de lier la numérotation des structures à l'aide de l'argument optionnel entre crochets (ex. : [`chapter`]). Il est également possible de lier la numérotation de différentes structures en ajoutant entre les deux arguments de la définition de la seconde structure le nom donné à la première. Par exemple, combinant les deux options pour lier la numérotation des propositions à celle des théorèmes et lier les deux à la numérotation des chapitres, on écrira dans le préambule :

```
\newtheorem{thrm}{Théorème}[chapter]
\newtheorem{prop}[thrm]{Proposition}
```

Il est encore possible de spécifier le nom d'une structure, en indiquant celui-ci en argument entre crochets lors de l'appel de la structure.
Ex.

```
\begin{thrm}[Pythagore]
Texte du Théorème de Pythagore
\end{thrm}
```

produira :

Théorème 2 (Pythagore) *Texte du Théorème de Pythagore*

5.2 Espaces en mathématique

LaTeX ignore complètement les espaces dans les formules mathématiques. Plus exactement, il adapte lui-même les espaces. Dans certains cas, il est utile d'augmenter l'espace et dans d'autres de le diminuer. Pour cela, LaTeX offre cinq commandes :

`\,` petit espace (demi-caractère)	`\:` moyen espace
`\;` large espace	`\ ` espace d'un caractère
`\!` petit espace négatif	

Les commandes `\quad` et `\qquad` permettent de produire des espacements plus grands. Comparons :

ab (espace normal)	`$a b$ (espace normal)\\`
ab	`$a \! b$\\`
$a\,b$	`$a \, b$\\`
$a\:b$	`$a \: b$\\`
$a\;b$	`$a \; b$\\`
$a\ b$	`$a \ b$\\`
$a\quad b$	`$a \quad b$\\`
$a\qquad b$	`$a \qquad b$`

Seule `\,` peut aussi être utilisée sans le mode mathématique. Ces commandes peuvent être utilisées, mais il ne faut pas en abuser car LaTeX adapte généralement les espacements mathématiques de manière appropriée. Pour citer les utilisations les plus courantes, il faut ajouter un petit espace avant le dx dans une intégrale, juste après une fraction et une racine ; un espace négatif est utilisé pour rapprocher les signes d'intégration d'une intégrale double ainsi qu'après la barre de fraction / d'un quotient.

5.3 Polices de caractères mathématiques

L'écriture de lettres dans un environnement mathématique impose pour celles-ci la police italique. Il est cependant possible d'utiliser d'autres aspects de caractères (certains sont réservés aux lettres majuscules) :

caractères romans (normaux) :	Ba	`$\mathrm{Ba}$`
caractères romans gras :	**Ba**	`$\mathbf{Ba}$`
caractères d'imprimerie :	Ba	`$\mathtt{Ba}$`
majuscules calligraphiées :	$\mathcal{B}$	`$\mathcal{B}$`
majuscules doublées :	$\mathbb{B}$	`$\mathbb{B}$`
majuscules droites :	B	`$\mathsf{B}$`

La commande `\mathrm{`*texte*`}` permet d'insérer un texte en mode mathématique pour qu'il soit affiché dans la police principale du document. Cependant elle est compilée en mode mathématique et ne considère donc pas les espaces. Pour insérer plusieurs mots et respecter une compilation en mode textuel, il faut utiliser la commande `\textrm{`*texte*`}`[1] La différence est clairement marquée par l'exemple suivant :

$f(x) = 1 \quad \text{pourtout} x > 0$
$f(x) = 1 \quad \text{pour tout } x > 0$

est produit par

```
$f(x)=1 \quad \mathrm{pour tout } x \leq 0$\\
$f(x)=1 \quad \textrm{pour tout } x \leq 0$
```

La commande `\mathbf` permet d'obtenir des symboles romans (normaux) gras ; pour obtenir des symboles gras en mode mathématique qui respectent la mise en forme mathématique (italique), il faut basculer en version grasse par la commande
`\mathversion{bold}` *avant d'entrer en mode mathématique* et revenir en version normale par `\mathversion{normal}` *à nouveau dans le mode texte* pour retrouver les caractères mathématiques habituels dans le mode mathématique suivant :

Avant d'entrer en mode mathématique
$\boldsymbol{\mu, M}$
et après être revenu en mode textuel,
μ, M

```
Avant d'entrer en mode
mathématique\\
\mathversion{bold} $\mu,M$\\
et après être revenu
en mode textuel,\\
\mathversion{normal}
$\mu, M$
```

Le module `amsmath` fournit en plus les commandes `\boldsymbol{...}` pour l'obtention de caractères gras et `\pmb{...}` pour un gras épais.

