

# CURS D'INTRODUCCIÓ A L<sup>A</sup>T<sub>E</sub>X

Optica't UAB

Versió 1.0

Benvolguts i benvolgudes a aquest text introductori de LaTeX. Som Optica't, un grup de divulgació d'òptica física format per estudiants de la Universitat Autònoma de Barcelona. Aquest text està pensat per a facilitar l'aprenentatge d'aquesta eina a alumnes que comencin els seus estudis universitaris. Clarament, està orientat a graus científics, el nostre aprenentatge de LaTeX es basa en les necessitats i la curiositat que hem desenvolupat al llarg dels cursos en carreres com física o nanociències. Tot i això, pensem que LaTeX és una eina molt útil per a més disciplines i esperem que molta gent pugui fer profit d'aquest curs.

# Índex

<b>1</b>	<b>Introducció i conceptes bàsics</b>	<b>5</b>
1.1	Què és LaTeX i la seva utilitat	5
1.2	Entorns per a treballar amb LaTeX	5
1.2.1	Overleaf	6
1.2.2	LaTeX d'escriptori	7
1.2.3	Eines alternatives	7
1.3	Estructura bàsica d'un document	8
1.3.1	Sintaxi	8
1.3.2	Tipus de document	11
1.3.3	Paquets	12
1.3.4	Entorns	12
<b>2</b>	<b>Format de text i estructura</b>	<b>15</b>
2.1	Format del text pla	15
2.2	Justificació i espaiat del text	16
2.3	Llistes	18
2.4	Múltiples columnes	20
2.5	Estructura per a un document científic	22
2.5.1	Pàgina de títol	22
2.5.2	Capítols, seccions, subseccions i índex	23
2.5.3	Footnotes i etiquetes	25
<b>3</b>	<b>L'entorn matemàtic</b>	<b>28</b>
3.1	Formes de cridar un entorn matemàtic	28
3.2	Sintaxi de l'entorn matemàtic	31
3.2.1	Operacions i relacions matemàtiques	33
3.2.2	Parèntesis i claudàtors	33
3.2.3	Alguns accents, símbols i formats útils	33
3.2.4	Límits, sumatoris, productoris i integrals	33
3.3	Matrius i alineació	33
3.4	Exemples	33
<b>4</b>	<b>Figures i taules</b>	<b>34</b>
4.1	Format de pàgina	34
4.2	L'entorn <code>minipage</code>	35
4.2.1	Incloure imatges al document	36
4.3	L'entorn <code>figure</code>	41
4.4	Com col·locar subfigures	42
4.5	L'entorn <code>table</code>	42
<b>5</b>	<b>La bibliografia</b>	<b>45</b>
5.1	L'arxiu <code>.bib</code>	45
5.2	Referenciar i imprimir la bibliografia	45

<b>6</b>	<b>Personalització avançada</b>	<b>46</b>
6.1	Ús d'arxius externs	46
6.1.1	Arxiu externs com a preàmbul (comandament input)	46
6.2	Paquets útils	46
6.2.1	colorbox	46
6.2.2	babel	46
6.2.3	fancyhdr	46
6.2.4	hyperref	46
6.2.5	wrapfig	46
6.2.6	mhchem	46

## Agraïments

# Capítol 1

## Introducció i conceptes bàsics

Dediquem aquest capítol a explicar des de zero com funciona LaTeX i com podem començar a crear els nostres primers documents. La finalitat del capítol és que us quedeu amb els conceptes bàsics de com funciona l'eina, la seva sintaxi, els possibles errors que trobareu, etc.

### 1.1 Què és LaTeX i la seva utilitat

LaTeX és un editor de textos que funciona de forma similar a un llenguatge de programació compilat. Està basat en el llenguatge de baix nivell TeX, i està pensat principalment per a escriure textos científics de forma senzilla i elegant. La filosofia d'aquesta eina és que qui redacti no es preocupi del format del document, només en què ha d'escriure. A diferència d'un editor de textos WYSIWYG (en anglès, 'el que veus és el que obtens') al treballar amb LaTeX no podem veure el resultat del que redactem al moment d'escriure-ho. En LaTeX es treballa directament amb codi i després un compilador ho interpreta i genera un resultat (per exemple imprimeix el document per pantalla o genera un arxiu `.pdf`).

En un principi pot semblar que escriure en codi i no veure què és el que estem generant és contraproduent i pot fer menys atractiva la idea de treballar amb aquest editor i no amb altres més populars. Però, com veureu al llarg d'aquest text, aquest desavantatge queda opacat per la quantitat d'eines útils i de personalització que s'obtenen en treballar amb LaTeX.

Aquest curs se centra a abastar tot el necessari per a començar a escriure textos científics, com ara lliuraments de diverses assignatures de la carrera o per al treball de fi de grau. Però, en ser només un text introductori, no podem explicar totes les eines que ofereix LaTeX bàsic i encara menys totes les eines que ha creat la comunitat. És necessari dir també que la utilitat d'aquest editor de textos va més enllà dels textos científics semblants a *papers*. LaTeX té incorporades eines per a escriure llibres, per a escriure música, per a fer presentacions... Us convidem a informar-vos més enllà d'aquest text perquè podeu jugar amb totes les possibilitats que ofereix LaTeX tot i que no estúdieu a cap branca científica.

### 1.2 Entorns per a treballar amb LaTeX

Per començar a treballar amb LaTeX primer necessitem saber on! És a dir, necessitem saber quin és el nostre entorn de treball. Igual que passa amb els llenguatges de programació, podem córrer LaTeX a dins d'una gran varietat d'entorns. Cada entorn és lleugerament diferent de l'altre, les seves característiques depenen de la versió i de quin sigui el flux de treball pensat específicament per a aquell entorn. Escollir un bon entorn pot arribar a ser una mica enrevessat, per tant, per a facilitar la tasca a les persones que encara no han fet servir LaTeX o que no estiguin familiaritzades en treballar amb programes de manera local centrarem el curs en l'entorn d'Overleaf. Sense ser exhaustius també parlarem sobre altres entorns com algunes eines d'escriptori o a escriure directament en un document de text amb eines més específiques de programació.

Per al lector o lectora que ja tingui experiència amb Overleaf i vulgui fer servir les altres eines, pròximament crearem una guia que complementi aquest curs amb més informació, de moment ens centrarem en que pugueu escriure el vostre primer document en LaTeX.

