Formation doctorale LaTeX

Yannis Haralambous (Télécom Bretagne)

Plan du cours

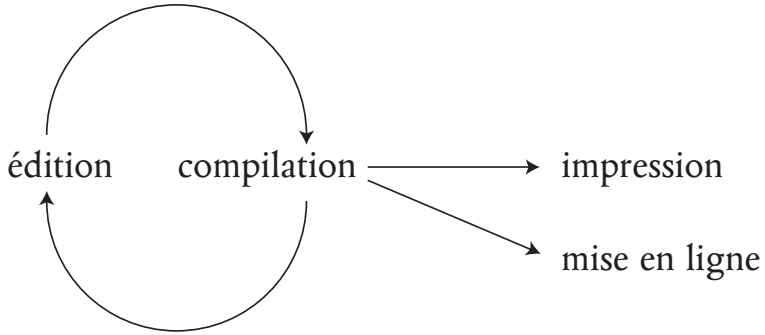
- Philosophie, généralités, morphologie, syntaxe, commandes de base, bon nombre de choses utiles
- Index, bibliographie
- Présentations
- Mathématiques (micro- et macro-typographie)
- Pages Web avec formules (MathJax)
- Diagrammes commutatifs, lettrage de figures
- LaTeX avancé, les coulisses de TeX
- Personnalisation de style

Philosophie

- TEX est un des plus anciens logiciels libres (1978) et le premier à avoir massivement déclenché la créativité de ses utilisateurs ;
- TEX est un typographe dans la machine, un langage de programmation, une syntaxe pour écrire les mathématiques, et pour certains : une « religion » ;
- LATEX est une surcouche du langage de programmation TEX ;
- on utilise le paradigme de la programmation : édition du source, compilation, débogage ;
- que l'on peut traduire en jargon d'imprimerie : préparation de copie, composition, correction d'épreuves ;
- sauf que là où il y avait un auteur, un éditeur, un imprimeur, et un correcteur, aujourd'hui il n'y a qu'une seule personne : l'« utilisateur ».

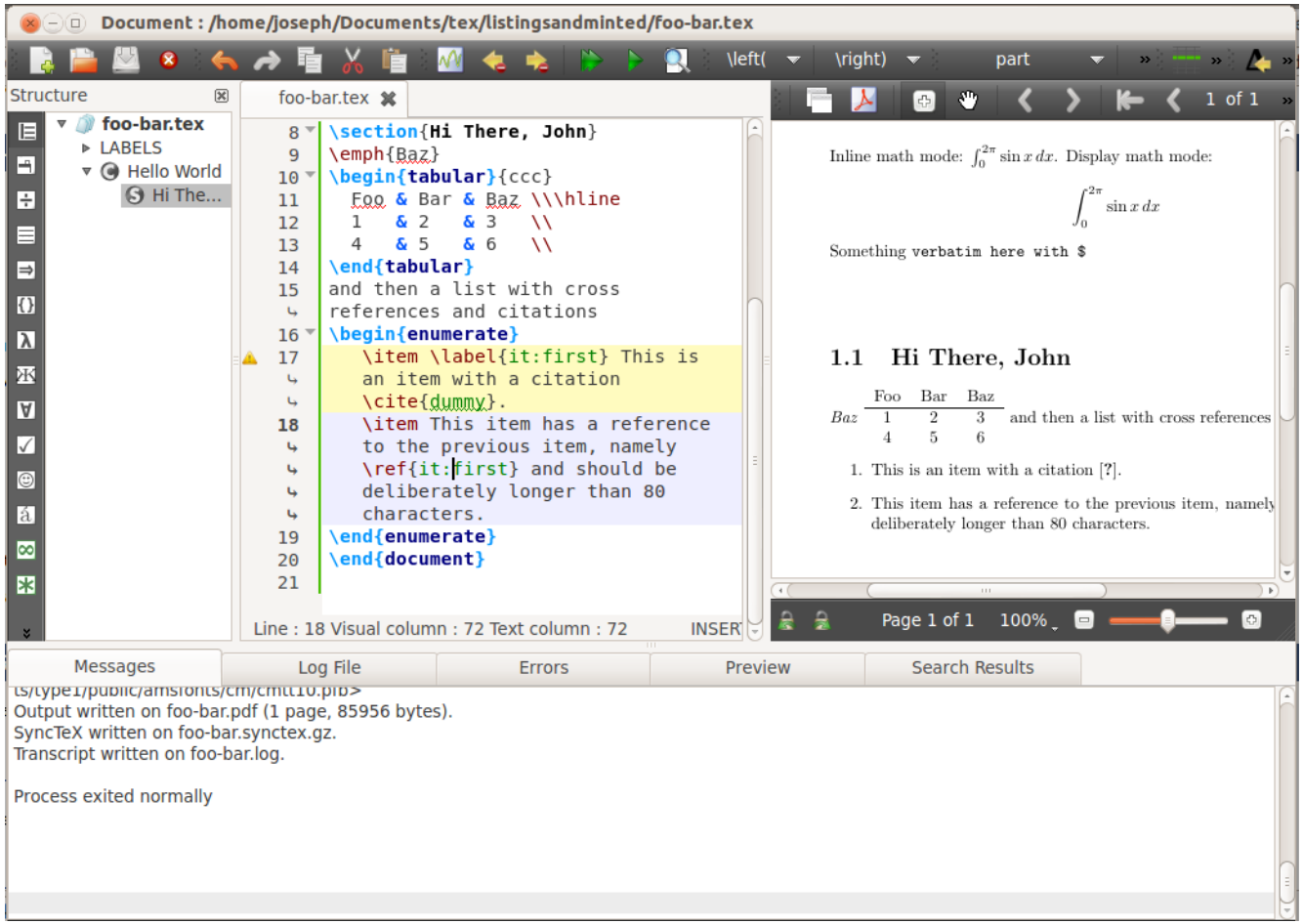
Généralités — mode opératoire

- LaTeX est un système de préparation de document, basé sur le langage de programmation TeX ;



- mode opératoire sous Unix ou assimilé : on écrit le code LaTeX dans un fichier `toto.tex`, et puis plusieurs approches :
  1. on compile : `latex toto`
     et on obtient `toto.dvi`. On convertit en PostScript :
     `dvips -o toto.ps toto`
     et éventuellement en PDF : `ps2pdf toto.ps`
  2. ou alors on compile directement `pdflatex toto`
  3. ou alors `xelatex toto` si le document comporte des écritures non-latines.

Il existe aussi des IDE...



TEXShop (Mac), TEXnicCenter (Windows), TEXstudio, etc., cf.

http://tex.stackexchange.com/questions/339/latex-editors-ides

Un projets, plusieurs fichiers

- Le nom de fichier devient le nom de projet ;
- l'extension de nom de fichier indique sa nature : `.tex`, `.log`, `.dvi`, `.ps`, `.pdf`, `.aux`, `.sty`, `.cls`, `.idx`, `.ind`, `.bib`, `.bbl` et tant d'autres...
- dans la suite on s'intéressera avant tout aux fichiers `.tex` qui contiennent le code source de notre document.

Conventions du mode texte, orthotypographie

- Plusieurs blancs = un blanc ;
- plusieurs lignes blanches = une ligne blanche = un changement de paragraphe ;
- on écrit `--`, `---`, `<<`, `>>`, ` ``, ` '', `!``, `?`` pour –, —, « , », “, ”, ¡, ¿ ;
- on écrit `~` pour le blanc insécable ;
- dans un document français composé en France, en Belgique ou en Suisse (package **babel**, option **francais**), on laissera un blanc devant la double ponctuation ainsi qu'avant et après le tiret long [**babel** insère un blanc insécable devant le deux-points et une espace fine insécable dans les autres cas] ;
- dans un document français composé au Canada, ou un document anglais (package **babel**, option **english**), on ne laissera pas de blanc devant la double ponctuation ou autour du tiret long.

L'antislash \

- Si l'arabe est la langue du *dhad*, TEX est le langage de l'antislash ;
- que les anglophones appellent *backslash* ;
- et les Québecois *contre-oblique* ;
- et qui est arrivé dans nos claviers grâce au/à cause du langage ALGOL.

Modes, caractères spéciaux

- Mode *texte* (par défaut, contrairement à C, Perl, etc.) ;
- mode *mathématique* (entre dollars) : `x, $x$` donnent : x, x ;
- mode *commentaire* (précédé d'un %) ;
- mode *verbatim* (environnement `verbatim`) ;
- les caractères spéciaux sont : `\` et `#` (commandes), `$` (mode math), `%` (commentaires), `^` (exposants), `_` (indices), `&` (tableaux), `{` et `}` (groupes). Pour les obtenir en tant que glyphes, on écrira : `\textbackslash`, `\#`, `\$`, `\%`, `\^{}`, `\_`, `\&`, `\{`, `\}`.

Commandes, groupes

- Un *nom de commande* commence par une contre-oblique `\` ;
- il est composé soit d'un seul caractère (quelconque) soit d'une ou plusieurs lettres (entre a et z, A et Z) ;
- Piège du débutant : le blanc qui suit un nom de commande est un « faux blanc » (il ne sert qu'à indiquer la fin de nom de commande), ainsi `\TeX␣est␣bien` donne « TEXest bien » ;
- un *groupe* est un bloc de texte entre accolades : {`un groupe`}. TEX considère un groupe comme un seul objet ;
- les commandes peuvent être suivies d'*arguments* : ceux-ci sont des caractères isolés ou des groupes : `\emph{n'est-ce pas}` donne « *n'est-ce pas* » ;
- le premier argument d'une commande peut être *optionnel*, on l'écrira entre crochets : `\section[titre court]{titre long}`.

Environnements

- Un *environnement* est une paire de commandes \begin{*mot-clé*} et \end{*mot-clé*}. Les environnements doivent être imbriqués ;
- les commandes de début d'environnement peuvent être suivies d'arguments :
  `\begin{tabular}{|c|c|c}`

Structure de document LaTeX typique

```
\documentclass[11pt]{article}
\usepackage[francais,english]{babel}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\begin{document}
... corps du document ...
\end{document}
```

- La commande `\documentclass` contient la classe du document. Les classes les plus courantes sont : `book`, `article` et `report`. Les options les plus importantes sont : `10pt`, `11pt`, `12pt`, `a4paper`, `landscape`, `final`, `draft`, `oneside`, `twoside`, `openright`, `openany`, `onecolumn`, `twocolumn`, `notitlepage`, `titlepage`.

Packages

- La commande **\usepackage** charge des *packages*. Les plus importants sont : **babel** (avec option **francais**, **english**, etc.), **graphics** (avec option **dvips**), **color** (avec option **dvips**), **fontenc** (avec option **T1**), **inputenc** (avec options **utf8**, **latin1**, **applemac**, etc.), **multicol**, **ifthen**, **fancyhead**, **array**, **amsmath**, **amsfonts**, etc.
- Un admin sous TEXLive, peut installer un package en écrivant :

  ```
  sudo tlmgr install nom_du_package
  ```

  ainsi que mettre à jour tous les packages :

  ```
  sudo tlmgr update --self
  sudo tlmgr update --all
  ```

- les classes sont contenues dans des fichiers **.cls**, les packages (ou « feuilles de style ») dans des fichiers **.sty**. Parfois on trouve des « options de classe » dans des fichiers **.clo**.