Enfin, en plus de l'aspect de la police, il est possible de modifier la taille des caractères en mode mathématique, au moyen d'une des quatre commandes

[1] Le module `amsmath` fournit une commande qui possède les mêmes propriétés que `\textrm`.

illustrées ci-dessous :

$Abc\ 123$	`$\displaystyle{Abc\ 123}$`
$Abc\ 123$	`$\textstyle{Abc\ 123}$`
$Abc\ 123$	`$\scriptstyle{Abc\ 123}$`
$Abc\ 123$	`$\scriptscriptstyle{Abc\ 123}$`

5.4 Éléments d'une formule mathématique

Envisageons les différentes structures les plus couramment rencontrées dans les environnements mathématiques.

Les indices et exposants

Les indices et exposants sont respectivement produits par `^` et `_`; s'il y a plus d'un caractère sur lequel s'applique la commande, l'ensemble des caractères à écrire en indice ou en exposant est regroupé à l'intérieur d'accolades :

TAB. 5.1 – Indices et exposants

x_i	`x_i`	x^2	`x^2`
x_i^2	`x_i^2`	x_{n_i}	`x_{n_i}`

Les fractions

On génère une fraction par la commande `\frac{`*num*`}{`*den*`}` :

$\frac{4x+5}{2x-3}$	`\frac{4x+5}{2x-3}`

Les racines

Elles sont générées par la commande `\sqrt[n]{...}` où l'argument n définit la racine $n^{\text{ième}}$:

$\sqrt{7x^2+5}$	`\sqrt{7x^2+5}`
$\sqrt[3]{7x^2+5}$	`\sqrt[3]{7x^2+5}`

Les trois points

Il existe plusieurs manières et plusieurs dispositions pour écrire des points de suspensions mathématiques :

La commande `\ldots` peut s'utiliser également en mode texte. La distinction entre `\ldots` et `\cdots` est la hauteur sur la ligne. Par exemple on utilise `\cdots` dans $x_1 + \cdots + x_n$ tandis que l'on utilise `\ldots` dans $x_1, \ldots, x_n$.

Tab. 5.2 – Points de suspension mathématiques

$\ldots$	`\ldots`	$\cdots$	`\cdots`
$\vdots$	`\vdots`	$\ddots$	`\ddots`

Les accents

En mode mathématiques, on retrouve une partie des commandes permettant d'appliquer des accents sur des caractères, plus d'autres types propres au langage mathématique :

Tab. 5.3 – Les accents mathématiques

$\hat{a}$	\hat{a}	$\check{a}$	\check{a}	$\breve{a}$	\breve{a}
$\acute{a}$	\acute{a}	$\grave{a}$	\grave{a}	$\tilde{a}$	\tilde{a}
$\bar{a}$	\bar{a}	$\vec{a}$	\vec{a}	$\dot{a}$	\dot{a}
$\ddot{a}$	\ddot{a}				

Lorsqu'il y a plusieurs caractères à disposer sous (ou au dessus de) l'élément d'accentuation, les commandes diffèrent quelque peu pour devenir :

Tab. 5.4 – Les accents mathématiques étendus

$\widetilde{abc}$	\widetilde{abc}	$\overbrace{abc}$	\overbrace{abc}
$\widehat{abc}$	\widehat{abc}	$\underbrace{abc}$	\underbrace{abc}
$\overleftarrow{abc}$	\overleftarrow{abc}	$\overline{abc}$	\overline{abc}
$\overrightarrow{abc}$	\overrightarrow{abc}	$\underline{abc}$	\underline{abc}

La superposition de plusieurs caractères s'effectue au moyen de la commande `\stackrel` ainsi le symbole de définition $\stackrel{\triangle}{=}$ est obtenu par `\stackrel{\triangle}{=}`.

Matrices et tableaux mathématiques

Les matrices et systèmes d'équations reliées par une accolade sont rédigés sous forme de tableaux, au moyen de l'environnement `array` (voir section 4.3.2). À titre d'exemple :

$$\mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ x_{m1} & \dots & \dots & x_{mn} \end{pmatrix}$$

```
$$\mathbf{X}=\left(
\begin{array}{ccc}
x_{11} & x_{12} & \ldots & x_{1n}\\
x_{21} &        &        &       \\
\vdots &        & \ddots &       \\
x_{m1} &        & \ldots & x_{mn}
\end{array} \right)$$
```

