TEX  LATEX 2ε

Débuter avec LATEX

8 juin 2011

Autiwa

Résumé

Recueil des commandes et astuces que j'ai glané à droite à gauche au fil de mon apprentissage de LATEX

Table des matières

1 Généralités

1.1 Les Groupes

Pour écrire a0 au carré en tex, il ne faut pas uniquement mettre le 0 en indice et le 2 en exposant a_0^2, il faut utiliser les groupes pour que le rendu soit joli.

$$a_0^2$$
$$a_0{}^2$$

```
\[a_0^2\]
\[{a_0}^2\]
```

$\underbrace{\text{expression}}_{\text{commentaire}}$

```
\(\underbrace{\text{expression}}
_\text{commentaire}\)
```

permet de commenter et de donner des précisions sur une expressions. Je crois que overbrace fait la même chose mais par le haut.

1.2 Caractères spéciaux et commandes utiles

`\newpage` affichera les prochains caractères dans une nouvelle page.
\\ termine une ligne et renvoie à la ligne.
`\verb|commande non exécutée|` : la commande `\verb` permet de ne pas compiler du texte qui sera encadrée par n'importe quel caractère, le premier exemplaire de ce caractère se situant juste après la commande `\verb` et le deuxième juste à la fin.

$\backslash$

```
\(\backslash\)
```

mais il faut que ça soit en mode math

^ ou encore $\wedge$

```
\verb|^| ou encore \(\wedge\)
```

100%

```
100\%
```

— contenu de la parenthese — 1789–1989

```
--- contenu de la parenthese ---
1789--1989
```

1.3 Comment faire un paragraphe ?

Une ligne blanche dans le texte crée un saut de paragraphe avec indentation. Tant qu'il n'y a pas de ligne blanche, et même s'il n'y a qu'un mot par ligne, c'est le même paragraphe, qui sera formaté correctement dans le DVI. Et qu'il ait 1 ou 46 lignes blanches dans le fichier source, c'est la même chose : il n'y aura qu'un changement de paragraphe. Pour supprimer l'indentation, vous pouvez utiliser `\noindent`.

Pour faire une ligne blanche entre deux paragraphes, il faut laisser une ligne blanche et taper la commande `\bigskip`.

1.4 Textes en colonnes

Il est possible de présenter tout le texte en deux colonnes. Pour cela, on utilise l'argument **twocolumn** lors de l'appel de la classe, par exemple :

```
\documentclass[11pt, a4paper, twocolumn]{article}
```

On peut changer la disposition d'une page à l'autre :

– `\twocolumn` commence une nouvelle page en deux colonnes ;
– `\onecolumn` commence une nouvelle page en une colonne.

On peut utiliser un paramètre optionnel pour mettre un texte en une seule colonne au début d'une page à deux colonnes, comme par exemple un titre :

```
\twocolumn[\paragraph{''titre''}]
```

Les versions étoilées des environnements flottants *table** et *figure** permettent de placer les objets flottants sur la largeur de la page au lieu de la largeur d'une colonne.

Remarque : On peut aussi utiliser des environnements au lieu de commandes. Ainsi, on peut utiliser les environnements *twocolumn* et *onecolumn*, ils ont l'avantage de contenir tout le texte que l'on souhaite voir soumis au changement de nombre de colonnes.

Mais ces environnements ont un défaut majeur : ils commencent obligatoirement une nouvelle page. Impossible donc de mettre des titres en pleine pages, ou des équations, ou quoi que ce soit d'autre à part des flottants étoilés. Les paragraphes en dessous donnent la solution à ce problème.

Si l'on désire utiliser plus de colonnes, ou si l'on désire changer le nombre de colonne en cours de page, on utilise alors l'extension *multicol*. Cela permet d'utiliser l'environnement *multicols*, avec la syntaxe :

```
\begin{multicols}{''nombre de colonnes''}
   ''texte''
\end{multicols}
```

En résumé pour mettre du texte sur deux colonnes sans changer de pages, on fait comme ceci :

L'etat metallique est conduction de la chaleur et de l'electricite, un eclat particulier.

L'etat metallique est conduction de la chaleur et de l'electricite, un eclat particulier.

L'etat metallique est conduction de la chaleur et de l'electricite, un eclat particulier.

L'etat metallique est conduction de la chaleur et de l'electricite, un eclat particulier.

```
L'etat metallique est
conduction de la chaleur
 et de l'electricite,
un eclat particulier.
\begin{multicols}{2}
L'etat metallique est
conduction de la chaleur
 et de l'electricite,
un eclat particulier.

L'etat metallique est
conduction de la chaleur
 et de l'electricite,
un eclat particulier.
\end{multicols}
L'etat metallique est
conduction de la chaleur
 et de l'electricite,
un eclat particulier.
```

1.5 Appliquer une rotation

Le paquet *graphics* (à moins que ça ne soit *graphicx*) permet d'appliquer une rotation à pas mal de choses, que ça soit du texte, des boites, des tableaux, sans doute des images aussi.

Pour l'utiliser, on fait :

Mon texte

```
\rotatebox{90}{Mon texte}
```

1.6 Mettre en valeur du texte

roman
sans serif
machine a ecrire
moyen
gras
droit
penche
italique
PETITES MAJ
important
ceci est un texte important
𝔄𝔅ℭ𝔇𝔈𝔉𝔊ℌℑ𝔍𝔎𝔏𝔐𝔑𝔒𝔓𝔔ℜ𝔖𝔗𝔘𝔙𝔚𝔛𝔜ℨ

```
\textrm{roman}\\
\textsf{sans serif}\\
\texttt{machine a ecrire}\\
\textmd{moyen}\\
\textbf{gras}\\
\textup{droit}\\
\textsl{penche}\\
\textit{italique}\\
\textsc{petites maj}\\
\emph{important}\\
\textit{ceci est un texte
\emph{important}}\\
$\mathfrak{ABCDEFGHIJKLMN
OPQRSTUVXYZ}$
```

1.7 Aligner du texte par rapport aux lignes précédentes

matrix pour une matrice normale
pmatrix parentheses
bmatrix crochets
vmatrix lignes verticales
Vmatrix doubles lignes

```
\begin{tabbing}
\textbf{matrix} pour une \= matrice normale \\
\textbf{pmatrix} \> parentheses\\
\textbf{bmatrix} \> crochets\\
\textbf{vmatrix} \> lignes verticales\\
\textbf{Vmatrix} \> doubles lignes\\
\end{tabbing}
```

1.8 Les règles de typographie française

(Voir la partie 6.2 sur le package *babel*)

Il existe des commandes spécifiques aux documents français. Par exemple, pour les guillemets, ce ne sont pas les mêmes que les guillemets anglais. Il existe deux commandes, pour les guillemets longs ou courts (utilisez par exemple les guillemets courts pour des citation à l'intérieur de citations [1]) :

« guillemet “Francais” »

```
\og guillemet ``Francais'' \fg
```

Il faut ensuite rajouter le paquet *xspace* qui permet à *babel* de gérer les espaces comme il faut. Par exemple, pour l'utilisation des guillemets, au lieu de devoir écrire `\og{} texte en valeur \fg{} suite du texte`, il suffit d'écrire `\og texte en valeur \fg suite du texte`.

1.9 Personnaliser les listes

On peut soit modifier les listes existantes, soit en créer de nouvelles.

1.9.1 Personnaliser les listes numérotées

Ce que souvent les personnes veulent changer dans les listes numérotées sont les compteurs. Par conséquent, pour mieux comprendre, nous avons besoin d'introduire brièvement les compteurs de LaTeX. À tout objet que LaTeX numérote automatiquement, comme les en-têtes de section, les figures, et les listes, est associé un compteur qui contrôle la numérotation. De plus chaque compteur possède un format par défaut qui dicte à LaTeX la façon dont il doit être imprimé. De tels formats sont modifiés en utilisant des commandes internes de LaTeX :

1. Je ne connais la véritable règle alors pour ma part, j'utilise tout le temps les guillemets longs, et j'utilise les guillemets courts à l'intérieur des guillemets longs

Commande : Exemple
`\arabic` : 1, 2, 3 ...
`\alph` : a, b, c ...
`\Alph` : A, B, C ...
`\roman` : i, ii, iii ...
`\Roman` : I, II, III ...
`\fnsymbol` : destinés à la numérotation des apostilles (je sais pas ce que ça veut dire, mais ça a l'air intelligent), mais imprime une suite de symboles.

Il existe quatre compteurs différents qui sont associés aux listes à puces, chacun représentant les quatre niveaux possibles d'imbrication, et ils s'appellent : enumi, enumii, enumiii, enumiv. Chaque compteur contient plusieurs bits de données fournissant différentes informations. Pour obtenir l'élément numéroté, employez simplement la commande `\the` suivie immédiatement (c'est-à-dire sans aucun espace) du nom du compteur, par exemple `\theenumi`. Cette information est souvent désignée sous le nom de représentation du compteur.

Maintenant, laissons la plupart des technicités de côté. Pour effectuer des changements sur la mise en forme d'un niveau donné :

```
\renewcommand{\représentation}{\commande_de_mise_en_forme{compteur}}.
```

Évidemment, la version générique n'est pas vraiment claire, aussi une paire d'exemples clarifiera :
Redéfinition du premier niveau :

```
\renewcommand{\theenumi}{\Roman{enumi}}
\renewcommand{\labelenumi}{\theenumi}
```

Redéfinition du deuxième niveau :

```
\renewcommand{\theenumii}{\Alph{enumii}}
\renewcommand{\labelenumii}{\theenumii}
```

La méthode utilisée ci-dessus change d'abord explicitement le format d'impression employé par le compteur. Cependant, l'élément qui contrôle l'étiquette doit être mis à jour pour refléter le changement, et cet ajustement est effectué par la deuxième ligne. Une autre manière d'obtenir le même résultat est :

```
\renewcommand{\labelenumi}{\Roman{enumi}}
```

Cela redéfinit simplement l'aspect de l'étiquette, ce qui suppose que vous n'avez pas l'intention d'établir des renvois à un article spécifique de la liste, auquel cas la référence serait imprimée dans l'ancien format. Ce problème n'apparaît pas dans le premier exemple.

1.9.2 Personnaliser les listes à puces

Les listes à puces ne sont pas aussi complexes que les listes numérotées puisqu'elles n'utilisent pas de compteur. Ainsi, pour personnaliser de telles listes, vous pouvez juste changer les puces (étiquettes). Les puces sont accessibles via les commandes `\labelitemi`, `\labelitemii`, `\labelitemiii`, et `\labelitemiv` pour les quatre niveaux respectifs.

```
\renewcommand{\labelitemi}{\textgreater}
```

L'exemple ci-dessus imposerait aux puces du premier niveau d'être représentées par le symbole « > » strictement supérieur. Bien sûr, les symboles de texte en LaTeX ne sont pas très impressionnants.

1.9.3 Définition de nouveaux environnements de liste

L'environnement *list* permet de définir son propre style de liste. Sa syntaxe est la suivante :

```
\begin{list}{label}{mep}\end{list}
```

- l'argument *label* permet de définir le symbole qui sera associé à chaque élément de la liste.
- *mep* permet de définir la mise en page des éléments de la liste. Les paramètres utilisés pour définir cette mise en page sont les suivants :

- `\topsep` espace vertical supplémentaire (ajoute à `\parskip`) inséré entre le texte précédant la liste et le 1er objet de la liste
- `\partosep` espace vertical supplémentaire inséré devant la liste si celle-ci est précédée d'une ligne blanche
- `\itemsep` espace vertical supplémentaire (ajouté à `\parsep`) inséré entre les éléments d'une liste.