La page de titre, le sommaire

- On décrit la page de titre dans le préambule ;
- `\title` pour le titre ;
- `\author` pour le(s) nom(s) d'auteur (quand il y en a plusieurs, on sépare par `\and`) ;
- `\date` pour donner une date explicite (ou vide), sinon TEX mettra la date courante ;
- une fois entrés dans le corps du document, `\maketitle` ;
- et `\tableofcontents` pour obtenir un sommaire.

```
\title{Traité de pifométrie}
\author{Daniel Schmilblick\thanks{Université de Plouzané}
\and
Michel Tartenpion\thanks{ENS de Dresseurs de Phoques}}
\date{9 mars 2009}
\begin{document}
\maketitle

\tableofcontents
```

Structure de document

- des parties \part{*Titre de la partie*} (pas pour les articles) ;
- des chapitres \chapter{*Titre du chapitre*} (pas pour les articles) ;
- des sections \section{*Titre de la section*} ;
- des sous-sections \subsection{*Titre de la sous-section*} ;
- des sous-sous-sections \subsubsection{*Titre de la sous-sous-section*}.

Toutes ces commandes prennent un argument optionnel destiné au titre courant et au sommaire.

Commandes usuelles avec argument(s)

- Divisions hiérarchiques du document : `\part`, `\chapter`, `\section`, `\subsection`, `\subsubsection`, `\paragraph`, `\subparagraph` ;
- notes de bas de page : `\footnote` (mais aussi `\footnotetext` et `\footnotenumber`) ;
- enrichissement : `\emph`, `\textbf`, `\textit`, `\textsl`, `\textsf`, `\texttt`, `\textsc`, `\textul` (avec package `underlin`) ;
- couleur (package `color`) : `\textcolor{`*nom*`}{...}` ou `\textcolor[rgb]{`*r,v,b*`}{...}` ou `\textcolor[cmyk]{`*c,m,j,n*`}{...}`.

Commandes usuelles sans argument

- On limite, le cas échéant, leurs effet en les mettant dans un groupe ;
- taille des caractères : `\tiny`, `\scriptsize`, `\footnotesize`, `\small`, `\normalsize`, `\large`, `\Large`, `\LARGE`, `\huge`, `\Huge` ;
- la malheureuse ligature franco-française `\oe` `\OE` délaissée par ISO 8859-1 et ménacée d'extinction par sa non-présence sur les claviers ;
- `\TeX` (TEX), `\LaTeX` (LATEX), `\euro` avec package `eurosym` (€), `\today` (25 mai 2014) ;
- `\,` (espace fine), `\frenchspacing` (pas d'espacement étendu après un point de fin de phrase), `\\` (passage à la ligne forcé), `\noindent` (début de ligne sans retrait), `\quad` (un cadratin de blanc), `\qquad` (un double cadratin) ;
- `\ldots` pour les point de suspension dans un texte anglais (Attention ! on écrira simplement `...` dans un texte français).

Environnements usuels : citations, listes

- `quote` pour les citations sans retrait ;
- `quotation` pour les citations avec retrait ;
- `center` pour les blocs de texte centrés ;
- `itemize` pour les listes non énumératives. Chaque entrée de la liste est précédée de la commande `\item{}`, ou `\item[`*marque*`]` ;
- `enumerate` pour les listes énumératives, même utilisation que `itemize` ;
- `description` pour les « glossaires » : on met le lemme dans l'argument optionnel de `\item`.

Tableaux 1/6

- on utilise l'environnement `tabular` avec, en guise d'argument, un *format* ;
- on décrit le tableau ligne par ligne, chaque ligne étant structurée comme le format l'indique ;
- le format comporte une lettre `c`, `l` ou `r` par cellule selon qu'elle est centrée, justifiée à gauche ou justifiée à droite, un `|` pour chaque filet vertical et un `||` pour chaque filet vertical double ;
- dans la description de chaque ligne, les contenus des cellules sont séparés par des `&`, et à la fin de la ligne on met un `\\`, suivi d'un (ou de deux) `\hline` si l'on souhaite un filet horizontal au dessus de la ligne suivante ;

Tableaux 2/6

- Exemple :

```
\begin{tabular}{|c||c|}\hline
A & B\\\hline\hline
C & D\\\hline
\end{tabular}
```

- donnera :

A	B
C	D

Tableaux 3/6

- Pour avoir des cellules multi-colonnes, on écrira à la place de ces cellules :
  `\multicolumn{`*n*`}{`*format*`}{`*contenu*`}`
- Exemple :

```
\begin{tabular}{|c|c|c|c|}\hline
A & B & C & D\\\hline
\multicolumn{3}{|c|}{E} & F\\\hline
\end{tabular}
```

- donnera :

A	B	C	D
E			F

Tableaux 4/6

- Pour avoir des filets horizontaux partiels on écrira à la place de `\hline` un ou plusieurs :
  `\cline{`*début*`-`*fin*`}`
- Exemple :

```
\begin{tabular}{|c|c|c|c|}\hline
A & B & C & D\\\cline{1-2}\cline{4-4}
E & F & G & H\\\hline
\end{tabular}
```

- donnera :

A	B	C	D
E	F	G	H

Tableaux 5/6

- Pour avoir des cellules qui contiennent des paragraphes entiers de texte (charger le package `array` !) on écrira dans le format `m{`*`largeur`*`}`, `p{`*`largeur`*`}` ou `b{`*`largeur`*`}`, selon que la cellule paragraphe doit être centrée, justifiée vers le haut ou justifiée vers le bas, par rapport aux cellules mono-lignes ;
- Exemple :

```
\begin{tabular}{|c|p{2cm}|m{2cm}|b{2cm}|}\hline
Une ligne & Quelques mots & Quelques mots
& Quelques mots \\\hline\end{tabular}
```

- donnera :

Une ligne	Quelques mots de texte	Quelques mots de texte	Quelques mots de texte

Tableaux 6/6

- Pour remplacer le blanc inter-cellules par autre chose, on écrira `@{...}` dans le format, par exemple : `@{}c@{}c@{}` ;
- pour obtenir des cellules paragraphe d'égale largeur, on utilisera la lettre `X` (package `tabularx`) ;
- pour obtenir des cellules multi-lignes, on utilisera le package `multirow` ;
- pour obtenir des tableaux multi-pages, on utilisera le package `supertab` ou le package `longtable` ;
- pour obtenir des cellules coloriées, on utilisera le package `colortab` ;
- Théorème : *Tous ces packages sont compatibles.* (Démonstration laissée au lecteur.)

Références croisées

- `\label{`*mot-clé*`}` pour créer un marqueur ;
- `\ref{`*mot-clé*`}` pour obtenir un numéro du subdivision, de note, de figure, de table, etc. ;
- `\pageref{`*mot-clé*`}` pour obtenir un numéro de page ;
- le package *showkeys* permet de montrer tous les mots-clés utilisés ;
- le package *hyperref* transforme les appels de référence croisée en liens hypertexte dans les documents PDF.

Multicolonnage

- Package `multicol` ;
- `\begin{multicols}{nombre}` ;
- on dispose des dimensions `\columnsep` et `\columnseprule` ;
- et de la commande `\columnbreak`.

Longtemps, je me suis couché de bonne heure. Parfois, à peine ma bougie éteinte, mes yeux se fermaient si vite que je n'avais pas le temps de me dire : ≪ Je m'endors. ≫ Et, une demi-heure après, la pensée qu'il était temps de chercher le sommeil m'éveillait ; je voulais poser le volume que je croyais avoir dans les mains et souffler ma lumière ; je n'avais pas cessé en dormant de faire des réflexions sur ce que je venais de lire, mais ces réflexions avaient pris un tour particulier ; il me semblait que j'étais moi-même ce dont parlait l'ouvrage : une église, un quatuor, la rivalité de François I^{er} et de Charles-Quint.

Objets flottants

- Une invention de TEX ;
- les rôles changent : on fait des suggestions, le logiciel décide...
- les environnements `figure` et `table` permettent d'obtenir des objets flottants. Dans leur argument optionnel on peut indiquer `h` (ici), `t` (haut de page), `b` (bas de page), suivi également d'un point d'exclamation. Le placement des figures est un des points faibles de LATEX ;
- avec le package `float` on peut aussi utiliser `H` (utilisé tout seul) ;
- la commande `\caption` sert à indiquer la légende de la figure flottante ou du tableau flottant.
- ATTENTION : ne pas confondre la commande d'objet flottant avec le contenu de l'objet (`figure` n'est pas une figure, c'est un cadre invisible qui peut contenir une figure... ou autre chose).

TEX seulement : inclusion d'images EPS ou PS

- une image EPS contient normalement une ligne du type

  `%%BoundingBox: 0 0 3903 2695`

  Ces quatre nombres sont les coordonnées PostScript de l'enveloppe de l'image (*bounding box*), dans l'espace de coordonnées de la page PostScript.
- pour include une image EPS `toto.eps` de *bounding box* x_b, y_b, x_t, y_t dans un document LATEX on écrira (avec le package `graphics`) :
  `\includegraphics[`x_b`,`y_b`][`x_t`,`y_t`]{toto.eps}`
  [Attention : pas de point dans les noms de fichier !]
- avec `\includegraphics*` on cache tout ce qui se trouve à l'extérieur de la bounding box ;

pdfTEX seulement : incl. d'images PNG ou PDF

- *pdfTEX* ne reconnaît que le format PDF et quelques formats bitmap, comme JPEG et PNG ;
- pour inclure des images EPS dans des documents PDF générés par pdfTEX, il faut d'abord les convertir en PDF, à l'aide de *epstopdf* ;
- ATTENTION : utiliser *epstopdf* et non pas *ps2pdf*, pour avoir les bonnes marges.

Manipulation d'image

- Pour mettre l'image à l'échelle on écrira `\scalebox{`*f*`}{...}` ;
- ou `\resizebox{`*largeur*`}{`*hauteur*`}{...}`, où l'un des deux arguments peut être `!` pour garder les proportions.
- ou `\rotatebox{`*angle*`}{...}`, pour tourner l'image.
- Peut être appliqué à 

qu'à des images.

Changer de langue

- le package **babel** admet un certain nombre d'options : breton, bulgarian, croatian, czech, dutch, english, finnish, francais, german, polytonikogreek, hebrew, magyar, italian, norsk, polish, portuges, romanian, russian, serbian, slovak, slovene, spanish, swedish, turkish, ukenglish, ukrainian, etc. ;
- on met la langue dominante en dernier ;
- on utilise `\selectlanguage{langue}` pour changer de langue ;
- on utilise `\foreignlanguage{langue}{phrase}` si c'est juste une phrase ;
- on utilise l'environnement **hyphenrules** si l'on veut changer uniquement les règles de césure ;
- on dispose d'un test `\iflanguage{langue}{vrai}{faux}`.

Césure

- les règles de césure de chaque langue sont appliquées automatiquement ;
- utiliser `\-` pour forcer la césure potentielle ;
- utiliser `\discretionary{ck}{k-}{k}` pour les césures spéciales ;
- utiliser `\hyphenation{Ha-ra-lam-bous Bech-stein}` pour des mots qui arrivent souvent dans le texte ;
- pour déboguer lire le fichier log : il fournit une liste des motifs de césure chargés, les activer explicitement par la commande `\language`*i* ;
- pour éviter la césure utiliser `\language99` ;
- si un mot refuse de se couper mettre `\hskip0pt` ou `\penalty10000\hskip0pt` devant.