Les délimiteurs

On reprend sous cette appellation l'ensemble des parenthèses, crochets et accolades à l'intérieur desquels sont écrites des expressions mathématiques. S'ils sont exprimés par une commande simple, ces délimiteurs ont une taille fixe ; mais ils peuvent avoir une taille variable, qui s'adapte à ce qui se trouve à l'intérieur, et ajoutant respectivement `\left` et `\right` devant la commande du délimiteur. Ces deux commandes apparaissent toujours ensemble, toutefois lorsqu'on ne souhaite voir apparaître qu'un seul délimiteur, on peut obtenir un second délimiteur invisible simplement en plaçant un point (.) à la place de celui-ci (ex. : `\left[ ...\right.`). Dans certains cas il est nécessaire d'indiquer la taille exacte de ces délimiteurs par une commande qui les agrandit : ces commandes précèdent celles des délimiteurs et sont, de la plus petite à la plus grande, `\big`, `\Big`, `\bigg` et `\Bigg`. L'exemple suivant illustre l'agrandissement qu'ils imposent aux délimiteurs :

$$\big(\Big(\bigg(\Bigg(\quad \}\Bigg)\bigg)\Big)\big)$$

```
$(\big(\Big(\bigg(\Bigg( \quad
\}\big\}\Big\}\bigg\}\Bigg\}$
```

Les symboles à taille variable

C'est une propriété qu'ont les symboles tels que sommes et intégrales, de changer de taille suivant les conditions d'utilisation : ainsi un symbole d'intégrale sera plus petit lorsqu'il sera employé au numérateur d'une fraction. Ces symboles sont repris dans le tableau 5.6.

Ces symboles peuvent posséder des indices et exposants comme pour le signe d'intégration où cela représente les bornes. Par exemple l'intégrale de zéro à l'infini de la fonction f s'écrit :

$$\int_0^\infty f(x)\,dx$$

```
$\int_{0}^{\infty}f(x)\,dx$
```

Tab. 5.5 – Délimiteurs

(	(	)	)
⌈	[ou \lbrack	⌉	] ou \rbrack
{	\{ ou \lbrace	}	\} ou \rbrace
⌊	\lfloor	⌋	\rfloor
⌈	\lceil	⌉	\rceil
⟨	\langle	⟩	\rangle
∣	| ou \vert	∥	\\| ou \Vert
↑	\uparrow	↓	\downarrow
↕	\updownarrow	⇑	\Uparrow
⇓	\Downarrow	⇕	\Updownarrow

Tab. 5.6 – Symboles à taille variable

$\sum$	\sum	$\prod$	\prod	$\coprod$	\coprod
$\int$	\int	$\oint$	\oint	$\bigcap$	\bigcap
$\bigcup$	\bigcup	$\bigsqcup$	\bigsqcup	$\bigvee$	\bigvee
$\bigwedge$	\bigwedge	$\bigodot$	\bigodot	$\bigotimes$	\bigotimes
$\bigoplus$	\bigoplus	$\biguplus$	\biguplus		

Pour superposer des indices, il est possible à l'aide du module `amsmath` (compris dans le module `amslatex`) d'utiliser la commande `\substack{...}` ou l'environnement `subarray`, qui produisent respectivement des indices superposés centrés ou alignés ; les lignes d'indices sont séparées par la commande habituelle `\\`.

$$\sum_{\substack{0\leq i\leq n\\ 1\leq j\leq m}} P_{ij} = \sum_{\substack{r\in I\\ 0\leq s\leq 10}} Q_{rs}$$

```
$$\sum_{\substack{
       0\leq i\leq n \\
       1\leq j\leq m}} P_{ij} =
\sum_{\begin{subarray}{l}
      r\in I \\
      0\leq s\leq 10
      \end{subarray}} Q_{rs}$$
```

Les fonctions mathématiques

Les fonctions mathématiques doivent apparaître sous un format textuel (c'est-à-dire pas en italique) : il existe donc des commandes pour générer les plus usuelles :

La fonction *modulo* admet deux commandes possibles : `\bmod` pour l'opérateur binaire, et `\pmod` pour l'opérateur unaire :

$a \bmod b$
$x \equiv a \pmod b$

```
$a \bmod b$\\
$x \equiv a \pmod b$
```

Tab. 5.7 – Fonctions mathématiques

arccos	\arccos	arcsin	\arcsin	arctan	\arctan
arg	\arg	cos	\cos	cosh	\cosh
cot	\cot	coth	\coth	csc	\csc
deg	\deg	det	\det	dim	\dim
exp	\exp	gcd	\gcd	hom	\hom
inf	\inf	ker	\ker	lg	\lg
lim	\lim	lim inf	\liminf	lim sup	\limsup
ln	\ln	log	\log	max	\max
min	\min	Pr	\Pr	sec	\sec
sin	\sin	sinh	\sinh	sup	\sup
tan	\tan	tanh	\tanh		

Les lettres grecques

D'un usage fréquent en mathématique, elles sont produites par les commandes suivantes :