Exemple :

```
\newenvironment{maliste}%
{ \begin{list}%
{\)\bullet\)}%
{\setlength{\labelwidth}{30pt}%
 \setlength{\leftmargin}{35pt}%
 \setlength{\itemsep}{\parsep}}}%
{ \end{list} }
```

Utilisation :

```
\begin{maliste}
   \item premier élément
   \item deuxième élément
   \begin{maliste}
      \item petit 1
      \item petit 2
   \end{maliste}
\end{maliste}
```

1.10 Générer un index

index Pour générer un *index*, il faut tout d'abord rajouter les commandes suivantes dans l'entête du document

```
\usepackage{makeidx}
\makeindex
```

et cette commande juste avant la fin du document

```
\printindex
\end{document}
```

Pour rajouter une entrée dans l'index, on utilise la commande `\index{mot a indexer}`.*index*

Pour construire un index à plusieurs niveaux d'entrée, il faut utiliser les commandes `\index{niveau1}` comme précédemment puis, pour faire apparaître un sous-thème de ce niveau, on appellera `\index{niveau1!niveau1.1}`; comme l'illustre l'exemple suivant :

```
\begin{document}
Le sport\index{Sport} c'est fantastique~!

Mes sports préférés sont~:
\begin{itemize}
   \item l'équitation\index{Sport!Equitation} et en particulier
   les disciplines de dressage\index{Sport!Equitation!Dressage}
   et de complet\index{Sport!Equitation!Complet}~:
   \item l'escalade\index{Sport!Escalade} et surtout les
   sorties en falaise~;
   \item le judo\index{Sport!Judo}.
\end{itemize}
```

Pour spécifier un nom différent du nom qui sert à l'indexation, on utilise :

```
\index{equation@équation}
\index{fonction!gamma@$\Gamma(x)$}
```

Dans le premier cas, `equation` servira pour le classement alphabétique, mais c'est **équation** qui sera affiché. Dans le second exemple, `gamma` servira pour l'indexation, mais c'est $\Gamma(x)$ qui sera affiché.

Il arrive que l'on ai plusieurs noms pour la même idée. Dans ce cas, il existe une commande pour spécifier, dans l'index, d'aller voir à une autre référence :

```
\index{fonction!de Gauss|see{gaussienne}}
```

1.11 Créer ses propres commandes

1.11.1 Les bases

1.11.2 Pour gérer des formules

La commande `\ensuremath` assure que son argument sera imprimé en mode mathématique quel que soit le mode courant.

```
\documentclass{report}
\usepackage{french}
\pagestyle{empty}
\newcommand{\mc}{\ensuremath{(\alpha, \beta)}}
\begin{document}
Le couple \mc\ définit par \)\mc = x+y, x-y\), \dots
\end{document}
```

1.11.3 Pour aller plus loin

ifthen Il existe un paquet extrêmement pratique qui s'appelle *ifthen*. Pour voir en détail comment il fonctionne, rien de mieux que le mode d'emploi du paquet lui même. Voici quelques exemples de ce qu'il est possible de faire, suivit de l'exemple de la commande `\gras{}` que j'ai définie :

Un exemple simple :

```
\newcommand{\printTrueOrFalse}[1]
{
  \ifthenelse{\equal{#1}{true}}{TRUE}{}
  \ifthenelse{\equal{#1}{false}}{FALSE}{}
}

\printTrueOrFalse{true}
```

Un exemple un peu plus complexe :

```
\newcommand{\dayOfWeek}[1]
{
  \ifthenelse{\equal{#1}{0}}{Sunday}{}
  \ifthenelse{\equal{#1}{1}}{Monday}{}
  \ifthenelse{\equal{#1}{2}}{Tuesday}{}
  \ifthenelse{\equal{#1}{3}}{Wednesday}{}
  \ifthenelse{\equal{#1}{4}}{Thursday}{}
  \ifthenelse{\equal{#1}{5}}{Friday}{}
  \ifthenelse{\equal{#1}{6}}{Saturday}{}
}

\dayOfWeek{0}
\dayOfWeek{1}
\dayOfWeek{2}
\dayOfWeek{3}
\dayOfWeek{4}
\dayOfWeek{5}
\dayOfWeek{6}
```

J'utilise une commande que j'ai créé qui s'appelle `\gras`. Celle-ci possède deux arguments, dont un optionnel. En clair, le 2e argument est le texte qui s'affiche à l'écran et qui aura un changement de fonte et de couleur pour être mis en valeur. S'il n'y a pas de 1er argument, celui-ci vaudra par défaut le 2e, et j'indexe donc ce mot là. Si l'argument 1 existe, alors sa valeur sera le texte donnée à la commande `\index` lors de l'indexation.

En clair, voici la définition complète de la macro :

```
\newcommand{\oldgras}[2]{{{\color{marou} \textsl{#2}}\black\index{#1}}}
\newcommand{\gras}[2][]{\ifthenelse{\equal{#1}{}}%
                              {\oldgras{#2}{#2}}%   % argument optionnel vide
                              {\oldgras{#1}{#2}}}   % argument optionnel non-vide
```

1.12 Réaliser des boucles

LaTeX permet de réaliser des boucles de plusieurs manières. L'extension *ifthen* propose une commande `\whiledo` qui fonctionne ainsi :

```
\whiledo{test}{faire}
```

Par exemple, le fichier suivant

```
\documentclass{article}
\usepackage{ifthen}
\newcounter{cnt}
\begin{document}
\setcounter{cnt}{0}
\whiledo{\value{cnt}<100}{%
  \stepcounter{cnt}%
  \textbf{\thecnt}%
  \ifthenelse{\value{cnt}=100}{.}{, }%
}
\end{document}
```

produit-il une liste « 1, 2, . . ., 100. »

L'extension *multido* propose une autre méthode grâce à la commande particulièrement pratique `\multido` :

```
\documentclass{article}
\usepackage{multido}
\begin{document}
\multido{\nA=1+1}{99}{\textbf{\nA}, }\textbf{100}.
\end{document}
```

Remarquez que 100 nécessite un traitement spécial puisqu'il n'est pas suivi d'une virgule, étant le dernier élément de la liste. Ce traitement était effectué automatiquement dans la version avec *ifthen* mais pas dans l'exemple précédent. Bien sûr on peut combiner les avantages de *multido* et de *ifthen*. . .

```
\documentclass{article}
\usepackage{multido}
\usepackage{ifthen}
\begin{document}
\multido{\nA=1+1}{100}{\textbf{\nA}\ifthenelse{\nA=100}{.}{, }}
\end{document}
```

Bien sûr les extensions *multido* et *ifthen* sont orientées vers des applications numériques.

Il existe une commande interne `\@for` (les `@` impliquent de placer le code correspondant entre `\makeatletter` — fait de l'arobas une lettre — et `\makeatother` — fais de l'arobe autre chose qu'une lettre ou dans un fichier de style) dont la syntaxe est donnée par

```
\@for \@tempa :=liste d'items\do{%
  corps à exécuter pour chaque item \@tempa
}
```

où `\@tempa` enregistre successivement chacun des items qui doivent être séparés par une virgule dans la liste. L'application plus que numérique est liée aux listes. Un exemple d'utilisation pourrait être

```
\@for \@tempa :=titi,toto,tata\do{Bonjour \@tempa.\ }
```

qui affiche « Bonjour titi. Bonjour toto. Bonjour tata. ». Chose qui paraît difficilement concevable avec les extensions précédentes.

Néanmoins, plus que l'aspect boucle numérique ou liste, c'est l'aspect non développable qui nous préoccupe ici. Peu importe ce que cela signifie précisément, intuitivement cela correspond à l'idée que toutes les méthodes vues précédemment impliquent une machinerie lourde qui est mouline en silence et à l'insu de l'utilisateur et qui la rend inopérante dans certaines circonstances. Quelles circonstances ? Dans les titres de chapitre, section, etc. (`\chapter` et autres), légendes (`\caption`) et de manière surprenante dans les tableaux...

Il est par exemple interdit de faire

```
\begin{tabular}{c}
\multido{\nA=1+1}{100}{\textbf{\nA}\\}
\end{tabular}
```

qui donne un message d'erreur assez complexe.

Ce n'est pas étonnant car dans la vision des choses de LaTeX chaque case d'un tableau est une petite cellule qui aime son indépendance... À tel point que la première essaie de garder `\multido` pour elle toute seule, ce qui provoque un conflit d'intérêt que TeX n'arrive pas à résoudre. C'est cette indépendance qui explique que dans le tableau suivant

```
\begin{tabular}{cc}
\itshape cellule 1 & cellule 2 \\
\end{tabular}
```

seule la première cellule soit en italique.

Revenons à nos moutons... Le but est donc de créer une boucle développable. Le but est donc de préférer une forme de légèreté à la lourdeur des extensions multido, ifthen ou de la commande interne `\@for` pour que, pour parler de façon imagée, notre commande puisse s'échapper de l'emprise des cellules de tableaux. Le code est le suivant

```
\makeatletter
\newcommand*{\For}[1]{\noalign{\gdef\@Do##1{#1}}\@For}
\newcommand*{\@For}[3]{%
  \unless\ifnum#1 \ifnum#3>0 >\else <\fi #2
    \@Do{#1}%
    \expandafter\@For\expandafter{\number\numexpr#1+#3}{#2}{#3}%
  \fi
}
```

et l'utilisation est la suivante

```
\For{utilisation de la valeur (#1)}{init}{end}{incr}
```

Par exemple

```
\begin{tabular}{ll}
  \For{un nombre & #1\\}{1}{20}{1}
\end{tabular}
```

2 Les Tableaux

On définit un tableau comme suit :

a	b
c	d

```
\begin{tabular}{|c|c|}
\hline
a & b \\ \hline
c & d\\ \hline
 \end{tabular}
```

2.1 Définition des colonnes

`l`, `r` et `c` permettent respectivement d'aligner les colonnes à gauche, droite ou centré.`p{longueur}` définit une colonne paragraphe. la commande `m{longueur}` fait la même chose mais centre verticalement le texte, mais pour celà, il faut le package *array*.

Il est possible de définir plusieurs colonnes en même temps via la commande `*{num}{cols}`. Par exemple `*{4}{p{2cm}|}` permet de définir 4 colonnes de textes de largueur 2cm avec un trait vertical entre chacune d'elles.

2.2 Commandes à l'intérieur du tableau

`&` : permet de séparer les colonnes
`\\` : marque la fin d'une ligne.
`\hline` : trace un trait horizontal et les traits verticaux se définissent dans l'inventaire des colonnes.
`\cline{i-j}` : ligne horizontale entre les colonnes **i** et **j**
`\vline` : ligne verticale.
`\multicolumn{n}{cols}{item}` : mise de **item** sur **n** colonnes avec les règles **cols**.
`@{ : }` : permet de séparer les colonnes avec autre chose que `|`.

Remarque : À noter que pour @{ : }, les espaces sont là pour que les deux colonnes ne soient pas trop collées.