### 1.2.1 Overleaf

L'entorn per excel·lència per a aprendre LaTeX i per a fer petits projectes col·laboratius és Overleaf. Per a fer-ho servir no cal instal·lar-se res, només necessites connexió a internet i un usuari. Overleaf és un entorn de LaTeX en línia pensat per a fer més fàcil la col·laboració entre usuaris a l'hora de crear un document compartit. Els seus avantatges són:

- No requereix cap instal·lació, només tenir un compte. Per a poder accedir a un compte d'Overleaf només necessiteu un correu electrònic i entrar a la web oficial<sup>1</sup>
- Interpreta el codi fins i tot si aquest conté algun petit error
- Permet treballar conjuntament amb diverses persones<sup>2</sup>
- Permet classificar els teus projectes amb etiquetes
- Té instal·lats alguns paquets que el LaTeX pur no té per defecte
- Té un sistema de navegació i una visualització del log de compilació intuïtius
- Té integrades algunes eines que faciliten la generació de codi, com ara botons o *short cuts* que canvien el format del text com en Word o un generador de taules buides
- Proporciona un editor visual on es pot treballar en un entorn semblant a un editor WYSIWYG

Com aquesta serà l'eina que farem servir a la resta del curs, explicarem una mica més sobre com funciona. Un cop tinguem obert un compte podrem crear un nou projecte des del menú principal. Overleaf ofereix moltes plantilles preestablertes, però per a aprendre treballarem amb una plantilla buida. Un cop obert el nou projecte en blanc veurem la següent pantalla:

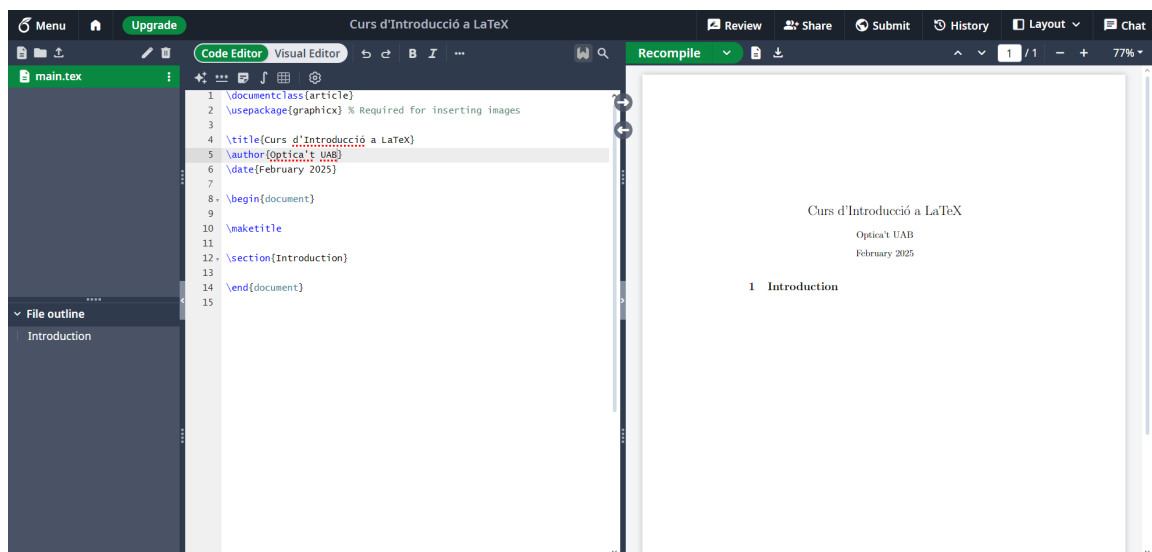


Figura 1.1: Entorn de treball d'Overleaf per a un projecte en blanc.

Observem de la Figura 1.1 que l'entorn se separa en tres parts. A l'esquerra del tot observem un menú, l'explorador d'arxius i de projecte. Aquest explorador funciona com a un gestor de carpetes i d'arxius, molt similar a l'explorador d'arxius de Windows. A la part superior de l'explorador aniran apareixent els arxius que fem servir al projecte. Podem mantenir un ordre d'aquests arxius utilitzant carpetes i canviant el nom de cada arxiu com vulguem. A la part inferior de l'explorador apareixerà un llistat de títols que ens serveixen per a moure'ns pel projecte de forma més senzilla, expliquem el seu funcionament amb més detall a la secció 2.5.2.

A la dreta de la pantalla s'observa un visor pdf. Aquest visor permet veure per pantalla com va quedant

<sup>1</sup>Us deixem l'enllaç vigent actualment: <https://es.overleaf.com>

<sup>2</sup>Actualment (inici del 2025) la limitació per a editar el mateix projecte d'Overleaf es limita a dos usuaris. La resta d'usuaris que entrin al projecte només poden llegir-ho.

el document que estem generant amb el codi que hem escrit. Notareu que a la part superior esquerra del visor hi ha un botó verd on posa **Recompile**. Cada cop que vulguem actualitzar el pdf amb la informació que hem escrit en codi hem de fer click a **Recompile**<sup>3</sup>. Diem a aquesta acció recompilar o compilar el codi.

Entre l'explorador i el visor tenim l'editor de text. Aquest editor és igual que qualsevol altre en programació. Allà és on escrivim totes les línies de codi necessàries per a generar el nostre document. Tot i això, aquesta finestra intermitja no sempre farà d'editor. Overleaf permet treballar amb més tipus d'arxiu que els de LaTeX com per exemple imatges. Per tant és més encertat pensar en l'editor com un entorn dinàmic de treball i no només com un simple editor de text.

Es pot configurar la visibilitat tant de l'editor com del visor pdf a partir de les opcions del desplegable **Layout**. Aquestes opcions deixen només visible un dels dos, els dos o permet separar-los en finestres diferents del navegador (això és útil si per exemple treballem a doble pantalla).

### 1.2.2 LaTeX d'escriptori

A part de l'Overleaf, també existeix la versió per escriptori de LaTeX sense necessitat de connexió a la xarxa. El procés de baixada i instal·lació pot resultar una mica enrevessat i feixuc si és el primer cop que es treballa amb programes com aquest.

El LaTeX d'escriptori funciona lleugerament diferent que els programes d'edició de textos que estem acostumats. Per fer-lo funcionar necessitem: el **MiKTeX** i el **TeXMaker**. El primer funcionarà com a administrador de paquets els quals haurem d'instal·lar en el nostre ordinador per fer-ne ús. El segon és l'editor en sí, on hi escriurem els nostres documents.

Per baixar-se el MiKTeX és el següent enllaç on hi ha l'opció per a Windows, Linux i Mac <https://miktex.org/download> i per baixar-se el TeXMaker <https://www.xmimath.net/texmaker/>. Per facilitar la instal·lació, crearem una carpeta en el nostre ordinador i en allà serà on hi posarem el MiKTeX.

- Quan comencem a instal·lar se'ns obrirà una pestanya que ens demanarà *Preferred Paper* en el qual hi posarem la opció A4 i a *Install missing packages* triarem l'opció *Ask me first*. Això vol dir que cada vegada que el MiKTeX detecti que estem utilitzant un paquet que no el tenim instal·lat, ens demanarà permís per instal·lar-lo. Cal remarcar que el procés d'instal·lació trigarà el seu temps així que millor prendre-s'ho amb calma.
- Quan aparegui l'*Update Check*, deseleccionem l'opció de *Check for updates now* així agilitzarem el procés.

D'aquesta manera, ja tenidrem el MiKTeX instal·lat. Ara falta el TeXMaker que ja té una instal·lació més normal.