Tab. 5.8 – Lettres grecques

α	\alpha	β	\beta	γ	\gamma
δ	\delta	ϵ	\epsilon	ε	\varepsilon
ζ	\zeta	η	\eta	θ	\theta
ϑ	\vartheta	ι	\iota	κ	\kappa
λ	\lambda	μ	\mu	ν	\nu
ξ	\xi	π	\pi	ϖ	\varpi
ρ	\rho	ϱ	\varrho	σ	\sigma
ς	\varsigma	τ	\tau	υ	\upsilon
ϕ	\phi	φ	\varphi	χ	\chi
ψ	\psi	ω	\omega		
Γ	\Gamma	Δ	\Delta	Θ	\Theta
Λ	\Lambda	Ξ	\Xi	Π	\Pi
Σ	\Sigma	Υ	\Upsilon	Φ	\Phi
Ψ	\Psi	Ω	\Omega		

Les relations binaires

On distinguera les symboles et les opérateurs de relations dans deux tableaux distincts ; précisons d'emblée que la négation des symboles s'obtient en ajoutant `\not` devant l'expression de ceux-ci. Par exemple, $\not\leq$ s'obtient en tapant `\not\leq` :

TAB. 5.9 – Symboles de relations binaires

$\leq$	\leq	$\geq$	\geq	$\prec$	\prec	$\succ$	\succ
$\preceq$	\preceq	$\succeq$	\succeq	$\ll$	\ll	$\gg$	\gg
$\subset$	\subset	$\supset$	\supset	$\subseteq$	\subseteq	$\supseteq$	\supseteq
$\sqsubset$	\sqsubset	$\sqsupset$	\sqsupset	$\sqsubseteq$	\sqsubseteq	$\sqsupseteq$	\sqsupseteq
$\in$	\in		\ni	$\vdash$	\vdash	$\dashv$	\dashv
$\equiv$	\equiv	$\sim$	\sim	$\simeq$	\simeq	$\asymp$	\asymp
$\approx$	\approx	$\cong$	\cong	$\neq$	\neq	$\doteq$	\doteq
$\propto$	\propto	$\models$	\models	$\perp$	\perp	$\mid$	\mid
$\parallel$	\parallel	$\bowtie$	\bowtie	$\bowtie$	\Join	$\smile$	\smile
$\frown$	\frown						

TAB. 5.10 – Opérateurs binaires

$\pm$	\pm	$\mp$	\mp	$*$	\ast
$\times$	\times	$\div$	\div	$\star$	\star
$\circ$	\circ	$\bullet$	\bullet		
$\cdot$	\cdot				
$\cap$	\cap	$\cup$	\cup	$\uplus$	\uplus
$\sqcap$	\sqcap	$\sqcup$	\sqcup	$\setminus$	\setminus
$\vee$	\vee	$\wedge$	\wedge	$\wr$	\wr
$\triangleright$	\triangleright	$\triangleleft$	\triangleleft	$\diamond$	\diamond
$\bigtriangledown$	\bigtriangledown	$\bigtriangleup$	\bigtriangleup	$\amalg$	\amalg
$\lhd$	\lhd	$\rhd$	\rhd	$\oplus$	\oplus
$\unlhd$	\unlhd	$\unrhd$	\unrhd	$\ominus$	\ominus
$\otimes$	\otimes	$\oslash$	\oslash	$\odot$	\odot
$\bigcirc$	\bigcirc	$\dagger$	\dag ou \dagger[a]	$\ddagger$	\ddag ou \ddagger[a]

[a] Disponibles également en mode textuel

Les symboles divers

On classera ici les symboles que les mathématiciens utilisent généralement pour décrire des variables ou des opérations, ou simplement qu'ils utilisent à titre de raccourcis conventionnels :

Tab. 5.11 – Symboles divers

$\aleph$	\aleph	$\hbar$	\hbar	$\imath$	\imath
$\jmath$	\jmath	ℓ	\ell	$\wp$	\wp
$\Re$	\Re	$\Im$	\Im	$\mho$	\mho
$\prime$	\prime	$\emptyset$	\emptyset	∇	\nabla
$\surd$	\surd	$\angle$	\angle	$\top$	\top
$\bot$	\bot	$\backslash$	\backslash	$\|$	\\|
$\forall$	\forall	$\exists$	\exists	$\neg$	\neg
$\flat$	\flat	$\natural$	\natural	$\sharp$	\sharp
∂	\partial	∞	\infty	$\Box$	\Box[b]
$\Diamond$	\Diamond[b]	$\triangle$	\triangle	©	\copyright[c]
¶	\P[c]	§	\S[c]	$\clubsuit$	\clubsuit
$\diamondsuit$	\diamondsuit	$\heartsuit$	\heartsuit	$\spadesuit$	\spadesuit

[b] Disponibles avec le module `latexsym`
[c] Peuvent être également utilisés en mode textuel

Les flèches

Plusieurs sortes de flèches existent et sont produites par les commandes suivantes :