Texte	
donnee 1	donnee 2
A	B

```
\begin{center}
\begin{tabular}[b]{|l|c|}
\hline
\multicolumn{2}{|c|}{Texte} \\
\hline \hline
donnee 1 & donnee 2 \\
A & B \\
\hline
\end{tabular}
\end{center}
```

`\multirow{n}*{item}` : nécessite le package *multirow*. Permet de fusionner n lignes entre elles.

k	p_G	test	
		DADWRD	RARWRD
2	1	90 n	228 n
3	p_d	202 n	449 n
4	p_d^2	424 n	891 n
5	p_d^3	866 n	1774 n

```
\begin{center}
\begin{tabular}{|c|c||c|c|}
\hline
\multirow{2}*{k}
 & \multirow{2}*{\(p_G\)}
 & \multicolumn{2}{c|}{test} \\
\cline{3-4}
& & DADWRD & RARWRD \\
\hline
2 & \(1\) & 90 n & 228 n \\
3 & \(p_d\) & 202 n & 449 n \\
4 & \(p_d^2\) & 424 n & 891 n \\
5 & \(p_d^3\) & 866 n & 1774 n \\
\hline
\end{tabular}
\end{center}
```

L'extension *slashbox* permet de scinder une cellule en deux selon la diagonale, par exemple pour la cellule en haut à gauche du tableau, on utilise :

```
\backslashbox{titre de la colonne}{titre de la ligne}
```

2.3 Définir une colonne en mode mathématique

Avec le package *array*, il suffit de déclarer `\begin{tabular}{>{$}c<{$}cc}` pour avoir une colonne en mode mathématique et deux colonnes de texte.

$2x+3=1$	equation difficile
$3x^2+y^2=3$	definition d'un cercle

```
\begin{tabular}{>{$}c<{$}cc}
2x+3=1 & equation difficile\\
3x^2+y^2=3 & definition d'un cercle
\end{tabular}
```

On peut aussi déclarer plein de choses avec cette manière :

`>{\bfseries}r<{}`	Permet de définir une colonne en gras
`>{\red}r<{}`	Permet de définir une colonne en couleur
`>{\centering}m{2cm}<{}`	Permet de définir une colonne de texte centré à la fois horizontalement, et verticalement (`m{2cm}`)
...	Et ainsi de suite

3 Manipuler les figures

3.1 Faire des figures et des schémas

Ceci est une question souvent très compliquée à traiter. Il existe le package *pstricks* qui permet de réaliser des figures. Même si, de mémoire, ce package est très puissant, il nous oblige à passer par un fichier .ps qui nous fait perdre pas mal de choses, en particulier les hypers liens, et peut-être même les polices vectorielles.

Bref, utiliser un logiciel tierce parait envisageable voire vital. Dans la pratique, j'utilise *Inkscape*. Je fais mon dessin normalement, puis je l'exporte en **.pdf**. Ça donne des figures très propres (pour peu qu'on fasse du vectoriel correct) et qui prennent peu de place tout en étant redimensionnable sans perte de qualité. Il suffit ensuite de l'ajouter au fichier **.tex**.

Pour les légendes et écritures, il existe deux solutions :

- Soit on écrit les choses par dessus via la méthode détaillée dans la section 3.2
- Soit on écrit directement dans le fichier vectoriel. L'avantage de cette dernière méthode est que c'est beaucoup plus facile à inclure que de chercher les bonnes coordonnées pour chaque items à écrire. Celà dit, avec une macro pour inclure du texte LaTeX dans un **.svg**, la taille des textes et formules générés augmente d'une manière non négligeable la taille du fichier **.svg**. À vous de voir ce que vous préférez.

3.2 superposer du texte sur des figures

Pour afficher des figures, je les réalise en dessin vectoriel sur *Inkscape* puis j'enregistre une copie en **.pdf**. Le PDF étant une sorte d'affichage vectoriel, on peut redimensionner sans pixelisation, et c'est pour l'instant le meilleur compromis que j'ai trouvé.

Il existe *PStricks* mais on ne peut plus compiler avec pdflatex et j'avoue que je trouve ça vraiment peu pratique. On peut aussi utiliser *Xfig* qui permet d'exporter en LaTeX mais on doit retoucher le texte qui n'est pas très joli et le code LaTeX prend de suite des proportions incroyables.

Donc dans la suite, je suppose que la figure est faite, et en .pdf. Pour superposer du texte[2], il faut procéder comme dans l'exemple ce-dessous :

```
\begin{figure}[htbp]
\centering
\setlength\unitlength{1cm}
\begin{picture}(10,10)
\put(0,0){\includegraphics[height=10cm]{huygens.pdf}}
\put(0,6){\)n_1\)}
\put(0,5.5){\)n_2\)}
\end{picture}
\caption{Construction de Huygens}
\end{figure}
```

2. c'est plus propre de le superposer que de l'inclure dans le .pdf, comme ça, on peut afficher des formules mathématiques avec LaTeX de la complexité que l'on veut.

3.2.1 Astuce

Pour trouver rapidement les coordonnées des points où l'on veut rajouter du texte, on peut ajouter temporairement un quadrillage en espaçant des traits plein tout les centimètres et des pointillés tout les demi-centimètres. Ceci marche dans mon cas puique l'unité de longueur est le centimètre et que je met mes figures dans un cadre de 10 par 10. À vous d'adapter suivant votre cas :

```
\multiput(0,0)(1,0){11}{\line(0,1){10}}
\multiput(0.5,0)(1,0){10}{\multiput(0,0)(0,0.2){50}{\line(0,1){0.1}}}
\multiput(0,0)(0,1){20}{\line(1,0){10}}
\multiput(0,0.5)(0,1){10}{\multiput(0,0)(0.2,0){50}{\line(1,0){0.1}}}
```

3.3 Gérer les tailles de figures

J'avais pris l'habitude d'ajouter les figures, et de spécifier, quand j'en avais besoin, la largeur ou la hauteur en centimètre. Il est pourtant possible de gérer la largeur beaucoup plus proprement, notament si on oscille entre le mode page, et le mode deux colonnes.

Pour celà, on utilise `[width=0.8\linewidth]` qui spécifie à LaTeX que la figure doit être redimensionnée pour que la largeur de la figure fasse 0.8 fois la largeur de la ligne, en clair, la largeur de la page. Et si entre temps on passe en mode colonne, la figure est redimensionnée à la largeur de la colonne.

De plus, si on utilise le mode deux colonnes et que l'on utilise `\begin{figure*}` la figure s'affiche sans tenir compte des colonnes (pas testé personnellement, mais il parait)

3.4 Mettre plusieurs figures à coté

Il faut pour celà utiliser le paquet *subfig*

```
\begin{figure}[htb]
\centering
\subfloat[1\ier{} figure]{\label{fig:figure1}\includegraphics[width=0.2\textwidth]{figure/logo-ubuntu.pdf}}\hfill
\subfloat[2\ieme{} figure]{\label{fig:figure2}\includegraphics[width=0.2\textwidth]{figure/logo-debian.pdf}}
\caption{Voici plusieurs figures dans un même environnement}
\end{figure}
```

Remarque : La commande `\hfill` permet d'optimiser l'espacement entre les figures. C'est surtout pratique quand il y a plus de deux figures sur la même ligne. Ça permet, pour deux figures, d'aligner la première à gauche, et la deuxième à droite.

(a) 1er figure

(b) 2e figure

FIGURE 1 – Voici plusieurs figures dans un même environnement

On peut appeler les figures avec des références séparées, ou rajouter un *label* à droite du *caption* pour faire référence à tout le groupe.

3.5 Fondre une figure dans un paragraphe

Il faut pour celà utiliser le paquet *floatflt* (il en existe cependant d'autres).

FIGURE 2 – Logo d'Ubuntu

du texte du texte

```
\begin{floatingfigure}[l]
{0.3\linewidth}
\includegraphics[width=0.3\linewidth]%
{figure/logo-ubuntu.pdf}
\caption{Logo d'Ubuntu}
\end{floatingfigure}
du texte du texte du texte du texte
du texte du texte du texte du texte
du texte du texte du texte du texte
du texte du texte du texte du texte
du texte du texte du texte du texte
du texte du texte du texte du texte
du texte du texte du texte du texte
du texte du texte du texte du texte
du texte du texte du texte du texte
du texte du texte du texte du texte
du texte du texte du texte du texte
du texte du texte du texte du texte
```

4 Mathématiques

4.1 Commandes utiles

$$\begin{aligned} f(x) &= x^2 + 8x + 16 && (4.1) \\ &= (x+4)^2 && (4.2) \end{aligned}$$

```
\begin{align}
 f(x) & = x^2 + 8x + 16 \\
 & = (x+4)^2
\end{align}
```

$$\begin{aligned} a &= 1 & b &= 2 && (4.3) \\ c &= 3 & d &= 4 && (4.4) \end{aligned}$$

```
\begin{align}
a&=1 & b&=2\\
c&=3 & d&=4
\end{align}
```

Permet d'aligner en colonne. Mettre une "*" permet d'enlever la numérotation des équations. En mettant un `\notag`, on supprime la numérotation d'une seule ligne. En répétant les signes « & », on peut alors mettre en page plusieurs équations sur la même ligne. Les « & » impairs séparent les deux membres d'une équation tandis que les « & » pairs séparent les équations

$\stackrel{\mathrm{déf}}{=}$ `$\stackrel{\mathrm{d\acute ef}}{=}$`

$\sum_{\substack{i=1\\ i\neq j}}$ `$\sum_{\substack{i=1\\ i\neq j}}$`

permet de mettre deux chaines de caractères l'une au dessus de l'autre. (du genre dans ce cas, un **déf** au dessus d'un signe égal)

4.2 Définitions à choix multiples (accolade)

$$f(x) = \begin{cases} f(x_0) & \text{si } x < x_0 \\ f(x_1) & \text{si } x_0 < x < x_1 \\ f(x_2) & \text{si } x > x_1 \end{cases} \tag{4.5}$$

```
\begin{align}
  f(x)&=
  \begin{cases}
    f(x_0)  & \text{si }  x<x_0\\
    f(x_1) & \text{si }  x_0<x<x_1\\
    f(x_2) & \text{si }  x>x_1
  \end{cases}
\end{align}
```

permet de faire une accolade avec des cas, le « & » aligne une deuxième partie.

4.3 Exposant, indice

Pour mettre du texte en exposant, on place le texte dans un bloc et on le fait précéder d'un chapeau ^.

Pour mettre du texte en indice, on place le texte dans un bloc et on le fait précéder d'un tiret de soulignement _ .

par exemple :

$$\begin{aligned} u_n &= 2^n \\ u_n + 1 &= 2^n + 1 \\ u_{n+1} &= 2^{n+1} \end{aligned}$$

```
\begin{align*}
u_n = 2^n \\
u_n+1 = 2^n+1 \\
u_{n+1} = 2^{n+1}
\end{align*}
```

On peut placer un objet au-dessus ou en dessous d'un autre.

$$\overset{a}{X}$$
$$\underset{b}{X}$$
$$\overset{a}{\underset{b}{X}}$$

```
\begin{align*}
\overset{a}{X}\\
\underset{b}{X}\\
\overset{a}{\underset{b}{X}}
\end{align*}
```

4.4 Faire des flèches adaptées à la longueur d'un texte

La commande ci-dessous permet de générer des flèches dont la longueur dépend de la longueur du texte qui est placé au dessus ou en dessous (ou de la chaîne la plus longue lorsqu'il y a à la fois un texte au dessus et un autre en dessous).