Un cop tinguem el MiKTeX i el TeXMaker en el nostre ordinador podrem començar a escriure el nostre primer document en el LaTeX d'escriptori. El funcionament del programa és molt semblant a l'Overleaf amb la única diferència que s'ha d'anar guardant. Això sí, serà necessari que el nostre ordinador disposi d'un lector de PDF i un administrador de referències com ara el JabRef <https://www.jabref.org/>.

És recomanable, que cada document es creï en una carpeta diferent ja que quan guardem els nostres documents, es guarda la versió PDF, versió de TextMaker i llavors altres arxius que no ens interessin (quan treballem a un nivell introductori, es clar). Cada vegada que compilem el document, l'arxiu PDF es reescriurà amb la nova informació.

### 1.2.3 Eines alternatives

Per a la gent més experimentada o amb més interès en la programació deixem el següent entorn que, tot i potser semblar més complicat de fer servir, permet personalitzar completament el teu entorn de treball. LaTeX és un llenguatge, no un programa, per tant desde qualsevol editor de text podem escriure en LaTeX i, fent servir el compilador corresponent podrem generar la sortida que vulguem. Ara bé, existeixen eines

<sup>3</sup>No cal fer click sempre, existeixen dues alternatives. La primera és fer servir *short cuts*, és a dir, prémer **Ctrl + s** o bé **Ctrl + Enter**. La segona és activar la funció automàtica de recompilar. Podeu accedir a aquesta opció desde el botó **Recompile**, prement la fletxa que té al costat.



que faciliten aquesta feina i ens permeten configurar el nostre entorn de forma més visual. Podriem fer una llista enorme amb eines de visualització útils i personalitzables, però una de les més populars (i des de la que s'està redactant aquest text) és Visual Studio Code. La instal·lació i configuració d'aquest entorn queda fora dels propòsits d'aquest curs, però us convidem a informar-vos una mica sobre aquesta eina ja que no serveix només per a LaTeX. Visual Studio Code permet treballar amb LaTeX (i molts més llenguatges) en local, té integrat un explorador de directoris molt visual (no cal barallar-se amb la bash per a navegar per les carpetes) i permet afegir extensions pensades per a treballar amb LaTeX.

## 1.3 Estructura bàsica d'un document

De forma similar a un llenguatge de programació quan escrivim un document en LaTeX necessitem seguir un cert ordre i respectar una sintaxi concreta per a que el compilador pugui comprendre i traduir correctament el nostre document. Aquestes normes que s'han de seguir no són complicades, de fet amb Overleaf són similars a escriure en Word. Com hem comentat abans, LaTeX està pensat per a que no ens haguem de preocupar per la forma sinó més per què escrivim, per tant és aquí on hi ha la diferència principal amb altres eines d'edició de textos.

### 1.3.1 Sintaxi

L'estructura d'un document de LaTeX se separa en dues parts ben diferenciades:

- **Preàmbul:** a on escrivim la “configuració” del document
- **Text principal:** a on escrivim el cos del document

Separem aquestes dues parts de forma explícita, reservant sempre les primeres línies de codi per al preàmbul i la resta per al text principal. Veiem com a exemple el codi que escriu Overleaf per defecte quan obrim un document en blanc:

#### Exemple 1.1 <sup>4</sup>

```

1  \documentclass{article}
2  \usepackage{graphicx} % Required for inserting images
3
4  \title{Curs d'Introduccio a LaTeX}
5  \author{Optica't UAB}
6  \date{February 2025}
7
8  \begin{document}
9
10 \maketitle
11
12 \section{Introduction}
13
14 \end{document}
```

Des de la primera línia fins a la setena és el preàmbul. La resta és el text principal i comença amb el comandament `\begin{document}`. L'exemple 1.1 ens permet veure els tres tipus de comandaments bàsics del preàmbul i com s'estructura un document. El preàmbul de l'exemple comença declarant a la primera línia el tipus de document que estem redactant. Seguidament, a la segona, declara el paquet `graphicx`. Els últims tres comandaments configuren tres objectes predefinits (el títol, l'autor i la data). Veiem doncs que al preàmbul els comandaments bàsics que podem fer servir són:

- Declaració de tipus de document
- Declaració de paquets
- Configuració d'entorns pre-establerts

<sup>4</sup>Tant Overleaf com LaTeX permet fer servir accents escrits directament amb el teclat o amb codi (ho veurem més endavant). Però, l'entorn que fem servir per a escriure els exemples no els detecta correctament. Això no serà un problema per a vósaltres a l'hora de redactar un document sense aquest recurs. Si us plau, feu servir accents.

Parlarem amb més detall d'aquests comandaments a les següents seccions (1.3.2, 1.3.3 i 1.3.4). La resta del document, allò que es plasmarà al compilar, es redacta en un entorn delimitat pels comandaments `\begin{document}` i `\end{document}`. El codi que hi ha escrit entre aquests delimitadors serveix per a imprimir al document el títol i el títol de la secció anomenada "Introduction". Més endavant ens interessarem per aquests comandaments. Encara més important per a aquesta part del codi és com es redacta text pla per a que s'imprimeixi com volem per pantalla. Veiem-ne alguns exemples:

### Exemple 1.2

```
1 \documentclass{article}
2
3 \begin{document}
4     Això s'escriu en text pla, amb una mida de font standar i amb
5     uns marges per defecte.
6 \end{document}
```

#### Sortida:

Això s'escriu en text pla, amb una mida de font standar i amb uns marges per defecte.

Com veiem el text s'escriu justificat a l'amplada del document i amb una mida de text normal. La mida i el tippus de lletra es pot modificar tant a la configuració del preàmbul com al mateix codi (veure secció 2.1). Si tenim més text aquest s'adaptarà en forma de paràgraf. Existeixen diverses formes de separar dos paràgrafs i cada forma crea un salt de línia de diferent mida i genera o no indentació per defecte.

### Exemple 1.3

```
1 \documentclass{article}
2
3 \begin{document}
4     Aquest es un text d'exemple que omplira l'amplada del document
5     formant un unic paragraf. Podem separar-ho en altres paragrafs de
6     diferents maneres.\\
7     Aquest paragraf esta separat per $\backslashbackslash$. S'aplica un
8     salt de linia normal, amb una separacio igual a la d'un salt de
9     linia del mateix paragraf i no aplica indentacio.
10
11     Deixar una linia en blanc tambe genera un altre paragraf com al
12     escriure $\backslashbackslash$ pero a mes aplica indentacio. Si
13     LaTeX detecta que hi ha mes d'un espai en blanc normalment avisa amb
14     un error de sintaxi. Overleaf permet fer aquest espais en blanc i
15     ho interpreta com a un de sol.Podem generar el mateix efecte
16     separant el paragraf amb el comandament $\backslash$par. LaTeX quan
17     es troba aquest comandament enten que s'ha acabat un paragraf i que
18     lo que ve a continuacio pertany a un seguent paragraf. \par Per
19     exemple aquest text esta separat pel comantament $\backslash$par
20     .\\
21     Aquest darrer paragraf se separa amb quatre $\backslash$. Aquesta
22     forma de separar paragrafs no genera indentacio i crea un salt de
23     linia mes gran, fent que el text se separi en dos de forma mes
24     visual.
25 \end{document}
```

#### Sortida:

Aquest es un text d'exemple que omplirà l'amplada del document formant un únic paràgraf. Podem separar-ho en altres paràgrafs de diferents maneres.