TAB. 5.12 – Flèches

$\leftarrow$	\leftarrow ou \gets	$\rightarrow$	\rightarrow ou \to
$\longleftarrow$	\longleftarrow	$\longrightarrow$	\longrightarrow
$\Leftarrow$	\Leftarrow	$\Rightarrow$	\Rightarrow
$\Longleftarrow$	\Longleftarrow	$\Longrightarrow$	\Longrightarrow
$\leftrightarrow$	\leftrightarrow	$\longleftrightarrow$	\longleftrightarrow
$\Leftrightarrow$	\Leftrightarrow	$\Longleftrightarrow$	\Longleftrightarrow
$\mapsto$	\mapsto	$\longmapsto$	\longmapsto
$\hookleftarrow$	\hookleftarrow	$\hookrightarrow$	\hookrightarrow
$\leftharpoonup$	\leftharpoonup	$\rightharpoonup$	\rightharpoonup
$\leftharpoondown$	\leftharpoondown	$\rightharpoondown$	\rightharpoondown
$\rightleftharpoons$	\rightleftharpoons	$\leadsto$	\leadsto[d]
$\uparrow$	\uparrow	$\Uparrow$	\Uparrow
$\downarrow$	\downarrow	$\Downarrow$	\Downarrow
$\updownarrow$	\updownarrow	$\Updownarrow$	\Updownarrow
$\nearrow$	\nearrow	$\searrow$	\searrow
$\swarrow$	\swarrow	$\nwarrow$	\nwarrow
$\iff$	\iff (plus d'espace)		

[d] Disponible avec le module `latexsym`

Le module `amsmath`

Ce module chargé dans le préambule fournit des listes supplémentaires de symboles pour l'écriture des mathématiques ; les tableaux suivants présentent ces symboles et les commandes qui les produisent.

TAB. 5.13 – Délimiteurs fournis par `amsmath`

$\ulcorner$	\ulcorner	$\urcorner$	\urcorner
$\llcorner$	\llcorner	$\lrcorner$	\lrcorner

Tab. 5.14 – Relations binaires fournies par `amsmath`

$\lessdot$	`\lessdot`	$\gtrdot$	`\gtrdot`
$\leqslant$	`\leqslant`	$\geqslant$	`\geqslant`
$\risingdotseq$	`\risingdotseq`	$\fallingdotseq$	`\fallingdotseq`
$\eqslantless$	`\eqslantless`	$\eqslantgtr$	`\eqslantgtr`
$\Doteq$	`\Doteq`	$\trianglelefteq$	`\trianglelefteq`
$\eqcirc$	`\eqcirc`	$\circeq$	`\circeq`
$\bumpeq$	`\bumpeq`	$\Bumpeq$	`\Bumpeq`
$\leqq$	`\leqq`	$\geqq$	`\geqq`
$\lll$	`\lll`	$\ggg$	`\ggg`
$\lesssim$	`\lesssim`	$\gtrsim$	`\gtrsim`
$\lessapprox$	`\lessapprox`	$\gtrapprox$	`\gtrapprox`
$\lessgtr$	`\lessgtr`	$\gtrless$	`\gtrless`
$\lesseqgtr$	`\lesseqgtr`	$\gtreqless$	`\gtreqless`
$\lesseqqgtr$	`\lesseqqgtr`	$\gtreqqless$	`\gtreqqless`
$\thicksim$	`\thicksim`	$\thickapprox$	`\thickapprox`
$\preccurlyeq$	`\preccurlyeq`	$\succcurlyeq$	`\succcurlyeq`
$\curlyeqprec$	`\curlyeqprec`	$\curlyeqsucc$	`\curlyeqsucc`
$\backsim$	`\backsim`	$\backsimeq$	`\backsimeq`
$\precsim$	`\precsim`	$\succsim$	`\succsim`
$\precapprox$	`\precapprox`	$\succapprox$	`\succapprox`
$\approxeq$	`\approxeq`	$\vDash$	`\vDash`
$\Vdash$	`\Vdash`	$\Vvdash$	`\Vvdash`
$\subseteqq$	`\subseteqq`	$\supseteqq$	`\supseteqq`
$\Subset$	`\Subset`	$\Supset$	`\Supset`
$\sqsubset$	`\sqsubset`	$\sqsupset$	`\sqsupset`
$\backepsilon$	`\backepsion`	$\varpropto$	`\varpropto`
$\therefore$	`\therefore`	$\because$	`\because`
$\shortmid$	`\shortmid`	$\shortparallel$	`\shortparallel`
$\between$	`\between`	$\pitchfork$	`\pitchfork`
$\smallsmile$	`\smallsmile`	$\smallfrown$	`\smallfrown`
$\vartriangleleft$	`\vartriangleleft`	$\vartriangleright$	`\vartriangleright`
$\trianglelefteq$	`\trianglelefteq`	$\trianglerighteq$	`\trianglerighteq`
$\blacktriangleleft$	`\blacktriangleleft`	$\blacktriangleright$	`\blacktriangleright`