$\xrightarrow[\text{en dessous}]{\text{au dessus}}$

```
\(\xrightarrow[\text{en dessous}]%
{\text{au dessus}}\)
```

4.5 Les matrices

Ce sont des environnements à utiliser en mode mathématique.

matrix : est une matrice sans aucun encadrement
pmatrix : est une matrice entourée de parenthèses
bmatrix : est une matrice entourée de crochets
vmatrix : est une matrice entourée de barres verticales
Vmatrix : est une matrice entourée de doubles barres verticales

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

```
\[\begin{vmatrix}
  a & b\\
c & d
  \end{vmatrix}\]
```

4.6 Mettre en page des équations

4.6.1 Pour commencer

Quelques commandes LaTeX utiles sont disponibles sous forme de paquets nommés *amsmath*. Ils peuvent être chargés sur la plupart des installations de LaTeX en plaçant dans le préambule de votre document cette commande.

Une commande très utile rend possible, au moyen de ces paquets, l'alignement des objets, par exemple le signe d'égalité « = » sur des lignes d'équations successives. Voici un exemple :

$$x = a + (b + a) \tag{4.6}$$
$$= 2a + b. \tag{4.7}$$

```
\begin{align}
x &= a + (b + a) \\
&= 2a + b.
\end{align}
```

Le symbole esperluette « & » est utilisé pour préciser les points dans chaque ligne qui seront supposés être alignés verticalement, mais n'apparaissent pas dans le résultat final.

Les équations sont numérotées par défaut. Les numéros d'équation peut être supprimés en remplaçant le mot **align** par le mot **align*** au début et à la fin des équations à aligner :

$$x = a + (b + a)$$
$$= 2a + b.$$

```
\begin{align*}
x &= a + (b + a) \\
&= 2a + b.
\end{align*}
```

Une autre solution est de les supprimer sur une ligne particulière en ajoutant l'expression `\notag`, comme nous le montrons ici :

$$x = a + (b + a)$$
$$= 2a + b. \tag{4.8}$$

```
\begin{align}
x &= a + (b + a) \notag \\
&= 2a + b.
\end{align}
```

Des annotations dans chaque ligne peuvent être ajoutées en les séparant des équations, en utilisant deux esperluettes « && » :

$$x = a + (b + a) \quad \text{Axiome C} \tag{4.9}$$
$$= 2a + b \quad \text{Axiomes A \& F.} \tag{4.10}$$

```
\begin{align}
x &= a + (b + a) && \text{Axiome C} \\
&= 2a + b && \text{Axiomes A \& F}.
\end{align}
```

Remarque : L'environnement *align* est fait pour mettre en page plusieurs équations sur la même ligne. Ainsi, les colonnes sont alternativement alignés à droite et à gauche pour pouvoir aligner chaque colonne d'équations sur les signes de relation. De plus, chaque colonne alignée à droite a un espace important avec la colonne immédiatement précédente, ceci pour séparer les différentes colonnes d'équations. D'où la présence des deux esperluettes dans l'exemple précédent. En effet, la première donne un espace important, mais un alignement à droite ; or on veut que les commentaires commencent au même endroit, d'où la présence de la 2e esperluette.

4.6.2 Sous-équations

Il s'avère souvent pratique, lorsqu'on a un ensemble d'équations se rapportant plus ou moins à la même chose, de les numéroter de la même manière de la façon suivante :

$$x = a + (b + a) \tag{4.11a}$$
$$y = 2c + d \tag{4.11b}$$

```
\begin{subequations}
\begin{align}
x &= a + (b + a) \\
y&= 2c + d
\end{align}
\end{subequations}
```

4.6.3 Mettre en page de longues équations

On peut utiliser *split* :

$$H_c = \frac{1}{2n}\sum_{l=0}^{n}(-1)^l(n-l)^{p-2}\sum_{l_1+\dots+l_p=l}\prod_{i=1}^{p}\binom{n_i}{l_i} \cdot [(n-l)-(n_i-l_i)]^{n_i-l_i}\cdot\left[(n-l)^2-\sum_{j=1}^{p}(n_i-l_i)^2\right]. \tag{4.12}$$

```
\begin{equation}\label{e:barwq}
\begin{split}
H_c&=\frac{1}{2n} \sum^n_{l=0}%
(-1)^{l}(n-{l})^{p-2}
\sum_{l _1+\dots+ l _p=l}%
\prod^p_{i=1} \binom{n_i}{l _i}\\
&\quad\cdot[(n-l )%
-(n_i-l _i)]^{n_i-l _i}\cdot
\Bigl[(n-l )^2-\sum^p_{j=1}%
(n_i-l _i)^2\Bigr].
\end{split}
\end{equation}
```

Ou encore dans cet exemple suivant qui numérote des sous équations, et en plus, met en page une longue équation (la deuxième) :

```
\begin{subequations}
\begin{align}
E_n&= E_n^0+\bra{\varphi_n}W\ket{\varphi_n}+\sum_{p\neq n}\frac{\abs{\bra{\varphi_p}W\ket{\varphi_n}}^2}%
{E_n^0-E_p^0}\\
\begin{split}
\ket{\psi_n}&= \left[1-\inv{2}\sum_{p\neq n}\frac{\abs{\bra{\varphi_p}W\ket{\varphi_n}}^2}{E_n^0-E_p^0}%
\right]\ket{\varphi_n}\\
&+\sum_{p\neq n}\left\{\frac{\bra{\varphi_p}W\ket{\varphi_n}}{E_n^0-E_p^0}+\sum_{p'\neq n}%
\frac{\bra{\varphi_{p'}}W\ket{\varphi_n}\bra{\varphi_p}W\ket{\varphi_{p'}}}{(E_n^0-E_p^0)(E_n^0-E_{p'}^0)}%
-\frac{\bra{\varphi_n}W\ket{\varphi_n}\bra{\varphi_p}W\ket{\varphi_n}}{E_n^0-E_p^0}\right\}
\end{split}
\end{align}
\end{subequations}
```

$$E_n = E_n^0 + \langle\varphi_n|W|\varphi_n\rangle + \sum_{p\neq n}\frac{|\langle\varphi_p|W|\varphi_n\rangle|^2}{E_n^0-E_p^0} \tag{4.13a}$$

$$\begin{aligned}|\psi_n\rangle = &\left[1-\frac{1}{2}\sum_{p\neq n}\frac{|\langle\varphi_p|W|\varphi_n\rangle|^2}{E_n^0-E_p^0}\right]|\varphi_n\rangle\\ &+\sum_{p\neq n}\left\{\frac{\langle\varphi_p|W|\varphi_n\rangle}{E_n^0-E_p^0}+\sum_{p'\neq n}\frac{\langle\varphi_{p'}|W|\varphi_n\rangle\langle\varphi_p|W|\varphi_{p'}\rangle}{(E_n^0-E_p^0)(E_n^0-E_{p'}^0)}-\frac{\langle\varphi_n|W|\varphi_n\rangle\langle\varphi_p|W|\varphi_n\rangle}{E_n^0-E_p^0}\right\}\end{aligned} \tag{4.13b}$$

Et un dernier exemple pour montrer comment faire usage de `\left(` et `\right)` dans un environnement *split*

```
\begin{align*}
\begin{split}
F&=\left|\cos\left(2\sqrt{\Delta^2+\abs{W_{12}}^2} \sfrac{t}{\hbar}\right)
-i\cancel{\sin\left(2\sqrt{\Delta^2+\abs{W_{12}}^2} \sfrac{t}{\hbar}\right)}\right.\\
&\left.+\cos\left(2\sqrt{\Delta^2+\abs{W_{12}}^2} \sfrac{t}{\hbar}\right)
+i\cancel{\sin\left(2\sqrt{\Delta^2+\abs{W_{12}}^2} \sfrac{t}{\hbar}\right)}-2\right|
\end{split}
\end{align*}
```

$$F = \left|\cos\left(2\sqrt{\Delta^2+|W_{12}|^2}\,{}^{t}\!/_{\hbar}\right) - \cancel{i\sin\left(2\sqrt{\Delta^2+|W_{12}|^2}\,{}^{t}\!/_{\hbar}\right)} + \cos\left(2\sqrt{\Delta^2+|W_{12}|^2}\,{}^{t}\!/_{\hbar}\right) + \cancel{i\sin\left(2\sqrt{\Delta^2+|W_{12}|^2}\,{}^{t}\!/_{\hbar}\right)} - 2\right|$$

Un exemple similaire montrant comment uniformiser la taille des parenthèses à l'aide de la commande *vphantom* qui permet d'introduire une hauteur fantôme d'une suite de caractères donnés, sans pour autant les imprimer à l'écran :

```
\begin{align}
\begin{split}
E&=\left(2+\vphantom{\frac{2}{\frac{2}{3}}}\right.\\
&\left.\frac{2}{\frac{2}{3}}\right)
\end{split}
\end{align}
```

$$E = \left(2 + \frac{2}{\frac{2}{3}}\right) \tag{4.14}$$

4.6.4 Aligner des équations terme à terme

La solution que j'ai trouvée est d'utiliser l'environnement *alignat* pour lequel le nombre entre accolade correspond au nombre de colonnes :

```
\begin{alignat}{4}
\dif U&= \left ( \pd{U}{S}\right )_{V,N}&&\dif S +\left ( \pd{U}{V}\right )_{S,N}%
&&\dif V +\left ( \pd{U}{N}\right )_{S,V}&&\dif N\\
&= T&&\dif S-p&&\dif V+\mu&&\dif N
\end{alignat}
```

$$\begin{aligned} \mathrm{d}U &= \left(\tfrac{\partial U}{\partial S}\right)_{V,N} \mathrm{d}S + \left(\tfrac{\partial U}{\partial V}\right)_{S,N} \mathrm{d}V + \left(\tfrac{\partial U}{\partial N}\right)_{S,V} \mathrm{d}N && (4.15)\\ &= T \quad \mathrm{d}S - p \quad \mathrm{d}V + \mu \quad \mathrm{d}N && (4.16) \end{aligned}$$

Remarque : Vous aurez sans doute remarqué que le nombre passé en argument à *alignat* ne vaut pas du tout le nombre d'esperluette (&). En effet, cet environnement est au départ prévu pour aligner n équations (tout comme *align*, sauf qu'il ne met pas des espaces gigantesques) sur une même ligne. On sépare les équations par un « & », et à chaque équation est associé un « & » supplémentaire qui doit être placé avant le symbole de relation. Chaque « & » impaire aura donc la partie de gauche justifiée à droite, et la partie de droite justifiée à gauche (ainsi, l'équation est bien accolée au symbole de relation).

Dans notre cas, pour aligner des parties d'une seule équation, on utilise la double esperluette, afin que l'on ai un alignement correct des termes.