Aquest paràgraf està separat per `\\`. S'aplica un salt de línia normal, amb una separació igual a la d'un salt de línia del mateix paràgraf i no aplica indentació.

Deixar una línia en blanc també genera un altre paràgraf com al escriure `\\` però a més aplica indentació. Si LaTeX detecta que hi ha més d'un espai en blanc normalment avisa amb un error de sintaxi. Overleaf permet fer aquest espai en blanc i ho interpreta com a un de sol. Podem generar el mateix efecte separant el paràgraf amb el comandament `\par`. LaTeX quan es troba aquest comandament entén que s'ha acabat un paràgraf i que lo que ve a continuació pertany a un següent paràgraf.

Per exemple aquest text està separat pel comandament `\par`.

Aquest darrer paràgraf se separa amb quatre `\`. Aquesta forma de separar paràgrafs no genera indentació i crea un salt de línia més gran, fent que el text se separi en dos de forma més visual.

El detall de com escriure `\` dins del codi per a que s'imprimeixi correctament ho explicarem més endavant. A l'exemple 1.3 podem veure quatre formes de separar paràgrafs. L'ús d'un salt o altre ve a gust de qui està redactant el document. En aquest text, per exemple, normalment separem els paràgrafs amb `\\` per a obtenir un efecte visual més polit. La separació de paràgrafs de vegades ve condicionada pel que té el paràgraf al seu voltant, usualment pel codi just a sobre d'ell<sup>5</sup>. En cas de voler afegir o treure una indentació desitjada o no podem fer servir el comandament `\indent` o `\noindent`.

La indentació a LaTeX pot portar una mica de confusió no només amb el que s'imprimeix en pantalla al generar el resultat, sinó també amb la indentació dins del propi codi. L'estructura del codi que escrivim ve imposada pel fet que hem de fer entendre al compilador què volem que surti exactament. Però, per a poder facilitar la nostra lectura del codi podem fer servir una jerarquia amb la indentació dins del codi. Fixem-nos novament en l'exemple 1.3, el text que s'acabarà imprimint al document es troba indentat respecte als delimitadors del document (el `begin` i el `end`). Podríem escriure aquest codi sense aplicar aquesta indentació i el resultat per pantalla acabaria sent el mateix que abans. Aquesta indentació global el compilador la ignora, però a nosaltres ens és útil per a saber que aquella part del codi pertany al document. En cas de tenir més entorns a dins del codi (com veurem a la secció 1.3.4) podem aprofitar aquesta indentació per a poder escriure un codi més polit que pugui ser fàcilment entès tant pel compilador com per a qui escriu o revisa el codi.

## Paraules reservades i comandaments

Com heu pogut observar als exemples que hem posat fins ara hi ha algunes paraules que les hem posat en color blau. Semblant a un llenguatge de programació, LaTeX té un conjunt de paraules i símbols reservats per a que el compilador entengui si estem escrivint text pla o si estem donant una ordre concreta. Els comandaments (ja siguin del preàmbul o a dins del text o entorns matemàtics) venen declarats començant pel símbol `"\"`<sup>6</sup>. Entorns visuals com Overleaf faciliten la tasca de saber quines són aquestes paraules reservades canviant el seu color al codi o fins i tot fent prediccions del codi que volem escriure.

No és viable aprendre de forma activa quines són aquestes paraules (no recomenem provar-ho, és una tasca molt farragosa). Lo més útil és practicar i amb el temps acostumar-se a la forma intuïtiva que té LaTeX de seleccionar aquestes paraules, ja que normalment són instruccions literals en anglès. Al llarg del curs explicarem algunes d'aquestes paraules reservades i a l'Annex deixarem algunes taules amb exemples útils que podeu fer servir al vostre dia a dia.

A més, veureu que acompanyant a aquestes paraules reservades normalment apareixen `{ }` o `[ ]`. Entre claus normalment s'escriu l'argument de la funció que estiguem cridant amb la paraula reservada. Entre claudators normalment s'escriuen les opcions de configuració de la funció.

<sup>5</sup>Això ho veureu a mesura que escriviu documents en LaTeX, lo més usual és veure-ho quan es comença un paràgraf després de posar una figura o una taula. Per exemple, al finalitzar l'exemple 1.3 comença un paràgraf amb indentació automàtica.

<sup>6</sup>És per aquest motiu que no hem escrit directament el símbol al codi per a que s'imprimeixi en la sortida del document, ja que LaTeX ho interpretaria com a un comandament i no com a un símbol

### 1.3.2 Tipus de document

Fixem-nos ara en la primera línia del preàmbul. Com hem dit aquest comandament declara quin tipus de document estem escrivint. Segons com configurem aquesta línia, el text que redactem sortirà amb una configuració predefinida o altre. Aquesta configuració és molt bàsica, però pot portar a problemes que semblen que no tenen solució si no es té en compte què hem configurat i que no. Existeixen molts tipus de documents que enten LaTeX, comentem per sobre els més comuns:

- **article** - Aquest és el tipus de document que normalment fareu servir a la carrera. Està pensat per a escriure documents científics curts, de forma que permet estructurar el text en seccions.
- **report** - En cas de necessitar fer un article més llarg podeu fer servir aquest tipus de document. Un **report** permet a més de seccions separar el text en capítols.
- **book** - Aquest és el tipus de document que fem servir per a redactar aquest curs. Treballa de forma similar a un report (és un text llarg i amb capítols) però afegeix detalls necessaris per a estructurar un llibre. Un exemple molt clar d'això és que de forma automàtica genera l'espai en blanc adient per a que els capítols comencin en una pàgina imparella.
- **letter** - En cas de voler escriure una carta amb una estructura formal podeu fer servir aquesta classe de document. **letter** permet declarar al preàmbul algunes variables predefinides (com la signatura o l'acomiadament) que serveixen per a estructurar la carta de forma molt senzilla.
- **beamer** - Aquest darrer format serveix per a crear presentacions. És una classe de document molt personalitzable, però és més complicada de fer servir i això la fa una mica limitada per a la gent que comença a fer servir LaTeX.