Écrire en arabe

```
\documentclass[12pt]{article}
\usepackage{fontspec,xunicode,xltxtra,arabxetex}
\defaultfontfeatures{Mapping=tex-text}
\newfontfamily\arabicfont[Script=Arabic]{Scheherazade.ttf}
\usepackage[arabic]{babel}
\begin{document}
\selectlanguage{arabic}\begin{arab}[utf]
```

أساسًا، تتعامل الحواسيب مع الأرقام فقط، وتخزّن الأحرف والمحارف الأخرى بإعطاء رقم لكل واحد منها.

```
\end{arab}\end{document}
```

Voir `https://www.writelatex.com/1100146gwfcsb#/2596462/` pour un exemple sous WriteLaTeX. La fonte peut être récupérée ici : `http://scripts.sil.org/cms/scripts/page.php?site_id=nrsi&id=Scheherazade`

Écrire en chinois

```
\documentclass[UTF8,nofonts]{ctexart}
\setCJKmainfont{WenQuanYi Zen Hei}
\setCJKsansfont{WenQuanYi Zen Hei}
\setCJKmonofont{WenQuanYi Zen Hei Mono}
\usepackage{fontspec}
\begin{document}
```

基本上，计算机只是处理数字。它们指定一个数字，来储存字母或其他字符。

```
\end{document}
```

Voir `https://www.writelatex.com/1103482vjjjkp#/2605956/` pour un exemple sous WriteLaTeX.

Écrire dans d'autres langues

```
\documentclass{article}
\usepackage{fontspec}
\setmainfont{Lucida Grande}
\usepackage[vietnamese,russian,greek,english]{polyglossia}
\begin{document}
```

Về cơ bản, máy tính chỉ xử lý số. Máy tính lưu trữ chữ và các ký tự khác bằng cách quy định một con số cho mỗi ký tự. По своей природе компьютеры могут работать лишь с числами. И для того, чтобы они могли хранить в памяти буквы или другие символы, каждому такому символу должно быть поставлено в соответствие число. Οἱ ἠλεκτρονικοὶ ὑπολογιστές, σὲ τελικὴ ἀνάλυση, χειρίζονται ἁπλῶς ἀριθμούς. Ἀποθηκεύουν γράμματα καὶ ἄλλους χαρακτῆρες ἀντιστοιχώντας στὸ καθένα τους ἀπὸ ἕναν ἀριθμό (ὀνομάζουμε μία τέτοια ἀντιστοιχία κωδικοσελίδα).

```
\end{document}
```

Le code informatique

- L'environnement **verbatim** et la commande **\verb** (dont le délimiteur d'argument peut être choisi librement) ;
- **\begin{multicols}{nombre}** ;
- le package **fancyvrb** permet d'ajouter des numéros de ligne, d'inclure des commandes dans le code, etc. :

```
\begin{Verbatim}[numbers=left,commandchars=\|\[\]]
Voici |emph[ce que] ça donne...
... quand |emph[on utilise] fancyvrb
\end{Verbatim}
```

```
1 Voici ce que ça donne...
2 ... quand on utilise fancyvrb
```

Index

- On met la commande **\makeindex** dans le préambule et on utilise le package *makeidx* ;
- on met la commande **\printindex** là où l'index doit être composé ;
- pour indexer *toto* on écrit **\index{***toto***}** ;
- pour indiquer un certain nombre de pages parlant de ≪toto≫, on écrira **\index{toto|(}** au début et **\index{toto|)}** à la fin ;
- pour appliquer la commande **\machin** au numéro de page, on écrira **\index{toto|machin}** ;
- les sous-entrées se séparent avec des points d'exclamation : **\index{***France***!***Finistère***!***Brest***}** ;
- pour trier une entrée de manière spéciale (par exemple ≪ β ≫ trié en tant que *beta*) écrire **\index{beta@β}**. Le faire aussi pour les accents : **\index{degenere@dégénéré}** ;
- ensuite lancer le programme **makeindex** ***nom de fichier*** et recompiler. Pour déboguer consulter les fichiers **.idx** et **.ind**.

Bibliographie

- On se crée une base de entrées bibliographiques `mabase.bib` écrites en BibTEX ;
- chaque entrée a une référence `toto`. On écrira `\cite{`*`toto`*`}` dans le texte ;
- ou `\cite[`*`p. 123`*`]{`*`toto`*`}` quand il y a un complément d'information ;
- on peut réunir plusieurs références `\cite{`*`toto,tata,titi`*`}` ;
- et on dispose aussi de `\nocite` pour inclure des entrées dans la biblio, de manière transparente (cf. `\nocite{*}`) ;
- enfin, on écrira les deux lignes suivantes pour obtenir la bibliographie :
  `\bibliographystyle{plain}`
  `\bibliography{mabase1,mabase2,mabase3}`
- ensuite on lance le programme `bibtex` *`nom de fichier`* et on recompile deux fois.

Exemple de fichier BibTeX

```
@string{YH = "Yannis Haralambous"}
@book{Haralambous:2003-fetc,
  year = {2003},
  url = {http://www.oreilly.fr/catalogue/284177273X},
  title = {{Fontes \& codages}},
  publisher = {O'Reilly France},
  author = YH,
  isbn = "978-2-84177-273-5",
}
```

On écrira donc `\cite{Haralambous:2003-fetc}` pour obtenir :

Références bibliographiques

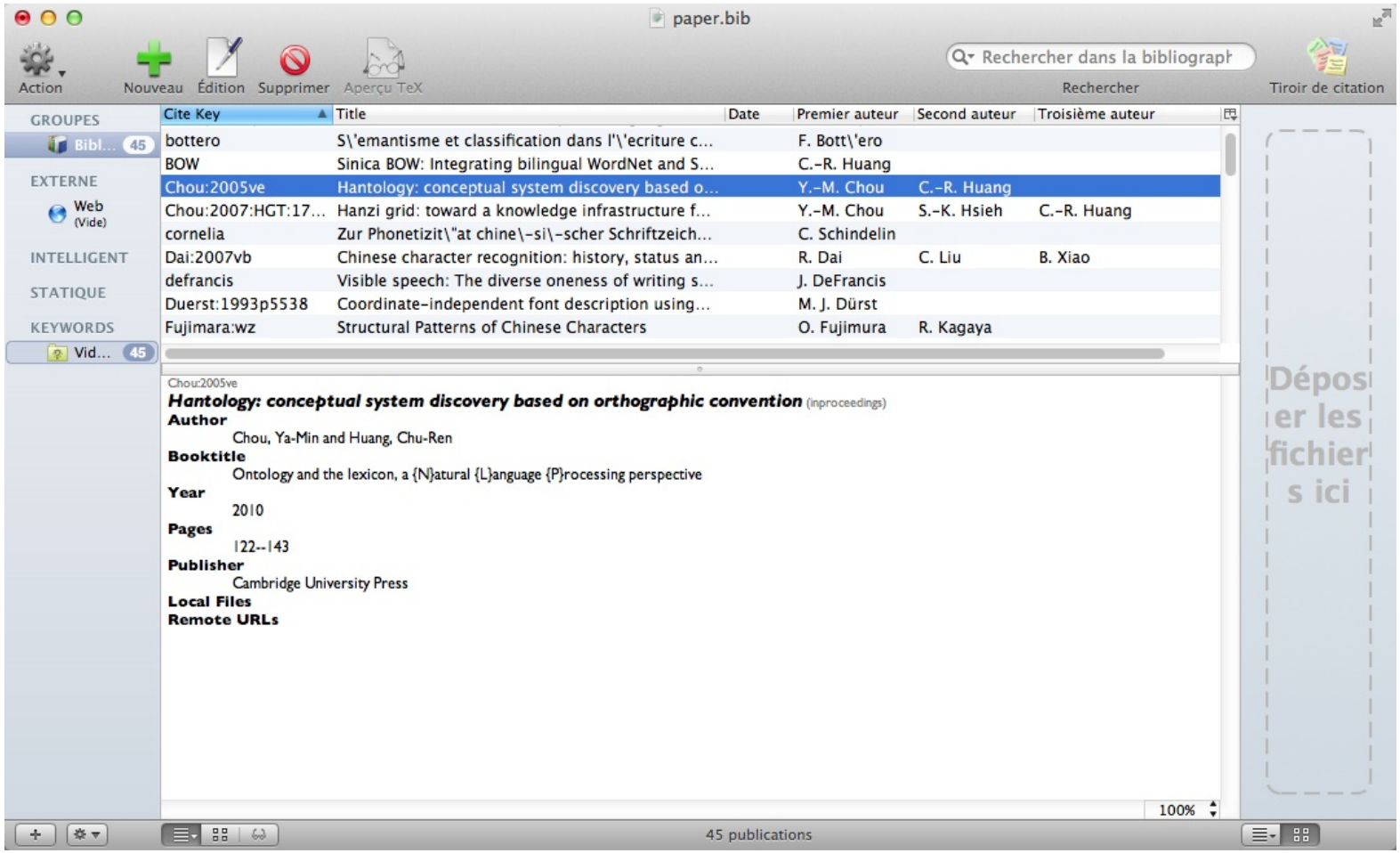
[1] Haralambous, Y., *Fontes & codages*, O'Reilly France, 2003.

Types d'entrées BibTEX

- `article` : author, title, journal, year ;
- `book` : author/editor, title, publisher, year ;
- `booklet` : title ;
- `inbook` : author, title, chapter/pages, publisher, year ;
- `inproceedings` : author, title, booktitle, year ;
- `manual` : title ;
- `masterthesis` : author, title, school, year ;
- `misc` : ;
- `phdthesis` : author, title, school, year ;
- `proceedings` : title, year ;
- `techreport` : author, title, institution, year ;
- `unpublished` : author, title, note.

(Nous n'avons indiqué que les champs obligatoires.)

Logiciel BibDesk (sur Mac)



Le package *beamer*

- il faut installer les packages *beamer*, *pgf*, *xcolor* ;

Le package *beamer*

- on demande la classe de document `beamer` ;
- on choisit un thème graphique :
  `\usetheme{tb}`

Le package *beamer*

- on choisit un thème graphique :
  `\usetheme{tb}`
- chaque transparent est un environnement `frame` :

```
\begin{frame}{Le package \emph{beamer}}
\begin{itemize}
\item il faut installer les packages \emph{beamer},
\emph{pgf}, \emph{xcolor} ;
...
\end{itemize}
\end{frame}
```

Le package *beamer*

- chaque transparent est un environnement `frame` :

```
\begin{frame}{Le package \emph{beamer}}
\begin{itemize}
\item il faut installer les packages \emph{beamer},
\emph{pgf}, \emph{xcolor} ;
...
\end{itemize}
\end{frame}
```

Couches, modes, transitions, animations

- pour obtenir différentes couches on fait suivre les `\item` d'un `<1>` ou `<2-3>` ou `<4->` ;
- pour indiquer si une commande doit être exécutée en mode « transparents » ou en mode « polycopié » on utilise
  `\mode<beamer>{...}`
  `\mode<handout>{...}`
- pour obtenir des transitions ou pour insérer des animations il faut utiliser le package `multimedia` ;
- la transition d'un transparent à l'autre peut être décrite par une commande du type `\transdissolve` suivie d'une indication de couche et éventuellement un argument optionnel (`duration=` secondes, `direction=` 0, 90, 180, 270) ;
- on peut inclure des animations (AVI, MPEG, etc.) à l'aide de la commande `\movie`, exemple :
  `\movie[width=3cm,height=2cm,poster]{}{monanim.avi}`

Le mode mathématique

[Dans la suite on utilisera systématiquement les packages `amsmath`, `amssymb` et `amsfonts`]

- Les formules mathématiques font partie intégrante de TEX. Pour écrire une formule on passe en *mode mathématique* ;
- en mode mathématique les blancs ne comptent plus et les lignes blanches sont interdites ;
- on inclut les formules mathématiques courtes par des dollars : `$x^2$` (x^2) et les formules en vedette par des doubles dollars : `$$x=1$$`

$$x = 1$$

Mathématiques de base

- En mode mathématique les lettres deviennent des variables et sont donc composées en italiques (« mathématiques ») ;
- les opérateurs `+`, `=`, `-`, `<`, `>`, la ponctuation, les parenthèses et les crochets s'utilisent normalement : $1 < x + 1 = y - 2 < 2$;
- les exposants et indices s'obtiennent à l'aide de `^` et `_` : `$x^{12}_{ab}$` donnera x^{12}_{ab} ;
- pour les tenseurs, utiliser `\phantom`. Pour obtenir ${x^{ijk}}_{\ell mn}$ au lieu de $x^{ijk}_{\ell mn}$, on écrira :

  `$x^{ijk}_{\phantom{ijk}\ell mn}$`.