4.6.5 Insérer du texte entre deux équations d'un même environnement

La commande *intertext* est très utile quand on veut placer du texte entre deux lignes d'équations sans sortir de l'environnement mathématique comme le montre l'exemple suivant :

$$
\begin{aligned}
A_1 &= \left| \int_0^1 (f(x) - g(x))\,\mathrm{d}x \right| + \left| \int_1^2 (g(x) - h(x))\,\mathrm{d}x \right| \\
&= \left| \int_0^1 (x^2 - 3x)\,\mathrm{d}x \right| + \left| \int_1^2 (x^2 - 5x + 6)\,\mathrm{d}x \right|
\end{aligned}
$$

Now the limits of the integrals are used

$$
\begin{aligned}
&= \left| \frac{x^3}{3} - \frac{3}{2}x^2 \right|_0^1 + \left| \frac{x^3}{3} - \frac{5}{2}x^2 + 6x \right|_1^2 \\
&= \left| \frac{1}{3} - \frac{3}{2} \right| + \left| \frac{8}{3} - \frac{20}{2} + 12 - \left(\frac{1}{3} - \frac{5}{2} + 6 \right) \right| \\
&= \left| -\frac{7}{6} \right| + \left| \frac{14}{3} - \frac{23}{6} \right| = \frac{7}{6} + \frac{5}{6} = 2\,\mathrm{FE}
\end{aligned}
\tag{4.17}
$$

Le code est le suivant :

```
\begin{equation}
\begin{split}
A_{1} & = \left| \int _{0}^{1}(f(x)-g(x))\dif x\right| +\left| \int _{1}^{2}(g(x)-h(x))
\dif x\right| \\
 & = \left| \int _{0}^{1}(x^{2}-3x)\dif x\right| +\left| \int _{1}^{2}(x^{2}-5x+6)
\dif x\right| \\
\intertext{Now the limits of the integrals are used}
& = \left| \frac{x^{3}}{3}-\frac{3}{2}x^{2}\right| _{0}^{1}+\left| \frac{x^{3}}{3}-
\frac{5}{2}x^{2}+6x\right| _{1}^{2}\\
& = \left| \frac{1}{3}-\frac{3}{2}\right| +\left| \frac{8}{3}-\frac{20}{2}+12-
\left( \frac{1}{3}-\frac{5}{2}+6\right) \right| \\
& = \left| -\frac{7}{6}\right| +\left| \frac{14}{3}-\frac{23}{6}\right| =\frac{7}{6}+
\frac{5}{6}=2\, \textrm{FE}
\end{split}
\end{equation}
```

Écrire un long texte est possible en utilisant la commande *parbox*

```
\begin{align}
a+b+c+d+ef & = g+h+i+j+k %
&\textrm{\parbox[t]{.25\linewidth}{%
this is a very long description of a formula}%
}
\end{align}
```

4.6.6 Insérer une `footnote` dans une équation

Dans l'environnement `\[ \]` ou **equation**, LaTeX n'accepte pas la syntaxe `\footnote{}`. Il faut donc marquer avec *footnotemark* comme le montre l'exemple suivant :

$$
\mathrm{c}^{\mathrm{te}} = -i\omega \frac{x}{v} \tag{4.18}
$$

On sait de plus que pour une onde on a $v = \frac{\omega}{k}$ [3]

$$
\mathrm{c}^{\mathrm{te}} = -i\omega \frac{x}{\frac{\omega}{k}} \tag{4.19}
$$

```
\begin{align}
\cte&=-i\omega \frac{x}{v}\\
\intertext{On sait de plus que pour une %
onde on a $v=\frac{\omega}{k}$\footnotemark}
\cte&=-i\omega \frac{x}{\frac{\omega}{k}}
\end{align}
\footnotetext{o\'{u} $k$ est la norme du%
vecteur d'onde}
```

[3] oú k est la norme duvecteur d'onde

4.6.7 Simplifier des termes d'une équation

Le paquet *cancel* permet d'utiliser trois commandes différentes pour simplifier les termes d'une équation ; très pratique pour représenter visuellement des simplifications dans une équation.

Le paquet *cancel* est présenté dans le paragraphe 6.3.

4.6.8 Ce qu'il ne faut pas utiliser et pourquoi

Il existe un environnement appelé *eqnarray* qui permet de mettre en page des équations, de les numéroter, et d'aligner des signes d'égalité. Cependant, celui-ci souffre de plusieurs défauts :

1. Il rajoute beaucoup d'espacement autour du symbole de relation, de façon injustifiée et incohérente avec les autres environnements.
2. Quand l'équation occupe toute la largeur de la page, *eqnarray* ne s'en rend pas compte et place le numéro d'équation en surimpression sur le texte. Les autres environnements standard, comme *equation*, ne présentent pas ce problème et placent le tag en-dessous.
3. Par ailleurs, eqnarray ne fonctionne pas correctement avec les commandes du package *amsmath*, incontournable pour composer les mathématiques. Par exemple, les commandes `\tag` et `\intertext` fonctionnent avec tous les environnements sauf *eqnarray*.

L'usage de `$$\dots $$` pour passer en mode mathématique hors-texte n'a jamais été supporté par LaTeX : c'est un héritage de TeX. Les héritages de TeX ne sont pas tous mauvais, mais celui-ci est à éviter pour (au moins) les raisons suivantes :

1. Il ne respecte pas les mécanismes de LaTeX, comme par exemple l'option fleqn de la classe standard article : cette dernière doit avoir pour effet d'aligner à gauche (au lieu de centrer) les équations hors-texte, mais les équations délimitées par `$$` restent obstinément centrées.
2. L'espacement vertical autour de l'équation est inconsistant. La plupart du temps, il sera correct, mais des comportements étranges peuvent survenir quand l'équation est précédée ou suivie de changements de paragraphes ou autres objets « complexes ».
3. Enfin, tous les packages bien faits pour LaTeX supposent que vous utilisez les constructions standard de LaTeX, et risquent donc de ne pas fonctionner avec `$$`. C'est le cas d'*amsmath*, et par exemple, de sa commande `\tag`.

Les environnement standard prévus par LaTeX pour les mathématiques hors-texte sont *displaymath*, *equation** et `\[\dots \]` : le dernier n'est guère plus long à taper que `$$` et rend par ailleurs le source plus lisible en différenciant l'ouverture de la fermeture du mode math.

Remarque : Il existe l'environnement *equation*, mais celui-ci ne permet de faire qu'une seule équation ce qui limite beaucoup les possibilités.

Par ailleurs, il est souvent pratique de faire des sous numérotations, comme le fait *subeqnarray*, mais nous l'avons vu plus haut, les environnements *eqnarray* ne doivent pas être utilisés, il faut donc utiliser autre chose à la place. Pour celà, se référer à la section [§ 4.6.2].

4.7 Mise en forme des vecteurs

Il existe principalement deux manières de noter des vecteurs. Soit un mettant une flèche sur le symbole (manière plutôt française), soit en mettant le caractère en gras (plutôt anglosaxonne). En LaTeX, après plusieurs péripéties, j'ai trouvé une manière relativement proprement de mettre en gras les vecteurs :

$$\boldsymbol{\nabla} f = \mathbf{0}$$

```
\[\boldsymbol{\nabla}f=\boldsymbol{0}\]
```

Personnellement, j'ai été habitué à noter les vecteurs avec des flèches. En écrivant à la main, on fait ça, alors pourquoi ne pas continuer par ordinateur ? Pour faire une flèche, j'utilise :

$$\overrightarrow{\nabla} f = \overrightarrow{0}$$

```
\[\overrightarrow{\nabla}f=\overrightarrow{0}\]
```

Chaque méthode a ses avantages et ses inconvénients. Pour ma part, j'ai adopté la technique suivante : j'ai définit une commance `\vect{}`. Et suivant les besoins, je pourrais définir deux versions de ma commande :

```
\newcommand{\vect}[1]{\overrightarrow{#1}}
%ou
%\newcommand{\vect}[1]{\boldsymbol{#1}}
```

Il fut un temps où la définition de ma commande était la suivante :

```
\newcommand{\vect}[1]{\overrightarrow{#1\mathstrut}}
```

`\mathstrut` permet d'aligner les différentes flèches des vecteurs entre elles. C'est vrai que c'est plus joli, mais, d'une manière générale, ça relève énormément les flèches par rapport aux symboles juste en dessous d'eux, et finalement, j'en suis venu à penser que c'est pas si esthétique que ça.

4.8 Symboles Utiles

$$\int_0^{+\infty} x^2\,\mathrm{d}x$$
$$\oint_{\mathscr{C}} \mathrm{d}\varphi$$
$$\iint_S \mathrm{d}r\,\mathrm{d}\theta$$
$$x \in \mathbb{R}$$
$$|x|$$
$$\|x\|$$
$$\mathscr{L}$$
$$\binom{n}{k}$$
$$\approx$$
$$\boxed{x^2 = 3}$$

```
\begin{align*}
\int_0^{+\infty}x^2\dif x\\
\oint_\mathscr{C}\dif \varphi\\
\iint_S\dif r \dif \theta\\
x\in\mathbb{R}\\
\abs{x}\\
\norm{x}\\
\mathscr{L}\\
\binom{n}{k}\\
\approx\\
\boxed{x^2=3}
\end{align*}
```

Å

```
\AA
```

5 Physique

Dans cette section je détaillerais les commandes à utiliser pour certaines fonctions ou notations.

5.1 Flèche du comportement asymptotique

Quand on veut mettre en page un comportement asymptotique d'une fonction, il est parfois difficile de mettre en dessous de la flèche vers quoi tend la variable. Voici la solution que j'utilise :

$$f(x) \xrightarrow[x\to 0]{} x \tag{5.1}$$

```
\begin{align}
f(x)\xrightarrow[x\to 0]{}x
\end{align}
```

5.2 Slash de Feynman

Une des solutions est d'utiliser la commande `\not` devant, mais celle-ci n'est pas satisfaisante. La commande à utiliser est `\slashed{}`, de la façon suivante :

$$(i\not{\partial} - m)\psi(\underline{x}) = 0$$

```
\begin{align*}
(i\slashed{\partial} -m)\psi(\underline{x})&=0
\end{align*}
```

Pour pouvoir utiliser cette macro, il faut ajouter le paquet *slashed*.

6 Packages

6.1 acronym

Le paquet *acronym* permet de définir, comme son nom l'indique, des acronymes tels SNCF. Il permet de définir la version courte et longue d'un acronyme, et on appelle ainsi l'un ou autre, ou les deux pour sa première apparition.

En fin de document, ou du moins, dans le document, il faut introduire l'environnement *acronym*. Celui-ci contiendra toutes les définitions d'acronymes. À l'intérieur de l'environnement, on peut utiliser la commande `\acro` pour définir un acronyme qui sera affiché dans la liste d'acronyme, ou `\acrodef` pour définir l'acronyme sans que celui-ci ne soit affiché dans la liste.

Voici un exemple :

```
\begin{acronym}[TDMA]
\acro{CDMA}{Code Division Multiple Access}
\acro{GSM}{Global System for Mobile communication}
\acro{NA}[\ensuremath{N_{\mathrm A}}]{Number of Avogadro\acroextra{ (see \S ref)}}
\acro{NAD+}[NAD\textsuperscript{+}]{Nicotinamide Adenine Dinucleotide}
\acro{NUA}{Not Used Acronym}
\acro{TDMA}{Time Division Multiple Access}
\acro{UA}{Used Acronym}
\acro{lox}[\ensuremath{LOX}]{Liquid Oxygen}%
\acro{lh2}[\ensuremath{LH_2}]{Liquid Hydrogen}%
\end{acronym}
```

Remarque : Juste après la définition de l'environnement, on peut remarquer **[TDMA]**, ceci permet de définir une largeur pour la « colonne » de la définition courte des acronymes. En effet, on voit que TDMA est le sigle le plus long de l'environnement, on le définit donc comme largeur limite pour aligner la colonne suivante à partir de cette largeur là.

Il y a de plus un paramètre optionnel qui permet de définir la définition courte de l'acronyme, si celle-ci diffère par rapport à l'identifiant de l'acronyme lorsqu'on l'appellera dans le code source du document. Concrêtement, ça sert quand le sigle est compliqué, et/ou défini en mode mathématique avec des exposants ou indices.