Tab. 5.15 – Opérateurs binaires fournis par `amsmath`

$\dot{+}$	`\dotplus`	$\centerdot$	`\centerdot`
$\ltimes$	`\ltimes`	$\rtimes$	`\rtimes`
$\divideontimes$	`\divideontimes`	$\smallsetminus$	`\smallsetminus`
$\Cup$	`\Cup`	$\Cap$	`\Cap`
$\veebar$	`\veebar`	$\barwedge$	`\barwedge`
$\doublebarwedge$	`\doublebarwedge`	$\circleddash$	`\circleddash`
$\circledcirc$	`\circledcirc`	$\circledast$	`\circledast`
$\boxplus$	`\boxplus`	$\boxminus$	`\boxminus`
$\boxtimes$	`\boxtimes`	$\boxdot$	`\boxdot`
$\leftthreetimes$	`\leftthreetimes`	$\rightthreetimes$	`\rightthreetimes`
$\curlyvee$	`\curlyvee`	$\curlywedge$	`\curlywedge`
$\intercal$	`\intercal`		

Tab. 5.16 – Flèches fournies par `amsmath`

$\dashleftarrow$	`\dashleftarrow`	$\dashrightarrow$	`\dashrightarrow`
$\leftleftarrows$	`\leftleftarrows`	$\rightrightarrows$	`\rightrightarrows`
$\upuparrows$	`\upuparrows`	$\downdownarrows$	`\downdownarrows`
$\leftrightarrows$	`\leftrightarrows`	$\rightleftarrows$	`\rightleftarrows`
$\Lleftarrow$	`\Lleftarrow`	$\Rrightarrow$	`\Rrightarrow`
$\twoheadleftarrow$	`\twoheadleftarrow`	$\twoheadrightarrow$	`\twoheadrightarrow`
$\leftarrowtail$	`\leftarrowtail`	$\rightarrowtail$	`\rightarrowtail`
$\upharpoonleft$	`\upharpoonleft`	$\upharpoonright$	`\upharpoonright`
$\downharpoonleft$	`\downharpoonleft`	$\downharpoonright$	`\downharpoonright`
$\leftrightharpoons$	`\leftrightharpoons`	$\rightleftharpoons$	`\rightleftharpoons`
$\Lsh$	`\Lsh`	$\Rsh$	`\Rsh`
$\looparrowleft$	`\looparrowleft`	$\looparrowright$	`\looparrowright`
$\curvearrowleft$	`\curvearrowleft`	$\curvearrowright$	`\curvearrowright`
$\circlearrowleft$	`\circlearrowleft`	$\circlearrowright$	`\circlearrowright`
$\leftrightsquigarrow$	`\leftrightsquigarrow`	$\rightsquigarrow$	`\rightsquigarrow`
$\multimap$	`\multimap`		

Tab. 5.17 – Caractères hébreux fournis par `amsmath`

$\digamma$	`\digamma`	$\varkappa$	`\varkappa`
$\beth$	`\beth`	$\daleth$	`\daleth`
$\gimel$	`\gimel`		

TAB. 5.18 – Négations des opérateurs binaires fournis par `amsmath`

$\not<$	`\nless`	$\not>$	`\ngtr`
$\not\leq$	`\nleq`	$\not\geq$	`\ngeq`
$\lneq$	`\lneq`	$\gneq$	`\gneq`
$\not\leqq$	`\nleqq`	$\not\geqq$	`\ngeqq`
$\lneqq$	`\lneqq`	$\lneqq$	`\lneqq`
$\varsubsetneqq$	`\varsubsetneqq`	$\varsupsetneqq$	`\varsupsetneqq`
$\not\subseteqq$	`\nsubseteqq`	$\not\supseteqq$	`\nsupseteqq`
$\not\leqslant$	`\nleqslant`	$\not\geqslant$	`\ngeqslant`
$\nmid$	`\nmid`	$\nparallel$	`\nparallel`
$\nshortmid$	`\nshortmid`	$\nshortparallel$	`\nshortparallel`
$\lvertneqq$	`\lvertneqq`	$\gvertneqq$	`\gvertneqq`
$\lnapprox$	`\lnapprox`	$\gnapprox$	`\gnapprox`
$\lnsim$	`\lnsim`	$\gnsim$	`\gnsim`
$\not\sim$	`\nsim`	$\not\cong$	`\ncong`
$\not\prec$	`\nprec`	$\not\succ$	`\nsucc`
$\not\preceq$	`\npreceq`	$\not\succeq$	`\nsucceq`
$\precneqq$	`\precneqq`	$\succneqq$	`\succneqq`
$\precnsim$	`\precnsim`	$\succnsim$	`\succnsim`
$\precnapprox$	`\precnapprox`	$\succnapprox$	`\succnapprox`
$\nvdash$	`\nvdash`	$\nvDash$	`\nvDash`
$\nVdash$	`\nVdash`	$\nVDash$	`\nVDash`
$\subsetneq$	`\subsetneq`	$\supsetneq$	`\supsetneq`
$\varsubsetneq$	`\varsubsetneq`	$\varsupsetneq$	`\varsupsetneq`
$\not\subseteq$	`\subsetneqq`	$\not\supseteq$	`\supsetneqq`
$\nsubseteq$	`\nsubseteq`	$\nsupseteq$	`\nsupseteq`
$\not\vartriangleleft$	`\ntriangleleft`	$\not\vartriangleright$	`\ntriangleright`
$\not\trianglelefteq$	`\ntrianglelefteq`	$\not\trianglerighteq$	`\ntrianglerighteq`
$\nleftarrow$	`\nleftarrow`	$\nrightarrow$	`\nrightarrow`
$\nLeftarrow$	`\nLeftarrow`	$\nRightarrow$	`\nRightarrow`
$\nleftrightarrow$	`\nleftrightarrow`	$\nLeftrightarrow$	`\nLeftrightarrow`