Mathématiques de base

- pour les racines on écrira `\sqrt[n]{2}` ($\sqrt[n]{2}$), le cas par défaut étant la racine carrée ;
- pour les flèches avec fonction on écrira `\xleftarrow` et `\xrightarrow` (la fonction au-dessus est l'argument obligatoire, celle au-dessous, l'argument optionnel) ;
- pour les fractions on écrira `\frac{x}{y}` ($\frac{x}{y}$) ;
- pour les fractions continues on utilisera `\cfrac`, exemple : `\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\dotsb}}}}`, qui donnera : $\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cdots}}}}$

Mathématiques de base

- on a six types de points de suspension : \dotsc ... pour les points entourés de virgules [attention, les virgules ne sont pas incluses !], \dotsb $\cdots$ pour les points d'opérateurs binaires, \dotsm $\cdots$ pour les points de multiplication, \dotsi $\cdots$ pour les points de signes d'intégrales, \dotso ... pour les autres cas horizontaux, \vdots $\vdots$ pour les points verticaux ;
- les lettres grecques s'obtiennent par des commandes \alpha ... \omega, \Gamma ... \Omega ;
- utiliser \varepsilon ε plutôt que \epsilon ϵ ;
- on a deux thétas (\theta θ, \vartheta ϑ), deux phis (\phi ϕ, \varphi φ), deux rhos (\rho ρ, \varrho ϱ) ;
- par contre, éviter d'utiliser \varsigma ς, \varkappa $\varkappa$, \varpi ϖ et \digamma $\digamma$.

Mathématiques de base

- Les lettres « rondes », les ensembles de nombres et les gothiques s'obtiennent resp. par `\mathcal` ($\mathcal{A}$, $\mathcal{B}$, ...), `\mathbb` ($\mathbb{Q}$, $\mathbb{R}$, ...), `\mathfrak` ($\mathfrak{A}$, $\mathfrak{B}$, ...). Attention : pas de bas de casse rondes ou « blackboard » ;
- pour les symboles gras, on utilise `\boldsymbol` (à consommer avec modération) ;
- pour le « prime », le « second », etc. on utilise des quotes : `a' b'' c'''` donnera $a'b''c'''$ (sauf quand on voudra combiner le prime avec un exposant : `x^{\prime\,n}` donnera x'^{n}.

Symboles mathématiques

- On dispose de plusieurs centaines de symboles mathématiques : des opérateurs \pm ±, \circ ∘, \cap ∩, \otimes ⊗, \star ⋆,...
- des relations \leq ≤, \parallel ∥, \subset ⊂, \in ∈, ...
- des flèches \to →, \longrightarrow ⟶, \iff ⟺ , \Longleftrightarrow ⟺, \uparrow ↑, \nearrow ↗, ...
- des symboles divers \aleph ℵ, \Re ℜ, \nabla ∇, \exists ∃, \forall ∀, \infty ∞, \ell ℓ, \emptyset ∅, \partial ∂, ...
- des délimiteurs \{ ({), \langle ⟨, | (|), \| (‖) ...

Les « grands » opérateurs 1/2

- On dispose de certains opérateurs de taille variable : \sum $\sum$ (à ne pas confondre avec \Sigma Σ), \prod $\prod$ (à ne pas confondre avec \Pi Π), \coprod $\coprod$, \int $\int$, \oint $\oint$, \iint $\iint$, \iiint $\iiint$, \iiiint $\iiiint$, \bigoplus $\bigoplus$, \bigotimes $\bigotimes$, ...
- ces opérateurs sont composés de manière différente selon le contexte : dans une ligne de texte, `\sum_{i=0}^{n}` donnera $\sum_{i=0}^{n}$, alors que dans une formule en vedette, il donnera

$$\sum_{i=0}^{n}$$

Les « grands » opérateurs 2/2

- On peut forcer la taille d'affichage et le placement des exposants/indices avec `\displaystyle` et `\textstyle` (À.C.A.M.) :

```
\displaystyle\sum_{i=0}^{n}\int_0^{\infty}f_i =
\textstyle\sum_{i=0}^{n}\int_0^{\infty}f_i
```

$$\sum_{i=0}^{n}\int_0^{\infty} f_i = \textstyle\sum_{i=0}^{n}\int_0^{\infty} f_i$$

- lorsqu'un grand opérateur nécessite des « exposants » ou « indices » de plusieurs lignes, on utilise `\substack`. Exemple : `\sum_{\substack{i=0\\j=0}}^{\infty}` donnera

$$\sum_{\substack{i=0\\j=0}}^{\infty}$$

Les « grands » délimiteurs

- Les délimiteurs peuvent prendre une taille quelconque ;
- pour cela on utilise les commandes `\left` et `\right` suivis du délimiteur en question : `\left(`, `\right\}`, `\left\langle`, etc. Exemple :

$$\left\langle \frac{\frac{1+x}{1-x}}{\frac{1+y}{1-y}} \right\rangle$$

- les commandes `\left` et `\right` vont de paire ; si, pour une raison quelconque, on ne veut qu'un seul grand délimiteur, on écrira un point à la place du délimiteur `\right.` ;
- le package `yhmath` propose des délimiteurs encore plus grands.

Les accents mathématiques 1/2

- Alors qu'en mode texte on écrit directement les lettres accentuées (à condition d'utiliser le package `inputenc`), en mode mathématique on a des commandes pour cela : `\acute` ($\acute{a}$), `\grave` ($\grave{a}$), `\hat` ($\hat{a}$), `\tilde` ($\tilde{a}$), `\bar` ($\bar{a}$), `\vec` ($\vec{a}$), `\dot` ($\dot{a}$), `\ddot` ($\ddot{a}$), `\dddot` ($\dddot{a}$), `\ddddot` ($\ddddot{a}$), `\breve` ($\breve{a}$), `\ring` ($\mathring{a}$, package `yhmath`) ;
- les accents peuvent se combiner, mais alors on écrira les commandes avec une lettre majuscule : `\Hat{\Hat{A}}` donnera $\hat{\hat{A}}$;
- quand l'expression à accentuer est plus large, on utilise `\widehat` ($\widehat{ABC}$), `\widetilde` ($\widetilde{ABC}$). Dans `yhmath` il existe également un `\wideparen` ($\wideparen{ABC}$), `\widetriangle` ($\widetriangle{ABC}$) [1] et même un `\widering` ($\widering{ABC}$) ;

1. Les Américains écriront plutôt $\triangle ABC$ pour le triangle $\widetriangle{ABC}$ et $\angle ABC$ pour l'angle $\widehat{ABC}$.

Les accents mathématiques 2/2

- les flèches extensibles au-dessus ou au-dessous des expressions s'obtiennent par `\overrightarrow`, `\underrightarrow`, etc. : $\overrightarrow{ABCDE}$;
- les barres extensibles s'obtiennent avec `\overline` et `\underline` ;
- les accolades extensibles s'obtiennent avec `\underbrace` et `\overbrace`. Exemple :

$$\underbrace{a+\overbrace{b+c}^{\text{3 termes}}+d}_{\text{3 termes}}$$

- (dans l'exemple ci-dessus, le bricolage est autorisé : on utilisera des commandes très sympathiques comme `\rlap`, `\phantom`, `\smash`)
- pour placer des symboles au-dessus ou au-dessous d'autres symboles, on utilisera `\overset` et `\underset`. Exemple : `\overset{*}{x}` produira $\overset{*}{x}$.

Les opérateurs textuels 1/2

- Il est de très mauvais goût [2] d'écrire $logx$ pour le logarithme de x. L'écriture correcte est $\log x$;
- on dispose d'un grand nombre de commandes pour les opérateurs textuels : `\log` log, `\cos` cos, `\max` max, `\sup` sup, `\lim` lim,...
- certains de ces opérateurs ont un comportement particulier vis-à-vis des exposants et indices, ainsi `\lim_{n\to\infty}` donnera : $\lim_{n\to\infty}$.

2. Et c'est un des indicateurs des mauvais TEXistes... donc, méfiez-vous !

Les opérateurs textuels 2/2

- Pour définir de nouveaux opérateurs textuels on utilise `\DeclareMathOperator` (dans le préambule !) qui prend en premier argument la commande ainsi définie, et en deuxième argument la chaîne de texte. Exemple : `\DeclareMathOperator{\tg}{tg}` ;
- attention : dans le deuxième argument de `\DeclareMathOperator` on est tout de même en mode mathématique. Les éventuels accents s'écrivent alors comme des accents mathématiques. Exemple : `\DeclareMathOperator{\Det}{D\acute{e}t}` ;
- pour obtenir un opérateur dont le comportement est similaire à celui de « lim », on utilise la même commande suivie d'un astérisque. Exemple : `\DeclareMathOperator*{\argmax}{arg\,max}` ;
- une solution de facilité pour ne pas passer par ces commandes est `\mathrm`.

Le texte dans les formules

- Pour inclure du texte [3] dans des formules, on se sert de `\text` ;
- exemple : `$x^{x^{\text{beaucoup...}}}$` qui donnera $x^{x^{\text{beaucoup...}}}$;
- à l'intérieur de `\text` on peut utiliser des accents et toute commande du mode texte ;
- attention : `\text` ne participe pas à l'espacement des formules mathématiques. Il convient d'ajouter des espaces : `$x\text{ ou }y$` (x ou y) est meilleur que `$x \text{ou} y$` ($x\text{ou}y$) ;
- le piège : `\text` hérite du contexte textuel courant. Si on est en italiques à l'extérieur de la formule, on le sera également dans `\text`. Donc, ne pas utiliser `\text` pour les opérateurs mathématiques, mais uniquement lorsqu'il s'agit d'un commentaire de la formule...

3. NE PAS CONFONDRE texte et opérateurs textuels : on écrira ≪$f(x) = x$ et ainsi de suite≫ en texte, mais ≪$\text{grad}(X)$≫ en romain mathématique puisque c'est un opérateur.