Pour être plus clair, j'appelle **full** le développement de l'acronyme, et **short** son sigle correspondant. Pour appeler un acronyme dans le document, on a 4 commandes :

- `\ac{acronym}` : Pour identifier l'acronyme à sa première utilisation. Cette commande renvoie le sigle suivie de sa signification (**full** + **short**).
- `\acf{acronym}` : Affiche le développement complet de l'acronyme. (**full** + **short**)
- `\acs{acronym}` : Affiche la version courte de l'acronyme, même si cette commande est utilisée avant son `\ac` correspondant (**short**)
- `\acl{acronym}` : Développe l'acronyme sans afficher la version courte. (**full**)

6.2 babel

Il est généralement utilisé par défaut pour permettre l'affichage correct des accents et cie dans la langue française et faciliter la rédaction du source en écrivant les accents directement dans le source. Il est appelé la plupart du temps ainsi :

```
\usepackage[frenchb]{babel}
```

6.2.1 Les acronymes, siglaisons et nom propres

babel propose la commande `\bsc` pour placer un bout de texte en petites capitales. Cela peut être pratique pour les acronymes, siglaisons et les noms propres.

Exemple :

La SNCF embauche Jacques DURAND. `La \bsc{sncf} embauche Jacques \bsc{Durand}.`

6.2.2 Les symboles divers

babel propose deux commandes pour afficher les degrés : `\degre` et `\degres`. La première sert pour les angles et la seconde pour les températures.

pour afficher des angles : 30°
pour afficher des temperatures : 0°C

```
pour afficher des angles : 30\degre\\
pour afficher des temperatures : 0\degres C
```

6.2.3 1er, 2e, etc.

babel fournit les commandes suivantes pour les abbréviations de premier, deuxième, etc. : `ier`, `iers`, `iere`, `ieres`, `ieme` et `iemes`. Ces commandes sont pratiques car elles utilisent les bonnes règles d'affichage des ces abbréviations.

On peux aussi utiliser les commandes `primo`, `secundo`, `tertio` et `quarto`.

1er, 1ers, 1re, 1res,
2e, 3es
1o , 2o , 3o , 4o

```
1\ier, 1\iers, 1\iere, 1\ieres,\\
 2\ieme, 3\iemes\\
\primo, \secundo, \tertio, \quarto
```

6.2.4 Écrire des grands nombres

L'option **frenchb** de l'extension *babel* fournit la commande `\nombre` qui met en forme automatiquement les grands nombres : avant et après le séparateur décimal, les chiffres sont groupés par trois et ces groupes sont séparés par une espace. Par exemple :

123 456,789 123

```
\nombre{123456,789123}
```

En particulier, hors du mode mathématique, la commande `\nombre` permet de supprimer l'espace ajoutée après la virgule.

Remarque : Il faut maintenant inclure le package *numprint* et il est de plus conseillé d'utiliser `\numprint{123456,789123}` à la place de `\nombre{123456,789123}`

6.3 cancel

Ce paquet permet de barrer de différentes manières des expressions, y compris en mode mathématique.

$$\mathrm{d}G = \cancel{T\,\mathrm{d}S} - \cancel{p\,\mathrm{d}V} + \dots \tag{6.1}$$
$$= \cancel{p\,\mathrm{d}V} \tag{6.2}$$
$$= \cancelto{0}{p\,\mathrm{d}V} \tag{6.3}$$

```
\begin{align}
\dif G&=\cancel{T\dif S}%
-\bcancel{p\dif V}+\dots\\
&=\xcancel{p\dif V}\\
&=\cancelto{0}{p\dif V}
\end{align}
```

Remarque : Les commandes *cancel*, *bcancel* et *xcancel* fonctionnent en mode math comme en mode texte. Par contre, la commande *cancelto* marche uniquement en mode mathématique.

Ce paquet permet par exemple de barrer des termes deux à deux pour qu'on voit plus facilement qui se simplifie avec qui (on peut donc simplifier au maximum trois ensembles, de manière différentes — slash normal, anti-slash, et croix) :

$$\mathrm{d}G = \cancel{T\,\mathrm{d}S} - \cancel{p\,\mathrm{d}V} + \mu\,\mathrm{d}N - \cancel{T\,\mathrm{d}S} - S\,\mathrm{d}T + \cancel{p\,\mathrm{d}V} + V\,\mathrm{d}p \tag{6.4}$$

```
\begin{align}
\dif G&=\cancel{T\dif S}-\bcancel{p\dif V}+\mu\dif N-\cancel{T\dif S}
-S\dif T+\bcancel{p\dif V}+V\dif p
\end{align}
```

Remarque : Si on veut barrer une puissance ou tout autre objet qui nécessite des parenthèses si on veut inclure plusieurs caractères, il faut utiliser un groupe. C'est à dire qu'on doit faire ça

$$2^{\cancel{2}}x = 3 \cdot \cancel{2} \tag{6.5}$$

```
\begin{align}
2^{\cancel{2}} x&=3\cdot\cancel{2}
\end{align}
```

6.4 color

Ce package est l'équivalent des commandes de couleur pour *pstricks*, ceci permettant d'utiliser les couleurs sans avoir à inclure le package *pstricks* tout entier. Ci-dessous les commandes pour définir de nouvelles couleurs :

gray taux de gris. Il est défini par un nombre décimal compris entre 0 et 1 inclu, 0 correspondant au noir et 1 au blanc.

rgb correspond aux taux de rouge 1,0,0, vert 0,1,0 et bleu 0,0,1. Chaque taux correspondant à un nombre décimal compris entre 0 et 1 inclu.

cmyk cyan 1,0,0 ;0, magenta 0,1,0 ;0, jaune 0,0,1 ;0, et noir 0,0,0 ;1. Ce sera cette fois quatre nombres décimaux compris entre 0 et 1 inclu.

La définition d'une couleur se fait alors dans l'entête (ce n'est pas une obligation) par la commande `\definecolor{nom}{modèle}{taux}`. Les différents taux sont séparés par une virgule (attention les nombres décimaux s'écrivent à l'anglaise, c'est à dire avec un « . »). Pour donner un exemple plus concret, si je veux définir une couleur à 25% de gris, j'utiliserai la ligne `\definecolor{gris25}{gray}{0.75}`

Autre exemple : j'aimerais une sorte de violet, je me dis moitié-moitié de rouge et de bleu :

```
\definecolor{violet}{rgb}{0.5,0,0.5}
```

Pour utiliser ces couleurs, on fait :

```
{\color{MyLightMagenta}This color is MyLightMagenta}
```

6.5 commath

$$\mathrm{d}x$$

```
\[\dif x\]
```

donne la différentielle de x, donc "dx"

$$\frac{\mathrm{d}^2 f}{\mathrm{d}x^2}$$

```
\[\od[2]{f}{x}\]
```

donne la dérivée d'ordre 2 de f par rapport à x, pour les dérivées premières, simplement oublier le [2] qui correspond à l'ordre de dérivation et qui par défaut vaut 1 (note :`\tod` pour affichage en texte, `\dod` pour affichage en hors texte)

$$\frac{\partial^2 f}{\partial x^2}$$

```
\[\pd[2]{f}{x}\]
```

donne la dérivée partielle d'ordre 2 de f par rapport à x, pour les dérivées partielles premières, simplement oublier le [2] qui correspond à l'ordre de dérivation et qui par défaut vaut 1 (note :`\tpd` pour affichage en texte, `\dpd` pour affichage en hors texte)

$$\frac{\partial^5 f}{\partial x^2 \partial y^3}$$

```
\[\md{f}{5}{x}{2}{y}{3}\]
```

donne la dérivée partielles d'ordre 5 de f dérivée deux fois par rapport à x et 3 fois par rapport à y (note :`\tmd` pour affichage en texte, `\dmd` pour affichage en hors texte)

$$f(\epsilon)\Big|_{\epsilon=0}$$

```
\[\eval{f(\epsilon)}_{\epsilon=0}\]
```

affiche l'évaluation de f en $\epsilon = 0$ (pour les décomposition en élément simple par exemple)

$$\left|\vphantom{\int_0^1}\right.$$

```
\[\sVert[4]\]
```

affiche une barre verticale dont la hauteur est déterminée par la valeur (entière ?) entre crochet.

Remarque : J'ai enlevé ce paquet de mes définitions car il entrait en conflit avec un paquet de toute ma liste de paquet, mais je ne sais pas lequel. En effet, à cause de ce conflit, je ne pouvais plus mettre de « : » en mode mathématique, ce qui est relativement facheux.

Pour remédier à ça, j'ai redéfini les commandes du paquets dans mon paquet à moi, et depuis, plus de soucis.

6.6 fancybox

permet de d'encadrer du texte, si on veut encadrer des formules de maths, elles ne doivent pas être en hors texte, chose que l'on peut compenser par la macro `\displaystyle` en début de formule mathématique.

texte

```
\doublebox{texte}
```

doublebox

texte

```
\fbox{texte}
```

fbox

texte

```
\shadowbox{texte}
```

shadowbox

texte

```
\ovalbox{texte}
```

ovalbox

texte

```
\Ovalbox{texte}
```

Ovalbox

6.7 fancyhdr

permet d'utiliser le pagestyle « fancy » et de mieux gérer les header et footer

l pour left, r pour right, et c pour center qui permet de gérer trois parties de head et foot avec la syntaxe `\lhead{} \chead{} \rhead{} lfoot{} cfoot{} rfoot{}`

`\thepage` entre le numéro de la page en cours et `\leftmark` affiche le titre de la section en cours (chapitre si en *report*), et `\rightmark` affiche le titre de la sous section en cours (section si en *report*).

`\fancyhead{}` permet d'effacer tout les champs du header

`\fancyfoot{}` permet d'effacer tout les champs du footer

`\fancyhf{}` permet d'effacer tout les champs des headers et footers bien sur dans l'optique de les redéfinir par la suite.

Pour afficher les titres en minuscules, il faut utiliser la syntaxe suivante :

```
\nouppercase{\leftmark}
```

Voici un exemple d'entête utilisé avec l'option *twoside* pour la classe (soit *report*, soit *article*) :

```
\fancyhead[LO,RE]{\thepage}
\fancyhead[RO]{\leftmark}
\fancyhead[LE]{\nouppercase{\rightmark}}
```

6.8 hyperref

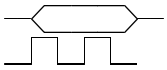
http://google.fr/ `\url{http://google.fr/}`

6.9 ifsym

Permet de tracer des diagrammes d'électroniques. Il faut ajouter certaines options parfois. dans mon cas, j'ai dû définir le package comme suit `\usepackage[electronic]{ifsym}`

```
\FallingEdge
\LongPulseLow
\PulseLow
\ShortPulseHigh
\LongPulseHigh
\PulseHigh
\RaisingEdge
\ShortPulseLow
```

Une commande pratique (la seule que je connais en fait) de ce package est `\textifsym{}`, dont les commandes sont `l m h d < > L M H D << >>`



```
\textifsym{mm<DDD>mm}\\
\textifsym{L|H|L|H|L}
```

6.10 lettre

Pour faire des lettres en français via LaTeX, il est généralement conseillé d'utiliser le package *lettre*. Suite à quelques déconvenues dans son utilisation, voici quelques astuces et un exemple type de lettre que je fais :

```
\documentclass{lettre}
\usepackage[francais]{babel}
\usepackage[autolanguage]{numprint}
\usepackage{xspace}
\usepackage{amsmath}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
```

```
\institut{moi}
\begin{document}
% Eventuellement impression d'un logo

\begin{letter}{nom du destinataire\\
adresse du destinataire\\
\numprint{00000} \textsc{Ville}}

\conc{Objet de la lettre}

\opening{Madame, Monsieur,}

CONTENU DE LA LETTRE

\closing{Dans l'attente de votre réponse, je vous prie d'agréer, Madame, Monsieur,
l'expression de mes salutations distinguées.}

% \ps{P.S. : }
\end{letter}
\end{document}
```

Le package *numprint* permet d'utiliser la commande `\numprint` qui permet de mettre en forme des nombres (en espaçant correctement les milliers par exemple).