TAB. 5.19 – Symboles divers fournis par `amsmath`

$\hbar$	`\hslash`	$\Bbbk$	`\Bbbk`
Ⓢ	`\circledS`	$\complement$	`\complement`
$\square$	`\square`	$\blacksquare$	`\blacksquare`
$\vartriangle$	`\vartriangle`	$\blacktriangle$	`\blacktriangle`
$\triangledown$	`\triangledown`	$\blacktriangledown$	`\blacktriangledown`
$\Game$	`\Game`	$\bigstar$	`\bigstar`
$\lozenge$	`\lozenge`	$\blacklozenge$	`\blacklozenge`
$\angle$	`\angle`	$\measuredangle$	`\measuredangle`
$\sphericalangle$	`\sphericalangle`	$\backprime$	`\backprime`
$\diagup$	`\diagup`	$\diagdown$	`\diagdown`
$\eth$	`\eth`	$\varnothing$	`\varnothing`
$\nexists$	`\nexists`	$\Finv$	`\Finv`

6

Index et bibliographie

Lors de la compilation, LaTeX mémorise un certain nombre d'éléments qu'il peut ensuite regrouper pour dresser automatiquement des listes ou des index. La forme la plus courante de récapitulatif est la table des matières mais il peut également générer la liste des tableaux et celle des figures ainsi qu'un index.

6.1 Table des matières

La création d'une table des matières est très simple et entièrement automatique. La commande

```
\tableofcontents
```

insère à l'endroit où elle est placée, une table des matières qui comprend, par défaut, les numéros, noms et pages des chapitres, sections et sous-sections du document.

La table des matières est précédée d'un titre. Suivant le style du document, elle provoque ou non le passage à la page avant et après.

Comme pour la numérotation des parties d'un document (voir section 3.1.2), il est possible de changer la profondeur des éléments insérés dans la table des matières en modifiant la valeur du paramètre `tocdepth` (qui vaut donc 2 par défaut). Ainsi, si on souhaite voir également apparaître les sous-sous-sections, on demandera une profondeur de 3 pour la table des matières en plaçant la ligne suivante dans le préambule du document :

```
\setcounter{tocdepth}{3}
```

Remarque *Afin d'obtenir la table des matières correcte, il est impératif de compiler deux fois le document, voire trois fois dans le cas d'une longue table des matières, car dans ce cas celle-ci s'étend sur plus d'une page et force le décalage de la numérotation du texte. LaTeX doit en effet mettre à jour le fichier dans lequel il mémorise les informations avant la construction de la table elle-même.*

6.2 Listes des tableaux et des figures

Les tableaux et les figures sont des éléments flottants de LaTeX (ce qui signifie qu'ils ne seront pas automatiquement insérés dans le texte à l'endroit où l'utilisateur les insère dans le code) auxquels il est possible d'associer un titre et une numérotation, au moyen de la commande `\caption`.

Les deux commandes pour l'obtention des listes récapitulatives sont

`\listoftables` et `\listoffigures`

pour les tableaux et les figures respectivement. La table des figures (des tableaux) reprend les titres des figures (des tableaux) qui sont présents dans le document avec l'indication de la page sur laquelle ils se trouvent. Cette liste est précédée d'un titre (dépendant du type de la liste) et comme pour la table des matières, la présentation dépend du style du document.