Les matrices

- Les matrices s'écrivent comme des tableaux : on sépare les cellules par `&` et les lignes par `\\` ;
- l'environnement de matrice dépend du délimiteur : `pmatrix` pour les matrices, `vmatrix` pour les déterminants, `bmatrix` avec des crochets, `Bmatrix` avec des accolades, `Vmatrix` avec des doubles barres verticales. `yhmath` définit un `amatrix`, avec des angles $\langle\ \rangle$. `\begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix}` donne :

$$\begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix};$$

- il existe aussi un environnement `matrix` sans délimiteurs, ainsi qu'un `smallmatrix` sans délimiteurs et de plus petite taille, comme ici : $\left(\begin{smallmatrix}a&b\\c&d\end{smallmatrix}\right)$.

Micro-typographie fine 1/2

- Pour TEX il existe huit types de symboles mathématiques : les symboles ordinaires, les opérateurs binaires, les symboles d'ouverture (d'expression), les symboles de fermeture, la ponctuation, les grands opérateurs, les relations, et les alphanumériques ;
- ainsi, =, qui est une relation, ne sera pas espacée comme . qui est un signe de ponctuation, comparer $a = b$ et $a.b$;
- pour forcer le comportement d'un symbole vis-à-vis de l'espacement, on dispose des commandes `\mathord`, `\mathbin`, `\mathopen`, `\mathclose`, `\mathpunct`, `\mathop`, `\mathrel` et `\mathalpha` ;
- exemple : pour utiliser le point comme opérateur binaire, on écrira `a\mathbin{.}b`, ce qui donne : $a \mathbin{.} b$.

Micro-typographie fine 2/2

- Parfois il convient d'espacer un peu « manuellement ». Ainsi, on laissera dans une intégrale un peu de blanc entre l'expression à intégrer et l'opérateur d'intégration : $\int f(x)\,dx$. Pour cela on dispose de certaines commandes qui laissent du blanc : `\,` (espace fine), `\:` (espace moyenne), `\;` (espace un peu plus large) ;
- Exemples : $\int f(x)\,dx$ est meilleur que $\int f(x)dx$;
- $(1-x)(1-x^2)(1-x^3)\,(1-y)(1-y^2)(1-y^3)$
  est meilleur que
  $(1-x)(1-x^2)(1-x^3)(1-y)(1-y^2)(1-y^3)$;
- parfois on veut, au contraire, rapprocher deux expressions : on y parvient en utilisant la commande `\!` (ÀCAM).

Formules en vedette

- La manière « propre » d'obtenir des formules en vedette sous LaTeX est d'utiliser l'environnement `equation`. Celui-ci va automatiquement numéroter les formules (compteur `equation`) :

$$ax^2 + bx + c = 0 \iff x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}; \tag{1}$$

- la numérotation se fait normalement à droite, sauf si on charge le module `amsmath` avec l'option `leqno` ;
- on peut forcer un numéro de formule à l'aide de `\tag{...}`, il sera placé entre parenthèses (selon la feuille de style). En utilisant `\tag*{...}` on obtient une expression sans parenthèses ;
- l'environnement `equation*` produit une formule en vedette non numérotée.

Équations trop longues 1/2

- Si l'équation est trop longue, on peut la couper en plusieurs lignes dans un environnement `multline`. La première ligne sera composée fer à gauche, la dernière fer à droite, et les autres lignes centrées. Le numéro se placera sur la dernière ligne si on le compose à droite, ou sur la première si on le compose à gauche. Exemple :

$$\begin{aligned}\Gamma[E(f(X)|Y),g(Y)] &= E[E(f(X)|Y)g(Y)] \\ &= E[f(X)\,g(Y)] = E[f(X)\,E(g(Y)|X)] \\ &= \Gamma[f(X),E(g(Y)|X)] \end{aligned} \quad (2)$$

```
\begin{multline}
\Gamma[E(f(X)|Y),g(Y)]=E[E(f(X)|Y)g(Y)]\\
= E[f(X)\,g(Y)]=E[f(X)\,E(g(Y)|X)]\\
=\Gamma[f(X),E(g(Y)|X)]
\end{multline}
```

Équations trop longues 2/2

- Si en plus on souhaite un alignement on utilisera l'environnement `split`, à l'intérieur d'un `equation`, en plaçant un `&` devant les endroits à aligner :

$$\begin{aligned}\Gamma[E(f(X)|Y),g(Y)] &= E[E(f(X)|Y)g(Y)]\\ &= E[f(X)\,g(Y)]\\ &= E[f(X)\,E(g(Y)|X)]\\ &= \Gamma[f(X),E(g(Y)|X)]\end{aligned} \tag{3}$$

```
\begin{equation}\begin{split}
\Gamma[E(f(X)|Y),g(Y)]&=E[E(f(X)|Y)g(Y)]\\
&= E[f(X)\,g(Y)]\\ &=E[f(X)\,E(g(Y)|X)]\\
&=\Gamma[f(X),E(g(Y)|X)]
\end{split}\end{equation}
```

Groupes d'équations 1/3

- Pour grouper plusieurs équations on utilise l'environnement **gather**. L'effet est le même que pour une suite d'**equation**, mais l'espacement vertical entre les équations est meilleur :

$$x^2 + y^2 + z^2 = 1 \tag{4}$$
$$2x^3 + 3y^3 + 4z^3 = 0 \tag{5}$$

```
\begin{gather}
x^2+y^2+z^2=1\\
2x^3+3y^3+4z^3=0
\end{gather}
```

Groupes d'équations 2/3

- L'environnement `align` permet de grouper plusieurs équations sur la même ligne, et des aligner sur plusieurs lignes. On utilisera $2n-1$ caractères `&` pour n équations (un pour chaque point d'alignement et un entre deux équations). Exemple :

$$f(x) = 1 \qquad f(y) = 1 \qquad f(z) = 1 \tag{6}$$
$$f'(x) = 0 \qquad f'(y) = 0 \qquad f'(z) = 0 \tag{7}$$

et ainsi de suite, jusqu'à :

$$f^{(n)}(x) = 0 \qquad f^{(n)}(y) = 0 \qquad f^{(n)}(z) = 0 \tag{8}$$

Groupes d'équations 3/3

```
\begin{align}
f(x)&=1 & f(y)&=1 & f(z)&=1\\
f'(x)&=0 & f'(y)&=0 & f'(z)&=0\\
\intertext{et ainsi de suite, jusqu'à :}
f^{(n)}(x)&=0 & f^{(n)}(y)&=0 & f^{(n)}(z)&=0
\end{align}
```

- la commande **\intertext** permet d'insérer du texte entre deux formules sans perdre les points d'alignement.

Mini-groupes d'équations 1/2

- Les environnements **gather** et **align** utilisent toute la largeur de la page. Pour produire des mini-groupes on peut utiliser **gathered** et **aligned**, à l'intérieur d'un **equation** ou autre environnement de formule en vedette :

$$\vec{v_1} = \vec{v_2} \iff \left\{ \begin{array}{r} 2(a+b) = c \\ x = y + z + w \end{array} \right\} \iff \sqrt{|t_1 \wedge t_2|} = 0; \qquad (9)$$

```
\begin{equation}
\vec{v_1} = \vec{v_2} \iff \left\{
\begin{aligned}2(a+b)&=c\\x&=y+z+w\end{aligned}
\right\} \iff \sqrt{|t_1\wedge t_2|}=0 ;
\end{equation}
```

Mini-groupes d'équations 2/2

- Pour produire une liste de cas on utilise `cases`, à l'intérieur d'un `equation` :

$$f(x) = \begin{cases} \frac{1}{x} & \text{si } x > 0 \\ +\infty & \text{si } x = 0 \\ -\frac{1}{x} & \text{si } x < 0 \end{cases} \tag{10}$$

```
\begin{equation}
f(x) = \begin{cases}\frac 1x & \quad\text{si $x > 0$}\\
                    +\infty & \quad\text{si $x = 0$}\\
                    -\frac 1x & \quad\text{si $x < 0$}
       \end{cases}
\end{equation}
```

Théorèmes, lemmes, etc. 1/2

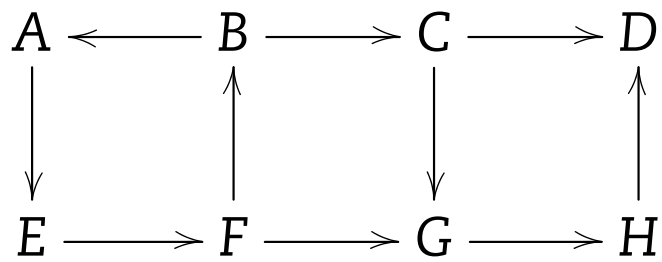
- Plutôt que de définir une fois pour tous les cas possibles de théorèmes, lemmes, corollaires, principes, axiomes, énoncés, remarques, notes, conjectures, et j'en passe, avec leurs numérotations respectives et interdépendantes, LaTeX propose une manière standard de définir de telles structures ;
- en écrivant `\newtheorem{theo}{Théorème}` on définit une nouvelle structure nommée « Théorème » dont l'environnement LaTeX porte le nom `theo`. Cette structure a sa propre numérotation ;
- en écrivant `\newtheorem{theo}{Théorème}[chapter]` la numérotation de nos « théorèmes » sera remise à zéro dans chaque nouveau chapitre ;
- en écrivant `\newtheorem*`, la nouvelle structure n'est pas numérotée ;

Théorèmes, lemmes, etc. 2/2

- En écrivant `\newtheorem{thei}[theo]{Théorème important}` on définit une structure qui partage le même compteur que la structure `theo`.
- on dispose de six styles d'énoncé : `plain`, `change` (le numéro en premier), `margin` (le nombre dans la marge), et les mêmes trois avec le titre seul sur uhne ligne : `break`, `changebreak`, `marginbreak` ;
- on écrivant `\theoremstyle{...}` avant la définition d'énoncé, il sera dans ce style ;
- on peut personnaliser davantage, avec `\theorembodyfont` et `\theoremheaderfont`, qui prennent comme argument une description de la fonte à utiliser (sans le `\selectfont`).