 La commande `\numprint{dsfkjh}` génèrera une erreur. Il faut absolument que l'argument de la commande `\numprint` soit un nombre.

xspace permet de bien espacer les guillemets et les choses de ce style. *amsmath*, c'est juste parce que parfois j'utilise des maths dans mes lettres, mais pas utile pour vous si vous n'en faites pas. Les deux dernières lignes, inputenc et fontenc me permettent de dire que je rédige ma lettre en utf-8.

`\institut{nom}` permet d'aller chercher un fichier `nom.ins` qui contiendra les noms, adresses, numéro de téléphone de la personne qui envoit la lettre, c'est à dire vous, en l'occurence. Pour exemple, voici mon fichier .ins :

```
\address{M. \textsc{Nom} Prénom\\
Lieu Dit : \og machin \fg\\
\numprint{72700} \textsc{Ville}}
\telephone{00 00 00 00 00}
\email{adresse@mail.com}
\signature{M. \textsc{Nom} Prénom}
\lieu{\textsc{Ville}}
% \notelephone
\nofax
```

6.11 listings

Il s'utilise de la manière suivante :

```
\begin{lstlisting}[language={c++},title={nom du programme}]
#include <stdio.h>

int main(void)
{
int j=0, multiplication, entree=3;
  while (j <= 10)
    {
multiplication = entree * j;
printf ("%d x %d = %d\n", entree, j, multiplication);
```

```
j++;
    }
  return 0;
}
\end{lstlisting}
```

On peut modifier, via des options, l'affichage du code, les mots spéciaux et cie. Voici ce que je donne comme option à la suite de la déclaration du paquet :

```
\lstset{language=c++,columns=flexible,caption=\lstname,%
numbers=left,stepnumber=1,numberfirstline=true,%
rangeprefix=//*,rangesuffix=*//,includerangemarker=false,%
keywordstyle=\color{blue}\bfseries,identifierstyle=\color{red}\bfseries,%
commentstyle=\color{darkgray},showstringspaces=false}
\usepackage{moreverb}
```

Remarque : Il faut faire attention à ne pas trop modifier les choses. En particulier le caractère d'échappement pour pouvoir faire du LaTeX. Ça peut donner des erreurs de compilation suivant les caractères qu'on met dans l'environnement `lstlisting`.

L'extension *listings* permet de mettre du code source. On ne peut pas utiliser de caractères Unicode dans le code mise en forme par les commandes de cette extension.

Le code source est placé dans un environnement *lstlisting* ; la mise en forme stricte (y compris les espaces et les retours de ligne) est respectée, et les commandes LaTeX ne sont pas interprétées.

On définit le langage ainsi qu'un caractère d'échappement juste après l'appel de l'extension :

```
\usepackage{listings}
\lstset{language=TeX,
   basicstyle=\ttfamily\small,
   columns=flexible,
   escapechar=
```

Dans l'exemple ci-dessus, on indique :
- que l'on écrit du TeX ou du LaTeX, ce qui permettra à l'extension de reconnaître les mots-clefs et d'appliquer une mise en forme spécifique ;
- que le code sera en police Teletype de corps plus petit que le texte ;
- que les colonnes sont « flexibles », c'est-à-dire que l'écriture peut être un plus compacte au détriment éventuellement de l'alignement vertical ;
- que le caractère d'échappement est « + ».

Le texte compris entre deux caractères d'échappement est interprété par LaTeX, par exemple dans

```
{\large Texte en corps plus grand}
```

```
\begin{lstlisting}
{\large +\emph{Texte en corps plus grand}+}
\end{lstlisting}
```

la séquence `{\large` et le `}` final seront imprimés tels quels, tandis que le `\emph{Texte en corps plus grand}` sera interprété (on aura donc le contenu du bloc en italique). Il faut évidemment choisir un caractère d'échappement qui ne sera pas dans le code.

On peut mettre du code au sein d'un texte, avec la commande `\lstinlinec\dots c`. Le c indique un caractère qui marque le début et la fin du code ; il peut être choisi arbitrairement, mais ne doit évidemment pas faire partie du code. Par exemple

Pour mettre du texte en emphase, on utilise la commande `\lstinline-\emph-`.

Un grand nombre de langages sont disponibles : Pascal, Fortran, Basic, C, C++, Ada, Scilab, HTML, Java, ? On peut indiquer des variantes, par exemple

```
\lstset{language=[95]Ada}
\lstset{language=[Visual]C++}
\lstset{language=[77]Fortran}
```

On peut aussi indiquer les options lors de l'appel de l'environnement *lstlisting* :

```
\begin{lstlisting}[language=XML,escapechar=?]
\dots
\end{lstlisting}
```

6.11.1 code source matlab

Tout d'abord, il faut télécharger le paquet
http://www.mathworks.com/matlabcentral/files/8015/mcode.sty.

Ensuite, voici la marche à suivre :

1. Placer le fichier **mcode.sty** soit dans le même dossier que le fichier .tex, soit dans l'arborescence des paquets LaTeX (voir §10.2).
2. Ajouter dans le préambule

   `\usepackage[options]{mcode}`

   les options à inclure sont :
   - bw si vous comptez imprimer le document. (la mise en valeur se fera par du formattage — comme mise en gras ou en italique et niveaux de gris)
   - numbered si vous voulez que les lignes de codes soient numérotées
   - framed si vous voulez encadrer votre code source.
   - final qui n'affiche pas du tout le code source, me demandez pas à quoi ça sert...
3. incluez le code matlab soit en incluant directement le fichier .m

   `\lstinputlisting{/path_to_mfile/yourmfile.m}`

   ou en plaçant le code source dans un environnement *lstlisting*

```
\begin{lstlisting}
% Example Matlab code for calculating hypotenuse
% § $c = \sqrt{a^2+b^2}$ §
a = 3;
b = 4;
c = sqrt(a^2+b^2);
\end{lstlisting}
```

Remarque : Dans le code source, vous pouvez afficher du code LaTeX en utilisant

`§ YOUR LATEX CODE §`

6.12 mathrsfs

$\mathscr{ABC}$ `\(\mathscr{ABC}\)`

$\mathscr{L}$ du laplacien `\(\mathscr{L}\) du laplacien`

6.13 moderncv

Pour l'instant, c'est le meilleur paquet que j'ai trouvé pour faire un *CV*. Il existe pas mal de solutions, mais aucune ne me convenait, celle-ci se rapproche le plus de ce que je souhaitais.

6.14 moreverb

Il semble y avoir un conflit entre le package *moreverb* et le package *example*. C'est plutôt ennuyeux. Je n'ai pas à ce jour trouvé la solution. Pour avoir de nouveaux environnements verbatim. Le premier utile est `verbatimtab` qui permet de conserver l'indentation des lignes, ce qui est très pratique pour afficher du code source.

```
\begin{verbatimtab}
  while (j <= 10)
    {
multiplication = entree * j;
printf ("%d x %d = %d\n", entree, j, multiplication);
j++;
    }
\end{verbatimtab}
```

Le deuxième est `listing` qui en plus de conserver l'indentation, affiche les numéros de lignes. Il s'utilise de la manière suivante :

```
\begin{listing}{1}
#include <stdio.h>

int main(void)
{
int j=0, multiplication, entree=3;
  while (j <= 10)
    {
multiplication = entree * j;
printf ("%d x %d = %d\n", entree, j, multiplication);
j++;
    }
  return 0;
}
\end{listing}
```

On doit spécifier en argument de l'environnement `listing` le nombre pour le début de la numérotation des lignes.

6.15 natbib

Voir la section [§ 8.4]

6.16 paralist

Permet d'utiliser l'environnement *inparaenum* qui affiche une liste sans retour à la ligne entre les items, pour afficher une liste dans un paragraphe.

atome seul, et proche du zero absolu, (i) test ; (ii) test 2. On a vu, au travers de.

```
atome seul, et proche du zero absolu,
\begin{enuminline}
\item test ;
\item test 2.
\end{enuminline}
On a vu, au travers de.
```

Je n'ai pas réussi à définir mon propre environnement depuis le début, donc en attendant, j'utilise ce paquet pour définir mon environnement :

```
\newenvironment{enuminline}{\begin{inparaenum}[(i)]}{\end{inparaenum}}
```

6.17 pstricks

permet entre autre d'inclure de la couleur en LaTeX par les commandes **black**, **darkgray**, **gray**, **lightgray** et **white**, ainsi que les couleurs **red**, **green**, **blue**, **cyan**, **magenta** et **yellow** qui sont prédéfinies dans le package. À utiliser ainsi :

Bonjour

```
{\red Bonjour}
```

Il faut inclure la commande de couleur et le texte que l'on veut dans cette couleur dans un groupe pour que la définition de la couleur soit locale, le cas échéant, le reste du texte sera de la même couleur à moins qu'une autre couleur soit définie entre temps.

pour définir des couleurs, on procède comme suit :

`\newrgbcolor{ma couleur}{0 0 1}` où successivement sont définis les taux de **rouge**, **vert** et **bleu**

`\newgray{darkgray}{.25}` où 0 est **noir** et 1 est **blanc**

`\newcmykcolor{hercolor}{.5 1 0 .5}` où on a **cyan**, **magenta**, **jaune** et **noir** dans l'ordre.

6.18 shapepar

Permet de mettre en forme des paragraphes, par exemples avec une forme de cœur, d'étoiles ou d'autres formes. On utilise ces commandes de la même façon, par exemple :

mon
pa-
ra-
graphe,
c'est a dire un seul paragraphe,
et il met pas tres bien en
forme les retours
a la ligne, le
mieux etant un
para- graphe
ki- lo-
me- trique

```
\starpar{mon paragraphe, c'est a dire
 un seul paragraphe, et il met
pas tres bien en forme les retours
 a la ligne, le mieux etant
un paragraphe kilometrique}
```

Par défaut dans le paquet, on a ces formes là :

```
\squarepar
\circlepar
\CDlabel
\diamondpar
\heartpar
\starpar
\hexagonpar
\nutpar
```

On peut trouver d'autres formes, par exemple un chandelier, le drapeau du québec, une sorte de smiley, qui sont disponibles avec le paquet dans des fichiers annexes que je ne détaille pas ne pouvant pas inclure ces fichiers dans mon **.pdf**, mais ça se trouve relativement facilement sur le net, il est même possible que vous trouviez d'autres formes.

Il est de plus possible de créer ses propres formes, notament avec un patch que l'on applique à *Xfig* mais ça a pas l'air simple, pas forcément à jour, donc je ne sais pas comment ça fonctionne, mais je sais que ça existe.

7 Présentation avec LaTeXet beamer

Pour faire une présentation, j'utilise la classe beamer.