Remarque *L'instruction* `\caption` *peut avoir un argument optionnel, qui sera alors inséré dans la liste récapitulative correspondante à la place du titre de la structure désignée (voir section 4.3).*

6.3 Index

En général, l'index n'est pas aussi simple à créer que la table des matières ou les listes des tableaux ou des figures. En effet, les titres des parties ou des éléments flottants sont clairement indiqués dans le document de telle sorte que LaTeX peut aisément les extraire pour en faire un récapitulatif. Pour l'index, il est inconcevable d'y inclure tous les mots du document et LaTeX ne peut donc pas décider sans aide des mots qui doivent y apparaître. On ne peut pas non plus préciser une liste de mots dont LaTeX devrait rechercher les occurrences, car il faudrait alors en préciser la casse (majuscule ou minuscule, avec ou sans accents, etc...)

L'utilisateur doit donc inclure des instructions dans son document afin que le compilateur les mémorise pour en faire un index. Ces instructions doivent être introduite en suivant une procédure définie :

1. Il faut placer la commande

   `\makeindex`

dans le préambule du document[1]. Celle-ci a pour effet de créer un nouveau fichier *name.idx* qui contiendra des instructions associant le mot devant apparaître dans l'index et la ou les pages sur lesquelles il se trouve.

2. Chaque fois que l'on désire faire apparaître un mot dans l'index, supposons le mot *compiler*, on doit placer à côté de lui la commande

   `\index{`*mot*`}`.

   Par exemple, on trouvera la ligne

   ```
   Il faut compiler\index{compiler} le fichier...
   ```

   Il est conseillé de placer la commande contre le mot sans laisser d'espace afin que la référence soit correcte même si c'est le dernier mot de la page. Remarquons que l'on peut très bien mettre cette instruction à côté de n'importe quel mot car LaTeX ne fais pas attention à la sémantique, mais uniquement à la syntaxe.
   Remarquons encore que cette commande ne produit aucune sortie dans le fichier .dvi, c'est juste la demande d'écriture dans le fichier .idx d'une instruction de la forme `\indexentry{compiler}{24}`, spécifiant dans ce cas l'occurence du mot `compiler` à la page 24. De plus, l'argument de cette commande peut comprendre n'importe quel caractère, y compris les caractères réservés, pour autant que les accolades soient présentes par paire (une ouvrante et une fermante).

3. A partir du fichier .idx, il faut créer un ficher d'entrée compréhensible pour LaTeX sous l'appellation *name.ind*. De nombreux programmes fournissent une aide pour la création de ce document (c'est le cas du module `makeidx`). Le nouveau document sera alors inclus dans le fichier d'entrée pour apparaître dans le fichier imprimable à l'endroit où l'utilisateur aura placé la commande

   `\printindex`.

La syntaxe de l'argument *mot* de la commande `\index` spécifie la manière dont les éléments apparaîtront dans l'index : le tableau suivant donne les options possibles et leurs effets dans l'index.

Tab. 6.1 – Syntaxe de la commande `\index`

Code source	Impression	
`\index{hello}`	hello, 1	Entrée normale
`\index{hello!bonjour}`	hello, 3	Sous-élément de 'bonjour'
`\index{Sam@\textsl{Sam}}`	*Sam*, 1	Entrée formatée
`\index{Sam@\textbf{Sam}}`	**Sam**, 2	Entrée formatée
`\index{Sam|textbf}`	Sam, **3**	Numéro de page formaté
`\index{Sam|textit}`	Sam, *1*	Numéro de page formaté

[1] Il faut également charger dans le préambule le module `makeidx` sauf avec l'extension `french` de `babel` qui gère automatiquement les index.

Remarque *Des instructions tout à fait équivalentes existent pour créer un glossaire, à partir des instructions* `\glossary` *et* `\glossaryentry`*.*

6.4 Bibliographie

La création de la bibliographie se fait au moyen de l'environnement `thebibliography`. A l'endroit où elle doit être générée, la définition de la bibliographie débute par la commande

`\begin{thebibliography}{`*max-larg*`}`

où *max-larg* représente la largeur maximale des étiquettes des entrées de la bibliographie et contrôle la mise en page : on utilisera usuellement la valeur 99.

Chaque élément de la bibliographie est introduit par une instruction

`\bibitem[`*bibnumber*`]{`*biblioref*`}`

qui génère une entrée dont l'étiquette est numérotée *bibnumber*. Si ce paramètre optionnel est omis, c'est le numéro de l'entrée qui est affiché. Le second paramètre *biblioref* est le nom de l'étiquette qui sera rappelé dans le texte par la commande

`\cite{`*biblioref*`}`.

La fin de l'environnement bibliographique est naturellement définie par l'instruction

`\end{thebibliography}`.

Lors de la compilation, LaTeX produit une page sur laquelle il affiche le titre "Bibliographie" de la même manière qu'un nom de chapitre, puis la succession des éléments bibliographiques numérotés. La bibliographie est renseignée dans la table des matière.

Il existe également des programmes , tel `\BibTeX`, qui permettent de gérer de manière plus perfectionnée les bibliographies.

Index