L'approche XY-pic 1/6

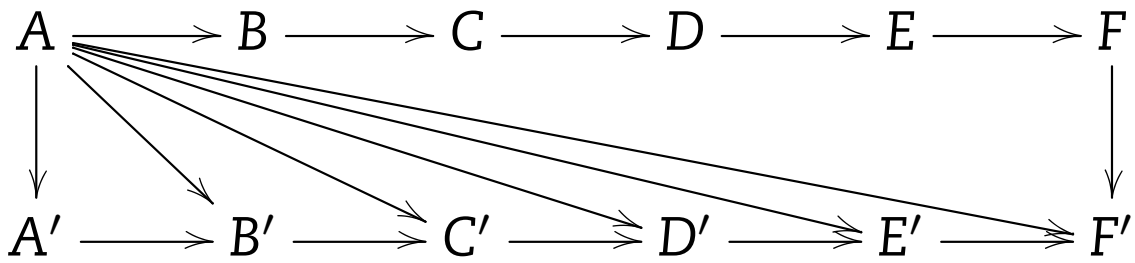
- On charge le package `xy` avec l'option `all` (et `ps` et `dvips` si on n'est pas sous *pdfTEX*) ;
- un diagramme commutatif est écrit dans un élément `\xymatrix` (lui-même dans un environnement mathématique) ;
- comme dans `CD` on écrit les sommets, mais les flèches se mettent juste après les sommets *dont elles partent*. La syntaxe est `\ar[.]` où `.` est `u` (vers le haut), `d` (le bas), `r` (droite), `l` (gauche) :




```
$$\xymatrix{
A \ar[d] & B \ar[l] \ar[r] & C \ar[r] \ar[d] & D \\
E \ar[r] & F \ar[u] \ar[r] & G \ar[r] & H \ar[u]}$$
```

L'approche XY-pic 2/6

- Pour ajouter des labels aux flèches, on utilise `^` (au-dessus), `|` (au milieu) et `_` (au-dessous) après la commande `\ar[.]` ;
- pour centrer une entrée sur la flèche, écrire un tiret après le chapeau (resp. le souligné ou la barre verticale) : `^-`, `_-`, `|-` ;
- les flèches diagonales s'obtiennent en mettant dans l'argument optionnel de `\ar` autant de lettres `u`, `d`, `r` et `l` qu'il y a des pas dans la direction correspondante :

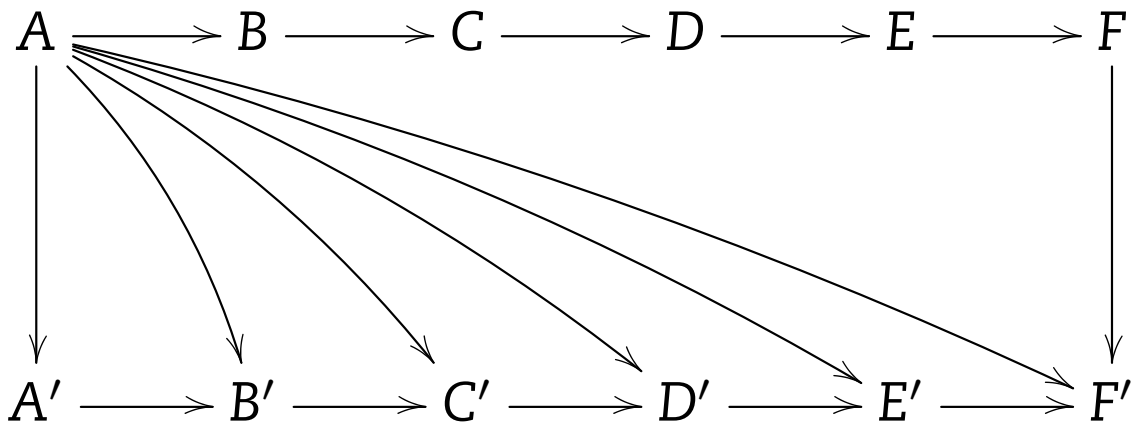


L'approche XY-pic 3/6

```
$$\xymatrix{
A \ar[r] \ar[d] \ar[dr] \ar[drr]
\ar[drrr] \ar[drrrr] \ar[drrrrr] &
B \ar[r] & C \ar[r] & D \ar[r] & E \ar[r] & F \ar[d]\\
A' \ar[r] & B' \ar[r] & C' \ar[r]
& D' \ar[r] & E' \ar[r] & F'
}$$
```

L'approche XY-pic 4/6

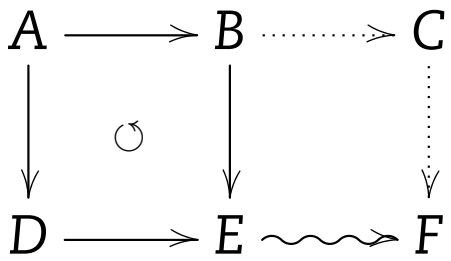
- Les flèches deviennent curvilignes quand on ajoute `@/^/` ou `@/_/` :




```
$$\xymatrix{A \ar[r] \ar[dd] \ar@/^/[ddr]
\ar@/^/[ddrr] \ar@/^/[ddrrr] \ar@/^/[ddrrrr]
\ar@/^/[ddrrrrr] & B \ar[r] & C \ar[r] &
D \ar[r] & E \ar[r] & F \ar[dd]\\
{} & {} & {} & {}& {}\\ A' \ar[r] & B' \ar[r]
& C' \ar[r] & D' \ar[r] & E' \ar[r] & F'}$$
```

L'approche XY-pic 5/6

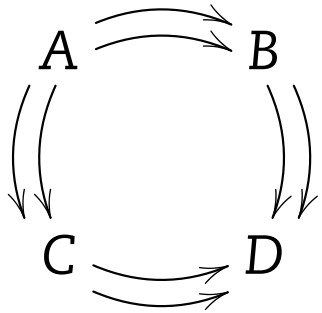
- Les flèches de différents styles s'obtiennent par `@{=>}` (double flèche), `@{.>}` (flèche pointillée), `@{:>}` (flèche pointillée double), `@{~>}` (flèche ondulée), `@{-->}` (flèche en petits traits), `@{}` (flèche vide) ;




```
$$\xymatrix{A \ar[r] \ar@{}[dr]|{\circlearrowleft}
\ar[d] & B \ar@{.>}[r] \ar[d] & C \ar@{.>}[d]\\
D \ar[r] & E \ar@{~>}[r] & F}$$
```

L'approche XY-pic 6/6

- On peut déplacer les flèches transversalement à l'aide de `@<1ex>` :




```
$$\xymatrix{A \ar@<.5ex>@/^/[r] \ar@<-.5ex>@/^/[r]
\ar@<.5ex>@/_/[d] \ar@<-.5ex>@/_/[d]
& B \ar@<.5ex>@/^/[d] \ar@<-.5ex>@/^/[d]\\
C \ar@<.5ex>@/_/[r] \ar@<-.5ex>@/_/[r] & D}$$
```

MathJax

- Les pages Web ordinaires utilisent des balises XML pour représenter leur structure, enrichies de code JavaScript (exécuté sur le client) ;
- Inclure dans l'élément `head` le code :

```
<script type="text/javascript"
src="https://d3eoax9i5htok0.cloudfront.net/mathjax/latest/
MathJax.js?config=TeX-AMS-MML_HTMLorMML">
</script>
```

- puis écrire les formules en TEX en utilisant `\( \)` à la place des simples dollars, et les doubles dollars normalement pour les formules en vedette.
- Plus d'info : `http://www.mathjax.org/docs/2.0/start.html`

Exemple

```
<html>
<head>
<script type="text/javascript"
src="https://d3eoax9i5htok0.cloudfront.net/mathjax/latest/MathJax.
</script>
<title>Pomponnette, c'est maintenant que tu reviens ?</title>
</head>
<body>
<h1>Oh qu'elle est belle, ma formule !</h1>
La vie continue : \( \forall \varepsilon, \exists \eta,
|x-x_0|<\eta \Rightarrow |f(x)-f(x_0)|<\varepsilon \).

Une simple vérité :
$$-1\leq \sin(x)\leq 1.$$
</body>
</html>
```

Résultat

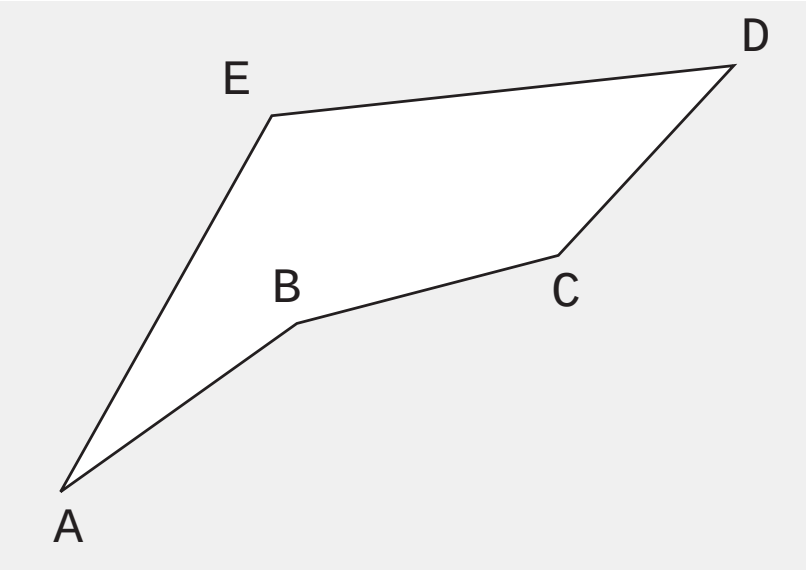
Oh qu'elle est belle, ma formule !

La vie continue : $\forall \varepsilon, \exists \eta, |x - x_0| < \eta \Rightarrow |f(x) - f(x_0)| < \varepsilon$. Une simple vérité :

$$-1 \leq \sin(x) \leq 1.$$

Lettrage de figures : PSFrag 1/5

- À l'aide du package **psfrag** (ne marche pas sous WriteLaTeX) on peut placer des formules mathématiques composées par LaTeX dans une figure EPS ;
- soit la figure **simple.eps** suivante, dont les labels sont des chaînes de caractères uniques, composés en Courier :

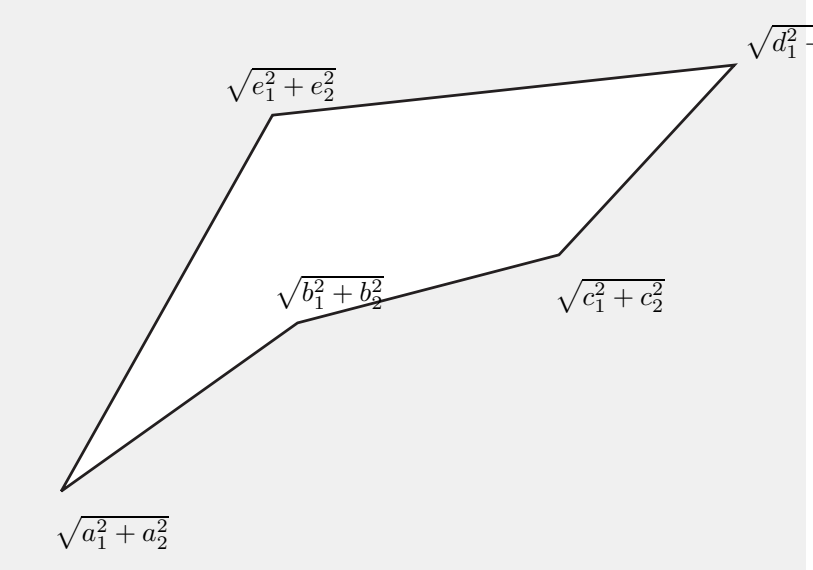


PSFrag 2/5

- En incluant le code suivant avant `\includegraphics` :