Cada classe de document es pot configurar com més ens convingui. L'estructura del comandament sencer és `\documentclass[...]{class}` on entre claudators podem escriure la configuració i entre claus la classe de document que vulguem. Les configuracions poden ser molt variades i es pot posar més d'una entre els claudators sempre que les separem per comes. Deixem algunes configuracions comunes amb una breu explicació i alguns exemples:

- Es pot modificar la mida de la lletra del text pla general posant 10pm, 11pm, 12pm, etc. La unitat de mesura pm s'anomena punt i correspon a aproximadament 0,35 mm al paper.
- Podem configurar la mida del paper segons desitjem. Per defecte en un document **book** la mida és A4, però en altres pot variar. Podem explicitar diferents mides com **a4paper** (mida A4), **letterpaper** (mida de carta, 216 × 279 mm), **b5paper** (format B5, 176 × 250 mm), etc.
- Configurem el format d'impressió a una o doble cara, amb els comandaments **oneside** i **twoside** respectivament. Aquesta configuració canvia per exemple quan comença un capítol, si a la següent pàgina o a la següent pàgina imparella. Cada configuració està pensada per a que es pugui enquadrar a una o a dues cares.
- Configurem la distribució del text ja sigui a una o a doble columna<sup>7</sup> amb els comandaments **onecolumn** per a una columna i **twocolumn** per a dues.
- Podem escollir si els capítols comencen a qualsevol pàgina (**openany**) o a les pàgines imparelles (**openright**).

**Exemple 1.4 Article en format A4 amb lletra de mida 10pm:**

```
1 \documentclass[10pm]{article}
```

**Exemple 1.5 Report mida A4, a dues columnes, lletra mida 12pm i a una cara:**

```
1 \documentclass[a4paper,12pm,oneside,twocolumn]{report}
```

**Exemple 1.6 Configuració d'aquest document:**

```
1 \documentclass[12pm,twosides,onecolumn,openany]{book}
```

Veiem a l'exemple 1.6 que hem aplicat **openany** i no hem explicitat **a4paper** (no és necessari, **book** ho té onfigurat per defecte). Això es deu a que aquest document està escrit per a llegir-se en format digital i no en paper. La versió en paper d'aquest document no hauria d'escriure l'opció **openany** i deixar per defecte **openright** (es podria escriure explícitament, però **book** té aquesta darrera opció per defecte).

<sup>7</sup>Això configura el text de forma general, tot el text s'estructurarà en una a dues columnes. Per a obtenir una millor personalització de la doble columna farem servir altres comandaments (veure secció 2.4).

### 1.3.3 Paquets

Com hem dit existeixen molts comandaments que podem escriure a partir de paraules reservades. Amb aquests comandaments podem, com si fos un llenguatge de programació, configurar funcions que ens modifiquen el que imprimim per defecte en un document escrit en LaTeX. Però, aquesta tasca és molt farragosa, ja que per a fer servir eines molt comunes hauriem de saber la configuració exacta de cada detall que necessitem escriure. És per això que, novament com als llenguatges de programació, existeixen els paquets. Aquests paquets són codi en LaTeX escrit per a configurar diverses funcions que necessitem per a redactar, hi ha molts i de molt diferents. Alguns venen per defecte amb LaTeX i altres venen per part de la comunitat. El funcionament de cada paquet és molt diferent a cada un, fins i tot podem trobar alguns obsolets o incompatibles entre ells. Però, de forma senzilla podem explicar el seu funcionament dient que són comandaments que indiquen a LaTeX que existeixen noves paraules reservades preconfigurades.

Per a poder fer servir un paquet hem de cridar-lo explícitament al preàmbul amb el comandament `\usepackage{...}`, just després de declarar el tipus de document. Podeu trobar paquets molt interessants buscant per internet, us deixem un breu llistat amb alguns exemples:

- `\usepackage[a4paper]{geometry}` - Aquest paquet és útil per a configurar els marges del document, aquest exemple adapta els d'un de mida A4.
- `\usepackage{caption}` - Permet configurar la lletra dels títols de taula i peus de figura.
- `\usepackage{fancyhdr}` - Serveix per a configurar la capçalera i els peus de pàgina.
- `\usepackage{mhchem}` - Permet escriure més fàcilment fórmules i reaccions químiques.

Fent referència a l'explicació simplificada de lo que és un paquet podem veure l'exemple de `mhchem`. Aquest paquet indica a LaTeX que existeix una nova paraula reservada assignada al comandament `"\ce{...}"` la qual amb els arguments adients ens permet escriure fórmules o reaccions químiques més fàcilment.

Un detall important a l'hora d'utilitzar paquets és que no tots es troben instal·lats per defecte a LaTeX. Això depèn molt de l'entorn on es treballa, pot variar fins i tot segons la versió o la distribució que us hagueu instal·lat en local. Overleaf té l'avantatge que la majoria de paquets útils ja venen instal·lats i no us haureu de preocupar per això<sup>8</sup>. La resta d'usuaris que no feu servir Overleaf haureu d'instal·lar els paquets que no tingueu per defecte segons quin entorn estigueu fent servir, ja sigui des de la web dels creadors del paquet o directament llençant un comandament des de la bash en Windows o Linux.

El funcionament de cada paquet es pot trobar a internet, normalment es pot trobar a la bibliografia oficial del paquet o es pot demanar ajuda a eines d'intel·ligència artificial per a que us donin alguna breu explicació. En aquest curs farem servir i explicarem com funcionen diversos paquets comuns o útils.

### 1.3.4 Entorns

Posem novament èmfasi a les paraules reservades. Com hem dit anteriorment el codi que s'acabarà imprimint al document es troba entre els delimitadors `\begin{document}` i `\end{document}`. Les paraules clau `begin` i `end` són dues paraules reservades molt comunes en un codi de LaTeX, ja que obren i tanquen entorns específics dins del codi. Els arguments que accepten aquests delimitadors poden ser molt diversos, de fet podem obtenir nous arguments afegint paquets. L'ús de delimitadors per a construir entorns és molt útil tant per a accedir a les funcions que necessitem per a escriure el nostre text com per a seguir una jerarquia al codi (com vam dir al parlar de la sintaxi). Ja sigui indentat o no, el codi que es troba entre els delimitadors constitueix un bloc aïllat de la resta del codi. Qualsevol comandament (llevat d'aquells que actuen globalment) que escrivim dins de l'entorn delimitat només afectarà al codi que hi hagi dins.

Exemples d'entorns propis de LaTeX són l'entorn matemàtic (parlem més profundament al capítol 3), `equation`; l'entorn de justificació central, `center`; l'entorn de justificació cap a la dreta, `flushright`; l'entorn de les figures, `figure`; etc. Exemples d'entorns d'alguns paquets són: amb el paquet `subcaption` l'entorn de les subfigures, `subfigure`; amb el paquet `tcolorbox` l'entorn d'un quadre de text amb marcs i

<sup>8</sup>Per exemple, `mhchem` es pot fer servir a Overleaf sense cap problema, però en local s'ha d'instal·lar prèviament.

fons configurable, `tcolorbox`; amb el paquet `listings` l'entorn que permet escriure amb diferent format que el text normal codi en altre llenguatge de programació, `lstlisting`, etc. Aquests últims exemples necessiten primer una configuració extra al preàmbul, per tant són una mica més avançats que la resta d'entorns que venen per defecte amb LaTeX. Podeu veure com es fan servir aquests entorns al següent exemple:

### Exemple 1.7

```

1  \documentclass{article}
2
3  \usepackage{subcaption}
4  \usepackage{listings}
5  \usepackage{tcolorbox}
6
7  \begin{document}
8      Aquest text esta justificat per defecte ja que no es troba dins de
9      cap entorn, nomes es troba dins del document.\
10
11     \begin{center}
12         El text que es troba aqui esta justificat al centre. S'observa
13         com per a poder veure millor el codi fem servir indentacio, pero
14         com hem dit el compilador omet aquesta indentacio global. Tambe
15         es veu com separem en paragrafs. La combinacio d'espai en blanc
16         i  $\backslash\backslash$  es equivalent a quatre  $\backslash$ .
17     \end{center}
18
19     \noindent Parlarem mes endavant, pero podeu veure un exemple
20     matematic:
21
22     \begin{equation*}
23         a^2 + b^2 = c^2
24     \end{equation*}
25
26     \begin{flushright}
27         Podem tambe escriure text justificat cap a la dreta. Qualsevol
28         objecte dins d'aquet entorn que no tingui una justificacio fixa
29         es justificara cap a la dreta juntament amb el text. Un exemple
30         d'aixo pot ser una imatge.
31     \end{flushright}
32
33     \begin{tcolorbox}[colframe=red, colback=black!5!white, sharp corners
34     , width=0.85\textwidth, boxrule=0.2mm, parbox=false]
35         Aquest text te justificacio per defecte pero envoltat per una
36         caixa. La configuracio explicita dels colors i la mida es troba
37         entre claudators. La configuracio implicita es troba en el
38         preambul del document del curs, parlarem mes endavant d'allo.
39     \end{tcolorbox}
40 \end{document}

```

### Sortida<sup>9</sup>:

<sup>9</sup>Un detall que no és rellevant però que si que podreu notar en cas d'haver compilat aquest mateix codi o de similars és que LaTeX ens avisa que tenim un paquet carregat però que no fem servir, el paquet `subcaption`. Als exemples no mostrarem aquests avisos, ja que seria ficar-nos a explicar algunes sortides poc rellevants del log de LaTeX i al cap i a la fi no acaba afectant al document final. Aquesta notificació es marca com a error de sintaxi, però el compilador igualment genera el codi resultant ignorant l'error.

Aquest text esta justificat per defecte ja que no es troba dins de cap entorn, nomes es troba dins del document.

El text que es troba aqui esta justificat al centre. S'observa com per a poder veure millor el codi fem servir indentacio, pero com hem dit el compilador omet aquesta indentacio global. Tambe es veu com separem en paragrafs. La combinacio d'espai en blanc i `\\` es equivalent a quatre `\`.

Parlarem mes endavant, pero podeu veure un exemple matematic:

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

Podem tambe escriure text justificat cap a la dreta. Qualsevol objecte dins d'aquet entorn que no tingui una justificacio fixa es justificara cap a la dreta juntament amb el text. Un exemple d'aixo pot ser una imatge.

Aquest text te justificacio per defecte pero envoltat per una caixa. La configuracio explicita dels colors i la mida es troba entre claudators. La configuracio implicita es troba en el preambul del document del curs, parlarem mes endavant d'allo.

Veiem a l'exemple 1.7 com fem servir alguns dels entorns que hem mencionat abans. Els detalls del funcionament de cada un ho veurem deprés, de moment quedeu-vos amb com es treballa amb els entorns.

Aquesta forma de treballar és essencial, ja que ens permet fer servir comandaments locals que afecten a tot el codi que es trobi dins d'un entorn. Per exemple, existeix el comandament `centering`. Aquest és un comandament local, que afecta a tot el seu entorn fins a trobar uns delimitadors (si no hi ha un entorn escrit, els delimitadors són els del mateix document). El comandament `centering` justifica al centre tot alló que es trobi al seu voltant fins a trobar-se uns delimitadors. Si és text lo que hi ha dins d'aquest entorn, `centering` funciona igual que l'entorn `center`.

# Capítol 2

## Format de text i estructura

Aquest capítol del curs el dedicarem a aprendre a construir documents de LaTeX amb les eines més senzilles d'edició del format del text que en permet LaTeX pur. Veureu que els diferents apartats són, bàsicament, una recopilació dels comandaments més comuns que es fans servir per a redactar la majoria de textos científics amb aquesta eina.

### 2.1 Format del text pla

Al capítol anterior hem vist com funciona la sintaxi del text pla, com podem separar-lo per paràgrafs i com fer servir paquets i entorns útils que ens són dmolt bons per a poder redactar documents interessants. Ara parlarem de certs comandaments, també molt bàsics, que permeten configurar el tipus de lletra que fem servir.

Hi ha molts comandaments, ja siguin integrats a LaTeX o que provenen de paquets que ens modifiquen la font del text pla que redactem. Tots aquests comandaments funcionen de la mateixa forma: el text afectat ha de ser l'argument del comandament. Podem fer una breu classificació d'aquests comandaments amb alguns exemples que us deixem a continuació.

#### Modificacions comunes

`\textbf{Text en negreta}` **Text en negreta**                      `\textit{Text en cursiva}` *Text en cursiva*  
  
`\underline{Text subratllat}` Text subratllat

#### Tipus de font

`\text{Text normal}`<sup>1</sup> Text normal                      `\textsf{Text en Sans Serif}` Text en Sans Serif  
  
`\texttt{Text mecanografiat}` Text mecanografiat  
  
`\textsc{Text en majúscules petites}` TEXT EN MAJÚSCULES PETITES

Per a fer canvis en la mida de la lletra fem servir uns comandaments locals que afecten tot l'entorn on treballem. Tan com si escribim en un entorn concret d'alguna funció com si escrivim text pla podem delimitar la part del codi a la que afecten aquests comandaments fent servir claus com a delimitadors. Un exemple general (i inventat) d'aquesta sintaxi per als comandaments locals és:

`{\comandament Aquest text està afectat pel comandament exemple}.`

Posem a continuació alguns exemples de mida de lletra.

---

<sup>1</sup>Aquest comandament i un altre ens serà útil en entorns matemàtics. Per escriure text pla normal no cal fer-lo servir.



### Mida de lletra

`\tiny` Text tiny      `\footnotesize` Text de footnote      Text de footnote

`\large` Text large      `\huge` Text huge      Text huge

Podeu trobar més exemples i el llistat complet de mides a l'Annex. Naturalment aquestes modificacions de les fonts es poden convingar convenientment, deixem també alguns exemples.