Voici une partie de mon préambule pour faire une présentation

```
\documentclass{beamer}
\usetheme{Darmstadt}
\usefonttheme[onlylarge]{structurebold}
\setbeamerfont*{frametitle}{size=\normalsize,series=\bfseries}
\setbeamertemplate{navigation symbols}{}

\useackage{tikz}
\usetikzlibrary{arrows}
\newcommand*{\newblock}{}%pour pouvoir faire des bibliographies
```

Ensuite, pour faire une diapo, on utilise

```
\begin{frame}
contenu de la diapo
\end{frame}
```

Pour mettre la table des matières au début, j'utilise

```
\begin{frame}
\frametitle{Table des matières}
\tableofcontents
\end{frame}
```

Il est possible d'afficher du texte petit à petit, et de faire plein d'autres choses. je vous conseille d'aller voir la documentation de *beamer* qui est très complète.

8 Faire une bibliographie

8.1 Construire la bibliographie

le fichier .bib est de la forme :

8.2 Exemple

```
@InBook{cohen2,
author = {Cohen-Tannoudji, Claude and Diu, Bernard and Laloë, Franck},
editor = {Hermann},
title = {Mécanique Quantique II},
chapter = {},
publisher = {},
year = {1998},
OPTpages = {}
}
```

8.2.1 Pour un article

```
@Article{étiquette,
title = {le titre de la publi},
author = {liste des auteurs},
journal = {journal},
volume* = {volume},
year = {année},
pages* = {pages concernées}
month = {mois}
note* = {note}
}
```

les commandes avec une astérisque ne sont pas obligatoires.

8.2.2 Pour un article dans un proceeding de conférence

```
@InProceedings{étiquette,
author = {liste des auteurs},
title = {titre},
booktitle = {titre du proceeding de la conférence},
year = {année}
}
```

8.2.3 Pour une thèse

```
@PhDThesis{étiquette,
title = {titre},
author = {auteur},
school = {Université},
year = {année}}
```

8.2.4 Pour un livre

```
@book{étiquette,
author = {liste des auteurs},
title = {titre},
year = {année},
publisher= {éditeur}}
```

8.2.5 Pour un site web

```
@online{Doe:2009:Online,
author = {Doe, Ringo},
title = {This is a test entry of type {@ONLINE}},
month = jun,
year = {2009},
url = {http://www.test.org/doe/}
}
```

8.3 Afficher la bibliographie

J'ai récemment souhaité faire une bibliographie, et j'ai trouvé ça très compliqué. Je passe l'idée de faire une bibliographie intégrée au fichier **.tex** car je trouve ça peu pratique. Par contre, je vais essayer de détailler un peu plus l'autre approche. ça consiste à créer un fichier **.bib**. Dans ce fichier, on regroupe les articles ou autres parutions ainsi qu'une référence. Ce que je n'ai pas compris de suite, c'est que de lui même, il n'affiche que les articles qui sont référencés dans le texte, ce qui fait que comme je ne citais aucun article, ma bibliographie était vide à la fin.

Pour mettre une bibliographie, il faut donc rentrer les commandes suivantes :

```
\bibliographystyle{style}
\bibliography{nom-biblio}
\nocite{*}
```

où `style` est à remplacer par :
- alpha
- plain
- unsrt
- abbrrv

J'utilise par défaut **plain** mais j'avoue ne pas avoir essayé les autres. **nom-biblio** prendra pour valeur le nom de votre fichier .bib sans l'extension. Il est à noter que le fichier doit être dans le même dossier que le fichier **.tex**. il est possible je crois de donner des chemins relatifs mais pour éviter les confusions vu que je ne suis pas trop au courant, je n'en parlerais pas ici.

La commande `\nocite{*}` permet d'afficher tout les ouvrages qui n'ont pas étés cités.

Remarque : L'extension *tocbibind* permet de faire figurer la bibliographie dans la table des matières.

8.4 Paquet natbib

Ce paquet permet de citer sous la forme « auteur-année ». Il existe une page très bien faite qui parle des possibilités de natbib à cette adresse : http://merkel.zoneo.net/latex/natbib.php?lang=fr

Le minimum à savoir, c'est à dire ce dont je me sers, ce sont les commande suivantes :
- `\citet{gomes2005ocl}` qui met en forme de la façon suivante :
  `Gomes et al. (2005)`
- `\citep{gomes2005ocl}` qui met en forme de la façon suivante :
  `(Gomes et al., 2005)`

Les styles de bibliographies disponibles sont :
- plainnat
- abbrvnat
- unsrtnat

Ma suite de commande pour appeler la bibliographie est :

```
\bibliographystyle{plainnat}%style
\bibliography{stage}%nom du fichier .bib
\nocite{*}%rajoute les références non citées dans l'article.
```

Remarque : Pour faire fonctionner natbib avec *beamer*, se référer à la section [§ 9.3].

9 Faire une presentation avec LaTeX : beamer

9.1 Les bases

9.2 Animations élémentaire

Deux types d'animations sont très utiles et complémentaires. La première `\onslide<n>{contenu}` permet d'afficher l'ensemble `contenu` (qui peut être une image, du texte, ou tout autre boite LaTeX) sur le slide numéro n. En écrivant `<n->`, le contenu s'affichera à partir du slide n jusqu'à la fin de l'« animation » pour le frame courant. La chose importante est que `\onslide<n>{contenu}` réservera l'espace dans le frame, laissant un espace blanc en attendant que le contenu apparaisse au slide voulu.

Pour ne pas réserver d'espace, et ainsi pouvoir remplacer une chose par une autre, on utilise :

```
\only<n>{contenu}
```

9.3 Faire fonctionner natbib

Insérer immédiatement après `\bibliography{}` :

```
\def\newblock{}
```

10 LaTeX sous GNU/Linux

10.1 Installation

Pour installer LaTeX sous GNU/Linux, il faut tout d'abord installer les commandes qui serviront à compiler vos code source. Pour celà, il y a *tetex*, et *tex-live* qui sont les deux principaux. Tetex n'étant plus mis à jour, je conseille tex-live. Vous pouvez l'installer via *synaptic*, ce qui vous permettra de compléter les 3 paquets que j'installe ici par des modules plus spécifiques.

Sinon, en ligne de commande, ceci devrait suffire à avoir ses premiers textes :

```
sudo apt-get install texlive texlive-lang-french texlive-latex-extra
```

Une fois fait, il faut maintenant un éditeur pour nous faciliter la vie. Il y en a pas mal, comme *TeXmaker* qui existe aussi sous windows. Il y a aussi *Winefish*, un éditeur LaTeX basé sur *Bluefish* qui est pas mal. Mais j'ai opté pour ma part pour *Kile* que je trouve très bien fait, très complet et que l'on peut installer comme ceci :

```
sudo apt-get install kile
```

10.2 Astuces

Pour mettre à jour la base de donnée des packages quand on a rajouté manuellement un **.sty**, il suffit de taper en console **sudo texhash**

À noter que le chemin où se trouvent les packages, et donc, là où on en rajoute manuellement est : **/usr/share/texmf-texlive/tex/latex/**

10.3 LaTeXavec Inkscape

Pour avoir l'option Effet>Rendu>Formule LaTeX... il faut installer *pstoedit* en plus d'inkscape. L'option apparaît automatique au démarrage suivant.

11 Problèmes de Compilation

11.1 Erreurs

- En mode mathématique, un deux points « : » pose problème et pour ce faire, on doit mettre `\text{ : }` pour résoudre le problème de compilation.
- dans la table des matières, si on veut mettre dans les titres des parties en mode mathématique, on doit le faire à l'aide de `\)la formule\)`, sinon, il y a un problème à la compilation.
- Une erreur peut provenir des environnements, je viens justement avec ce texte d'en avoir un parce que le `\begin{example}a_0^2\end{example}` était sur une seule ligne. En le scindant en 3 lignes, la compilation s'effectue sans problèmes.
- Si à la compilation on a une erreur 70, ça signifie que l'on essaye d'inclure quelque chose dont le chemin est erroné. Typiquement, j'ai cette erreur quand j'essaye d'inclure des images que je n'ai pas placé au bon endroit, ou pas créé du tout.
- Il est impossible d'utiliser `$\overrightarrow{k}$` dans un *caption*

11.1.1 TeX capacity exceeded, sorry [input stack size=5000]

Elle peut survenir pour plusieurs raisons. La raison pour laquelle je l'avais était que j'avais mis une commande `\verb|[XYZ]STYLE|` dans un *caption*

11.2 Avertissements

11.2.1 binôme de newton

La commande `{n \choose k}` est obsolète et génère une erreur car elle utilise une macro qui ne devrait plus être utilisée. À la place, il faut utiliser `\binom{n}{k}`

11.2.2 \headheight is too small

```
\headheight is too small (13.0pt):Make it at least 24.1638pt.
```

Pour remédier à cet avertissement, il suffit de rentrer la ligne suivante dans le préambule :

```
\setlength{\headheight}{24.1638pt}
```

11.2.3 Hfootnote has been referenced but does not exist

```
{Hfootnote.2} has been referenced but does not exist, replaced by a fixed one
```

Ceci est sans doute dû au fait que le `\footnote{}` est défini à l'intérieur d'un environnement, d'une formule mathématique ou autre. Afin de remédier au problème, il faut utiliser `\footnotemark` et `\footnotetext`.

Au lieu d'utiliser :

```
\begin{exemple}
blabla\footnote{note de bas de page}
\end{exemple}
```

il faut faire :

```
\begin{exemple}
blabla\footnotemark
\end{exemple}
\footnotetext{note de bas de page}
```

L'avertissement devrait disparaître, et votre note de bas de page devrait apparaître.

11.2.4 Math dans les titres

Il est possible d'inclure des maths dans les titres, celà dit, ça génèrera un avertissement car même si écrire des maths dans les titres est possible, ce n'est pas le cas pour la table des matières. Pour remédier à ça, il faut spécifier un titre à mettre dans la table des matières qui ne contient pas de maths, comme le montre l'exemple suivant :

```
\subparagraph*[\dots à une distance d]{\dots à une distance $d$}
```

11.2.5 Problème de césure : l'avertissement Overfull \hbox

```
Overfull \hbox (5.128pt too wide) in paragraph at lines 916-918
```

Celà signifie que LaTeX ne savait pas comment couper un mot et plutôt que de faire n'importe quoi, il a préféré ne rien faire. Pour lui dire où couper le mot, il suffit d'insérer la commande `\-` au milieu du mot, sans espace ni avant, ni après la commande.

11.3 références erronées

J'avais depuis quelques temps des erreurs de numérotation des figures. Il se trouve que je ne mettais pas le `\label{}` au bon endroit. En effet, le `\label{}` doit être placé à la fin du `\caption{}`. D'une manière générale, voici un exemple de *figure* et de *subfig*

```
\begin{figure}[htbp]
\centering
\includegraphics[width=0.9\linewidth]{montage.pdf}
\caption{Schéma du montage}\label{fig:montage}
\end{figure}
```

```
\begin{figure}[htbp]
  \centering
  \subfloat[Image de l'aiguille référence]{\label{fig:aiguille}
\includegraphics[width=0.45\textwidth]{aiguille.png}}
\hspace*{0.05\linewidth}
  \subfloat[Image d'un plasma par ombroscopie]{\label{fig:plasma-ombroscopie}
\includegraphics[width=0.45\textwidth]{plasma-ombroscopie.png}}
  \caption{Exemples d'images obtenues par ombroscopie}\label{fig:ombroscopie}
\end{figure}
```

11.4 Entête qui n'a pas les mêmes marges

Il faut faire attention à définir les marges avant d'appeler le paquet *fancyhdr*. Le cas échéant, les marges pour le style de page *fancy* seront les anciennes marges.

Index