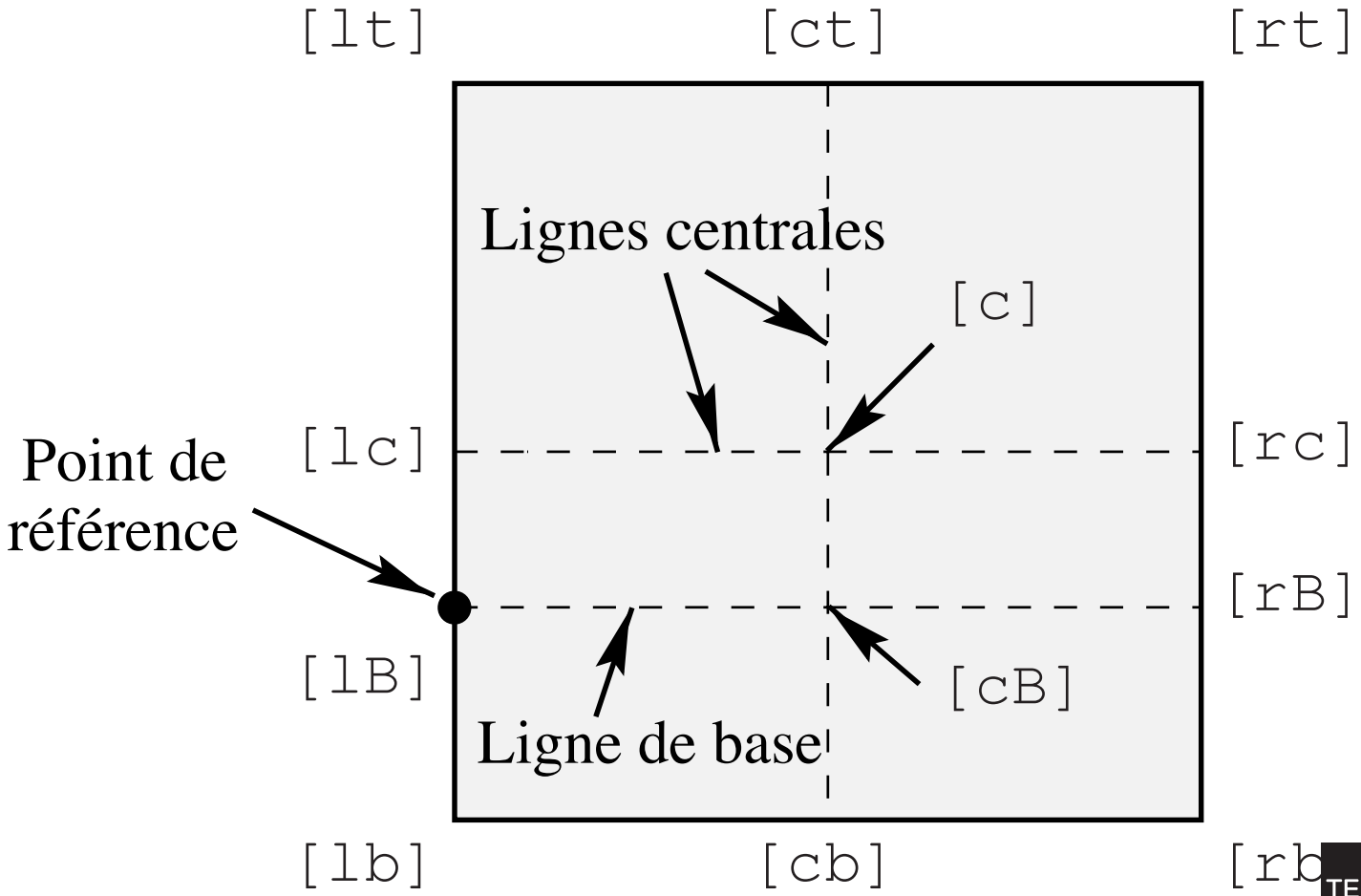
```
\psfrag{A}{$\sqrt{a^2_1+a^2_2}$}
\psfrag{B}{$\sqrt{b^2_1+b^2_2}$}
\psfrag{C}{$\sqrt{c^2_1+c^2_2}$}
\psfrag{D}{$\sqrt{d^2_1+d^2_2}$}
\psfrag{E}{$\sqrt{e^2_1+e^2_2}$}
```

on obtient :



PSFrag 3/5

- Pour optimiser le positionnement des formules on adopte la syntaxe suivante :

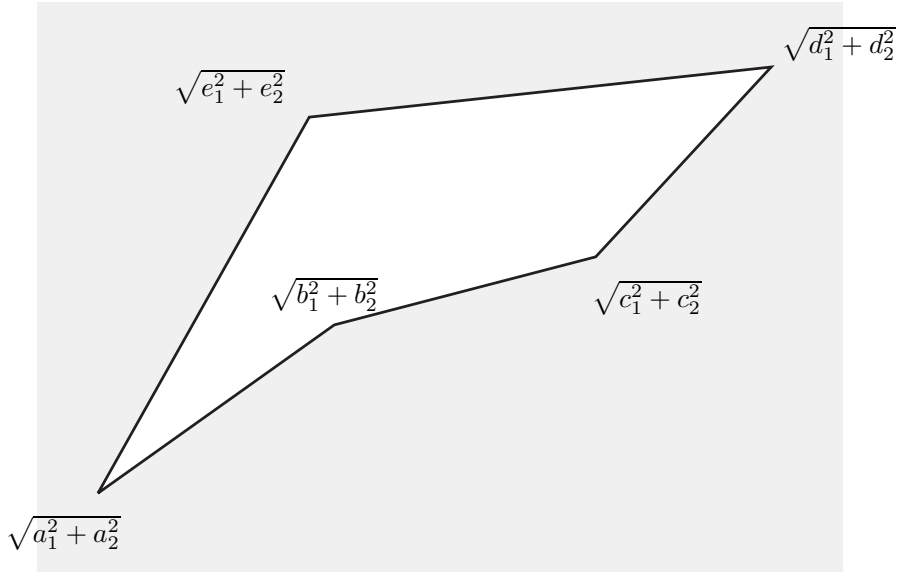


PSFrag 4/5

- La commande **\psfrag** prend quatre arguments optionnels : le *répère choisi au niveau de LaTeX*, le *répère choisi au niveau de PostScript*, le *facteur d'échelle d'agrandissement*, l'*angle de rotation*. Ainsi, avec :

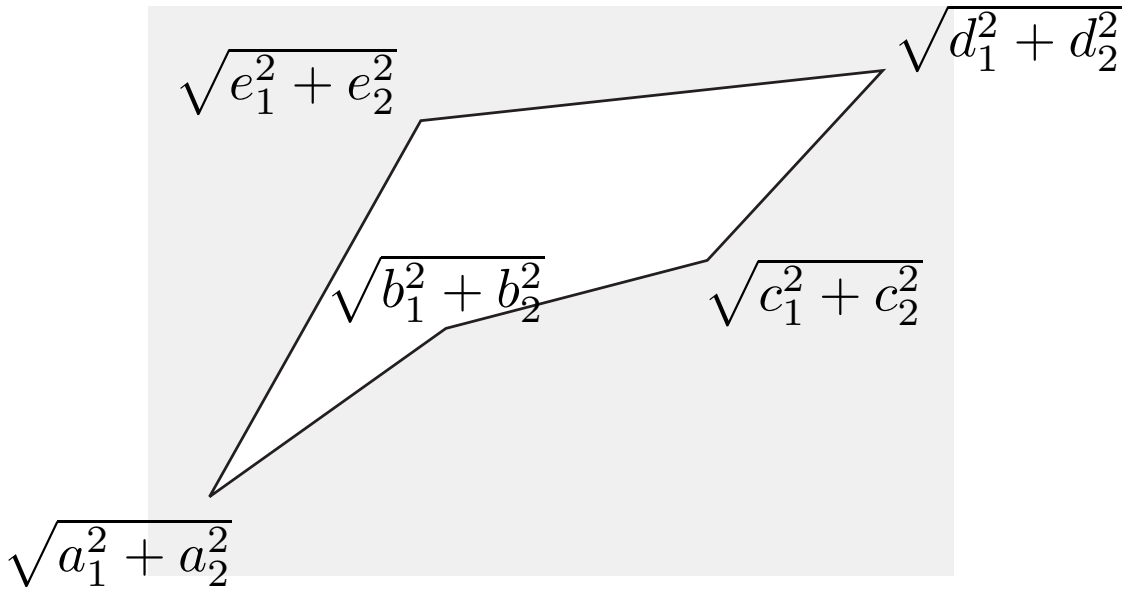
```
\psfrag{A}[rt][rt]{$\sqrt{a^2_1+a^2_2}$}
\psfrag{B}[cB][cB]{$\sqrt{b^2_1+b^2_2}$}
\psfrag{E}[rB][rB]{$\sqrt{e^2_1+e^2_2}$}
```

on obtient :



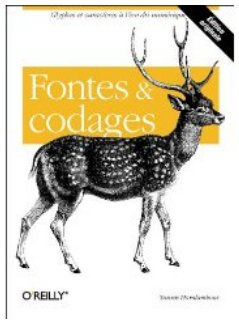
PSFrag 5/5

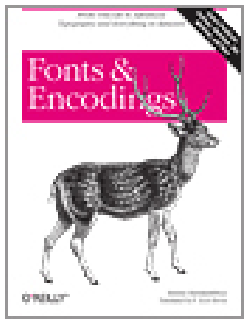
```
\psfrag{A}[rt][rt][2.0]{$\sqrt{a^2_1+a^2_2}$}
\psfrag{B}[cB][cB][2.0]{$\sqrt{b^2_1+b^2_2}$}
\psfrag{C}[lB][lB][2.0]{$\sqrt{c^2_1+c^2_2}$}
\psfrag{D}[lB][lB][2.0]{$\sqrt{d^2_1+d^2_2}$}
\psfrag{E}[rB][rB][2.0]{$\sqrt{e^2_1+e^2_2}$}
```



NFSS (*New Font Selection Scheme*)

- Pour décrire une fonte, NFSS utilise cinq attributs :
  1. le nom de famille, `\fontfamily{...}`,
  2. la graisse/chasse, `\fontseries{...}`, par défaut `m`, le gras étendu étant `bx` (on a aussi `\bfseries`),
  3. le ≪ style ≫, `\fontshape{...}`, par défaut `n`, les italiques étant `it` (on a aussi `\itshape`) et les petites capitales `sc` (`\scshape`),
  4. le corps et l'interlignage, `\fontsize{...}{...}`,
  5. le codage, `\fontencoding{...}`, par défaut `OT1` mais il vaut mieux utiliser `T1` ;
- après les différents choix, on utilise `\selectfont` pour envoyer la requête à NFSS.
- pour plus d'informations consulter





Définir des commandes « à l'ancienne »

- On écrit `\def\toto{...}`, et avec des arguments : `\def\toto#1#2{... #1 ... #2 ...}` ;
- on ne peut utiliser que neuf arguments ;
- quand on imbrique des définitions on utilise des doubles (triples, etc.) dièses : `\def\toto#1{\def\titi##1{... ##1 ... #1}}` ;
- on peut définir un motif d'utilisation de la commande : si l'on définit `\toto` avec `\def\toto,#1/#2..#3:{qqch}`, on l'utilisera alors de la même manière : `\toto,un/deux..trois:` ;
- si un des arguments de la commande est susceptible de contenir plus d'un paragraphe, on écrira `\long\def...`

Définir des commandes « à la LaTeX 2ε »

- On écrit `\newcommand{\toto}{...}`, et avec des arguments : `\newcommand{\toto}[2]{... #1 ... #2 ...}` ;
- `\newcommand` vérifie si la commande n'a pas été déjà définie, sinon on utilise `\renewcommand` (renouvellement forcé) ou `\providecommand` (pas de redéfinition, pas de message d'erreur) ;
- il nous permet de définir un argument optionnel (c'est forcement `#1`) : `\newcommand{\toto}[1][`*`valeur`*`]{... #1 ... #2 ...}`, qui s'utilisera `\toto{qqch}` ou `\toto[qqch]{qqch}`.

Définir des environnements

- On écrit : `\newenvironment{`*nom*`}[`*arg*`]{`*début*`}{`*fin*`}`
  où *arg* est le nombre d'arguments, *début* le code à exécuter au début de l'environnement, et *fin* celui de la fin ;
- on peut utiliser `#1`, `#2`, ... uniquement dans la partie *début* ;
- le contenu d'un environnement est automatiquement inclus dans un groupe ;
- on a également un `\renewenvironment`.