#### Exemple 2.1

```

1 \documentclass{article}
2
3 \begin{document}
4   \texttt{\LARGE Text mecanografiat i de mida LARGE}\\\\
5   \textsc{\textit{Text amb majuscles i en cursiva}}\\\\
6   \textsf{\textbf{\footnotesize Text en Sans Serif en negreta i de
7     mida footnotesize}}
8 \end{document}

```

Sortida:

Text mecanografiat i de mida LARGE

*Text subratllat i en cursiva*

Text en Sans Serif en negreta i de mida footnotesize

Però, s'ha d'anar amb cura ja que existeixen alguns comandaments incompatibles entre ells. Normalment quan això passa LaTeX avisa com un error de sintaxi. El document normalment compilarà, però escollirà de forma jeràrquica quin comandament afectarà a la lletra o quin no. Posem-ne un exemple:

#### Exemple 2.2

```

1 \documentclass{article}
2
3 \begin{document}
4   \textsc{\textit{\footnotesize Text en cursiva i en majuscles
5     petites}}\\\\
6   \textit{\textsc{\Large Text en majuscles petites i en cursiva}}
7 \end{document}

```

Sortida:

*Text en cursiva i en majuscles petites*

TEXT EN MAJUSCLES PETITES I EN CURSIVA

Veiem que els comandaments `\textit` i `\textsc` són incompatibles. La jerarquia que se segueix al codi és de dins cap a fora, és a dir, domina el primer comandament que selecciona el text amb les claus. La resta de comandaments per sobre són secundaris i afecten després a no ser que siguin incompatibles. A l'exemple 2.2 com a la primera línia primer apliquem cursiva i després per sobre apliquem el text en majúscules només obtenim la lletra en cursiva. A la segona línia la situació és a la inversa i només obtenim la lletra en majúscules petites. Veiem també com un comandament local com és el canvi de mida no és incompatible amb els altres dos, encara funciona tot i haver-hi un error de sintaxi.

## 2.2 Justificació i espaiat del text

De forma similar amb la font de la lletra, podem modificar la justificació i espaiat del text amb comandaments amb argument o amb comandaments locals. També veurem un nou tipus de comandament que és de tipus local, només que el seu efecte és encara més localitzat ja que només afecta al lloc on hem posicionat el comandament al codi.

Si no apliquem cap entorn o configuració extra al preàmbul el text pla normalment està justificat a tota l'amplada de línia<sup>2</sup>. Com hem vist a l'exemple 1.7 podem canviar aquesta justificació amb entorns com `flushright` o `center`. El primer justifica tot el text al marge dret, el segon justifica el text al centre. Anàlogament a `flushright` podem fer servir `flushleft` per a justificar el text al marge esquerra (aquesta és la configuració per defecte en editors de text com Word). Podem fer servir aquests entorns de forma localitzada amb els seus anàlegs de comandaments locals, respectivament, `\flushright`, `\centering`, `\flushleft`. Escollir si hem de treballar amb els comandaments d'entorn o locals depen tant del nostre flux de treball com de la compatibilitat amb altres comandaments que estiguem utilitzant. Com a petita guia podeu fer servir el següent flux de treball: fem servir comandaments d'entorn per a delimitar paràgrafs de text, però, per a treballar amb objectes que no siguin text fem servir comandaments locals sota els entorns adequats.

A part de poder configurar la justificació del text també podem canviar algunes distàncies concretes deixant espais en blanc amb una mesura precisa. Això ho aconseguim amb comandaments com `\hspace{...}` o `\vspace{...}`. Aquest comandaments, com hem dit abans, funcionen de forma local i justament a la part del codi on els col·loquem, els podem anomenar comandaments doblement locals. El primer genera un espai en blanc horitzontal, el segon fa el mateix però en vertical. L'argument de cada un d'aquests comandaments accepta la distància que ocuparà en el document aquest espai en blanc. Aquesta distància es pot posar amb diverses unitats, ja siguin centímetres, mil·límetres, punts, etc. S'ha de tenir cura, però, ja que un mal ús de qualsevol dels dos comandaments ens pot fer obtenir resultats no desitjats.

### Exemple 2.3

```

1  \documentclass{article}
2
3  \begin{document}
4  Aquest text es troba justificat de forma usual, sense estar dins de cap
   entorn. \hspace{2cm} Aquesta altra part tambe te una justificacio normal
   pero li afecta un espai en blanc de 2 cm.\\ \vspace{1cm}
5
6  Aquest salt de linia hauria d'apareixer amb un salt de linia igual al
   del text, pero, es veu afectat per l'espai en blanc d'1 cm.\\\
7  Tot i que no sigui d'us comu, tambe es poden afegir arguments amb
   distancia negativa, es a dir apropant els objectes que venene separats
   pel comandament que fem servir:
8
9  \begin{center}
10   Text exemple a l'esquerra \hspace{-0.5cm} Text exemple a la dreta
11 \end{center}
12 \end{document}
```

### Sortida:

Aquest text es troba justificat de forma usual, sense estar dins de cap entorn. Aquesta altra part tambe te una justificacio normal pero li afecta un espai en blanc de 2 cm.

Aquest salt de linia hauria d'apareixer amb un salt de linia lleugerament mes gran que l'espai d'interliniat, pero, es veu afectat per l'espai en blanc d'1 cm i apareix mes gran de lo normal.

Tot i que no sigui d'us comu, també es poden afegir arguments amb distancia negativa, es a dir apropant els objectes que venene separats pel comandament que fem servir:

Text exemple a l'esquerra      Text exemple a la dreta

Observem en aquest exemple 2.1 com aquest espaiat pot oferir-nos molt de joc en cas de voler col·locar text o objectes allà on vulguem. Veurem més endavant (capítol 3) com existeixen més formes d'aplicar

<sup>2</sup>Notem que és a l'amplada de línia i no l'amplada dels marges definit al preàmbul. Aquest detall no és rellevant si treballem a una columna, però veurem que si que s'ha de tenir en compte quan treballem amb més d'una columna (veure secció 2.4)

espaïat horitzontal amb mesures ja predefinides per la mida del text.

Un primer exemple de resultats no desitjats que hem mencionat abans és que amb l'espaïat horitzontal pot passar que una part del text o l'objecte que hi hagi a prop se surti dels marges o de la pàgina del document, quedant tallats i amb un error de sintaxi a LaTeX<sup>3</sup>. Un segon exemple, ara amb l'espaïat vertical, és que l'entorn on intentem cridar aquest espai en blanc sigui o bé un entorn directament incompatible amb aquest comandament o bé que aplicant-ho trenquem l'estructura lògica (o sintaxi) de la resta de paràgrafs. En aquests cas LaTeX interpreta que és un error i compila el document ignorant que existeix aquest espaiat. En cas de tenir algun problema que vingui d'aquest comportament dels comandaments d'espaïat ja s'ha de mirar detingudament si existeix alguna alternativa per a poder aplicar l'espai que estem buscant. Depenent de l'entorn on treballem la solució pot ser forçar a LaTeX a ignorar la sintaxi (això és una opció en alguns entorns), fer servir paquets que ofereixen noves formes d'aplicar espaiats, etc. Com a darrer exemple podem fixar-nos novament en l'exemple 2.1, on apliquem un espaiat horitzontal negatiu i les lletres del text s'encabalen.

Com a darrer comandament d'espaïat basic tenim un altre de local que serveix per a saltar a una pàgina en blanc nova des d'on se situa el comandament. Per a cridar-ho només es necessita escriure `\newpage` per a poder indicar a LaTeX on tallar el darrer paràgraf i continuar la resta del text a una pàgina següent. Aquest comandament també pot ser sensible en quant a compatibilitat amb alguns entorns provinents de paquets.