Les compteurs 1/2

- Parmi les compteurs prédéfinis on trouve : `part`, `chapter`, `section`, `subsection`, `subsubsection`, `paragraph`, `subparagraph`, `page`, `equation`, `figure`, `table`, `footnote` ;
- la valeur d'un compteur est donnée par `\arabic{`*nom*`}`, `\roman`, `\Roman`, `\alph` (pour $i <= 26$), `\Alph` ;
- ou `\the`*nom* « à l'ancienne » ;

Les compteurs 2/2

- On définit un nouveau compteur avec `\newcounter{`*nom*`}` (ou `\newcounter{`*nom*`}[`*ancien*`]` si le changement de valeur de l'ancien compteur doit annuler le nouveau) ;
- on change la valeur d'un compteur avec `\setcounter{`*nom*`}{`*valeur*`}` ;
- on incrémente un compteur de un avec `\stepcounter{`*nom*`}` ;
- ou avec `\refstepcounter{`*nom*`}` (qui a l'avantage de positionner `\ref` sur ce compteur) ;
- on incrémente un compteur de *valeur* avec `\addtocounter{`*nom*`}{`*valeur*`}` ;

Les dimensions et longueurs 1/2

- TEX utilise les dimensions suivantes : `pt` (point anglosaxon), `bp` (point PostScript), `dd` (point Didot), `sp` (très petit !), `mm`, `cm`, `in`, `pc` (= 12 `pt`), `cc` (= 12 `dd`), `ex` (la grosseur d'œil), `em` (le cadratin) ;
- on a des dimensions *rigides* et des dimensions *souples*. Une dimension souple est de la forme *`rig`* `plus` *`agr`* `minus` *`rét`* ;
- dans une dimension souple, l'agrandissement et le rétrécissement peuvent être infinis, et de plusieurs ordres d'infinité : `fil`, `fill`, `filll` (sous Ω également `fi`) ;
- avec `\kern1pt` on introduit du blanc horizontal rigide ;

Les dimensions et longueurs 2/2

- Avec `\hspace{10pt plus 1fil minus 5pt}` on introduit du blanc horizontal souple ;
- `\hspace*` fonctionne comme `\hspace` mais est opérationnelle également en début de ligne ;
- on dispose des commandes prédéfinies `\hfill` (blanc horizontal infini), `\hrulefill` (filet infini), `\dotfill` (points de suspension infinis) ;
- avec `\vspace` et `\vspace*` on introduit un blanc vertical (la version avec astérisque fonctionne également en haut de page) ;
- certaines commandes prédéfinies nous facilitent l'insertion de blancs verticaux : `\smallskip`, `\medskip`, `\bigskip`, `\vfill` (blanc vertical infini).

Modes et boîtes 1/2

- La mise en page sous TEX se fait dans différents modes, dont les plus importants sont le mode vertical et le mode horizontal ;
- TEX construit et imbrique des boîtes (horizontales et verticales) qui vont de la page tout entière jusqu'au glyphe, en passant par les lignes et les mots ;
- on peut créer des boîtes artificiellement : « à l'ancienne » ce sera des `\hbox` et des `\vbox` ;
- « à la LATEX » ce sera des `\mbox`. L'intérêt de mettre du texte dans un boîte est qu'il devient indivisible ;

Modes et boîtes 2/2

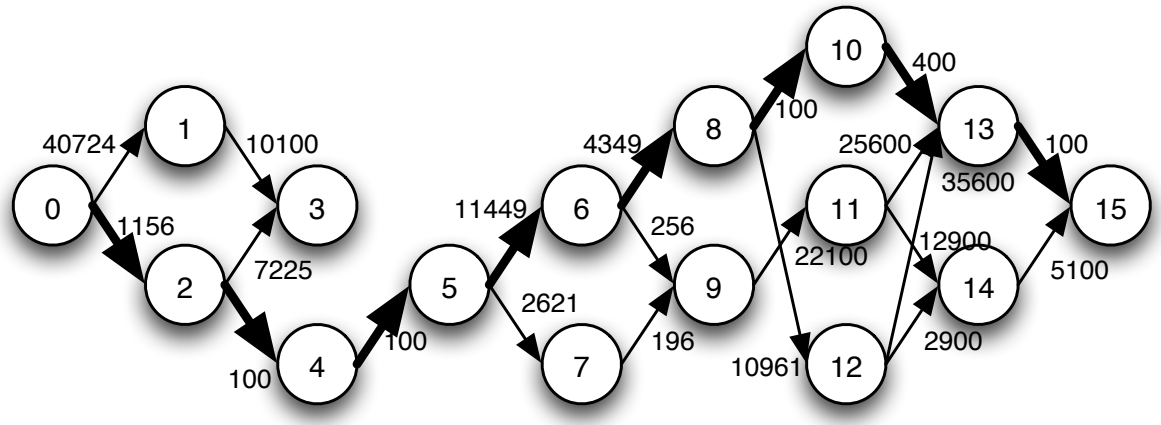
- `\fbox` va, en plus, encadrer le contenu de la boîte, comme ceci (l'épaisseur du filet est donnée par la longueur `\fboxrule`, le blanc entre le contenu et le filet par `\fboxsep`) ;
- `\parbox{`*`largeur`*`}{...}` peut contenir tout un paragraphe de texte ;
- `\fbox` ne peut pas contenir directement un paragraphe de texte, mais on peut mettre un `\parbox` dans un `\fbox`.

Moteur de composition

- TEX va couper les lignes en paragraphes selon 3 passes :
- première passe : sans césure (`\pretolerance`) ;
- deuxième passe : avec césure (`\tolerance`) et les valeurs de *stretch* et *shrink* données par la fonte ;
- troisième passe : en remplaçant le *stretch* par `\emergencystretch` ;
- on utilise un algorithme de plus court chemin dans un graphe acyclique.

Moteur de composition

L'homme écrit. Et parmi le grand nombre d'outils utilisés pour écrire, le dernier en date et le plus complexe n'est autre que l'ordinateur. L'ordinateur qui est en même temps outil d'écriture et de lecture, lieu de stockage, moyen de transmission. Il est devenu un véritable espace à l'intérieur duquel vit le texte, espace qui, comme l'avait prédit, entre autres, MacLuhan, a réussi à s'affranchir des barrières géographiques et à englober toute la planète.



Moteur de composition

Voici un petit texte qui va nous poser bien des problèmes puisque l'on va essayer de le composer dans une colonne bien étroite...
$p = 10000$
$t = 1000$
$e = 0\text{pt}$

Voici
un petit texte qui va nous poser bien des problèmes puisque l'on va essayer de le composer dans
une colonne bien étroite...
$t = 10000$
$e = 0\text{pt}$

Voici un petit texte qui va nous poser bien des problèmes puisque l'on va essayer de le composer dans une colonne bien étroite...
$t = 5000$
$e = 0\text{pt}$

Voici un petit texte qui va nous poser bien des problèmes puisque l'on va essayer de le composer dans une colonne bien étroite...
$t = 5000$
$e = 5\text{pt}$

L'algorithme de césure

- `\language` ;
- `o1`, `1m`, `2m1m` : `co2m1mu1ni1ca1tion` ;
- `1c`, `1n`, `2n1n` : `mé1co2n1nue` ;
- `.co4n4` : `co4n4science` ;
- ce qui manque : plusieurs niveaux, césure au niveau des caractères et non des glyphes.

DVI et fontes

- DVI contient des “opcodes” ;
- il permet de faire les choses suivantes : placer glyphe et avancer, placer glyphe sans avancer, se déplacer horizontalement, se déplacer verticalement, composer boîte noire, inclure spéciale, définir fonte, sélectionner fonte, nouvelle page, push et pop de la position ;
- une fonte TFM contient un certain nombre d’infos générales, et pour chaque glyphe : sa hauteur, profondeur, chasse et correction italique. Elle contient aussi des paires de crénage et des ligatures intelligentes (`LIG`, `LIG/`, `/LIG`, `/LIG/`, `LIG/>`, `/LIG>`, `/LIG/>`, `/LIG/>>`) ;
- une fonte virtuelle contient les mêmes opcodes qu’un fichier DVI (mis à part le début de page) ;
- luaTeX s’ouvre aux OpenType.

Articulation des fichiers 1/2

On crée un *package* (fichier `.sty`) dans lequel on met :

- `\NeedsTeXFormat{LaTeX2e}[`*`date`*`]` ;
- `\ProvidesPackage{nom_package}` ;
- `\ProvidesFile{nom_fichier}` ;
- `\DeclareOption{nom_option}{code_a_executer}` ;
- `\DeclareOption*{code_a_executer}` en cas de nom d'option inconnu ;
- `\ExecuteOptions{liste_options}` pour indiquer des valeurs par défaut ;
- `\ProcessOptions` pour traiter les options.

Articulation des fichiers 2/2

On a des options globales (celles de la classe) et locales (celles du package). Avec `\PassOptionsToPackage{liste_options}{package}` on peut transmettre des options à un package.

- `\usepackage{package}` dans le fichier `.tex` ;
- `\RequirePackage[liste_options]{package}` ne le charge que si nécessaire ;
- `\RequirePackageWithOptions{package}` le package appelé hérite des options de l'appelant ;
- `\AtBeginDocument{code}` code à exécuter en début de document ;
- `\AtEndDocument{code}` code à exécuter en fin de document ;
- `\IfFileExists{fichier}{si_oui}{si_non}` exécuter du code si un fichier existe ;
- `\InputIfFileExists{fichier}{si_oui}{si_non}` idem mais lit le code à partir du fichier après celui de `si_oui`.

Dimensions globales

- `\paperheight` et `\paperwidth` ;
- `\textheight` et `\textwidth` ;
- `\hoffset` et `\voffset` (par défaut : 1in) ;
- `\evensidemargin`, `\oddsidemargin`, `\topmargin` ;
- `\headheight` (hauteur du titre courant supérieur) et `\headsep` (distance entre t.c. et corps du texte) ;
- `\footskip` distance entre bas du corps du texte et bas du titre courant inférieur ;
- paramètres à changer avec `\setlength\dimension{valeur}` ;
- le package `layouts` permet de vérifier ;
- le package `crop` met des hirondelles.

Les titres courants

- package `fancyhdr`
- `\pagestyle{fancy}` ;
- remplir `\fancyhead[positions]{en_haut}` et `\fancyfoot[positions]{en_bas}` ;
- les positions : `L`/`C`/`R` suivi de `O`/`E` (impair/pair), séparés par des virgules ;
- exemple : `[LE,RO]` ;
- `\renewcommand\headrulewidth{0.4pt}` et idem pour `foot` pour le filet ;
- `\fancyheadoffset[positions]{taille}` pour aller dans la marge ;
- `\fancypagestyle{plain}{...définitions...}` pour créer/modifier un autre style (par exemple `plain` pour les premières pages de chapitre.

Les sections

On a une commande générique pour créer les sections :

```
\@startsection{nom}{niveau}{retrait}{avant}{après}{style}
```

- nom : celui par lequel on va l'appeler ;
- niveau : chapitre = 0, section = 1, etc. ;
- retrait : le retrait global du titre de section ;
- avant : glue verticale avant le titre ;
- après : glue verticale après le titre ;
- style : des instructions sans argument pour décrire le style.

On écrira donc :

```
\makeatletter \renewcommand\subsection{%
\@startsection{subsection}{2}{...}{...}{...}{...}} \makeatot
```