## 2.3 Llistes

Amb les eines que tenim ja podríem construir llistes similars a les que podem escriure amb altres editors de text. Per a fer-ho veiem el següent exemple on aprofitem la indentació natural de la separació de paràgrafs per a poder separar del marge esquerra cada ítem de la llista.

### Exemple 2.4

```

1 \documentclass{article}
2
3 \begin{document}
4 \begin{center}
5     \textsc{La meva primera llista}
6 \end{center}
7 Ara posarem un exemple d'una llista amb quatre ítems:\\
8
9 - Primer ítem\\
10
11 - Segon ítem\\
12
13 - Tercer ítem. Aquest tercer ítem el farem mes llarg que la resta per a
14   observar que es el que pasa si intentem fer una llista nomes aplicant
15   els coneixements que tenim fins ara.\\
16
17 - Quart ítem
18 \end{document}

```

### Sortida:

LA MEVA PRIMERA LLISTA
Ara posarem un exemple d'una llista amb quatre ítems:
- Primer ítem
- Segon ítem
- Tercer ítem. Aquest tercer ítem el farem mes llarg que la resta per a observar que es el que

<sup>3</sup>Aquest error no és crític, el compilador de LaTeX només ens avisa de que és un possible error i ens aconsella corregir-lo, però, no deixarà de compilar el que nosaltres li estiguem enviant.

pasa si intentem fer una llista només aplicant els coneixements que tenim fins ara.

- Quart ítem

Veiem com a l'exemple 2.4 si intentem afegir un ítem amb una mida més gran que l'amplada de línia automàticament salta sense indentació. Tot i que això es pot corregir aplicant els comandaments que coneixem hem de dir que és una correcció una mica enrevesada i pot fer farragosa la tasca de crear qual-sevol·l·listat. És per aquest motiu que LaTeX inclou un entorn ja predefinit per a poder generar llistats i poder personalitzar-los lleugerament. Aquest entorn ve delimitat pel comandament `itemize`, posem un exemple de com es fa servir:

### Exemple 2.5

```

1  \documentclass{article}
2
3  \begin{document}
4  A continuació posarem un exemple de l'entorn \texttt{itemize} amb alguna
   configuració canviada.
5  \begin{itemize}
6      \item Primer ítem
7      \item[$\star$] Segon ítem
8      \item[$\square$] Tercer ítem. Fem també aquest ítem més llarg per a
        comprovar que la indentació queda com nosaltres volem sense haver de
        configurar cap cosa més.
9      \item[-] Quart ítem
10 \end{itemize}

```

### Sortida:

A continuació posarem un exemple de l'entorn `itemize` amb alguna configuració canviada.

- Primer ítem
- ★ Segon ítem
- Tercer ítem. Fem també aquest ítem més llarg per a comprovar que la indentació queda com nosaltres volem sense haver de configurar cap cosa més.
- Quart ítem

Aquest exemple 2.5 deixa veure clarament com l'entorn `itemize` deixa totes les seves entrades amb una correcta indentació, a més que permet modificar el símbol de l'ítem (el marcador) segons volguem aplicant els comandaments corresponents. De moment no cal que us preocupeu pels símbols ni la seva sintaxi ja que en parlarem d'ells al Capítol 3. Com a darrer exemple d'aquest apartat us mostrarem que també es poden fer llistes anidades, permetent-nos així aplicar la jerarquia que volguem a les nostres llistes.

### Exemple 2.6

```

1  \documentclass{article}
2
3  \begin{document}
4  \begin{itemize}
5      \item Primer ítem
6      \item Segon ítem
7      \begin{itemize}
8          \item Tercer ítem, aquest ítem es troba dins d'un altre \texttt{
            itemize}, per tant es troba en un nivell inferior i LaTeX li
            assigna un altre marcador i un grau d'indentació més.
9          \begin{itemize}
10             \item Quart ítem, aquest es troba a un nivell més baix
                encara
11             \item Cinquè ítem
12          \end{itemize}
13      \end{itemize}

```

```

14 \item Sise item
15 \end{itemize}
16 \end{document}

```

**Sortida:**

- Primer item
- Segon item
  - Tercer item, aquest item es troba dins d'un altre `itemize`, per tant es troba en un nivell inferior i LaTeX li assigna un altre marcador i un grau d'indentació més.
    - \* Quart item, aquest es troba a un nivell més baix encara
    - \* Cinque item
- Sise item

Notem com de forma automàtica LaTeX aplica un o altre marcador segons el nivell de l'ítem dins de la jerarquia. Amb comandaments més avançats es poden configurar quins són aquests marcadors per defecte per aquells que nosaltres volgüem.

## 2.4 Múltiples columnes

Per defecte LaTeX als seus tipus de documents més usuals aplica un format de text d'una sola columna. Amb aquest format ja ens és suficient per a poder treballar amb totes les eines que ens ofereix LaTeX, però existeixen comandaments que ens permeten treballar a dos o més columnes. Aquesta configuració és una mica delicada, en tenir com a condició que el compilador ha d'entendre el codi que escribim doncs hem d'adaptar algunes eines comunes al format de múltiple columna. Per exemple, els entorns per a col·locar figures o taules no funcionen correctament dins d'un entorn amb múltiples columnes. Tots aquests problemes acaben tenint una solució, més o menys complicada. En aquest apartat ens centrarem només en donar certs exemples de com cridar aquests entorns tant al preàmbul com en mig del document. A la resta del curs farem comentaris de si alguna funcionalitat en concret funciona correctament amb doble columna o no i alguna solució rellevant.

No és lo més usual, però com hem dit podem aplicar comandaments que afecten a tot el document ficant-los al preàmbul. En cas de voler aplicar doble columna a tot el document des del principi, podem posar a la configuració del tipus de document el paràmetre `twocolumn`. LaTeX entén per defecte que volem el text distribuït en dos columnes. A més també entén que aquesta distribució s'ha de completar per a omplir tot l'espai en blanc possible de l'amplada de línia. Si escribim a doble columna una quantitat de text que hauria d'omplir una columna sencera (és a dir, la meitat esquerra del document) només amb aquest comandament no sortirà el que s'esperaria. LaTeX no acaba la primera columna i, al arribar al límit inferior de la pàgina, salta a la següent columna. El que fa per defecte és directament omplir totes dues columnes alhora i de forma coherent. En el cas de tenir aquesta quantitat de text que hem dit abans, LaTeX el distribuïria en dues columnes que arribarien d'alçada fins a la meitat del document.

Per a treballar a doble columna (o amb més de dues, però això és més inusual) podem escriure dins de l'entorn `multicols`, del paquet `multicol`. Abans de poder fer-lo servir assegureu-vos que teniu el paquet declarat al preàmbul. Per a poder posar-vos exemples amb text suficient per a que es puguin apreciar les distribucions de columnes farem servir el clàssic *Lorem Ipsum*.<sup>4</sup>

### Exemple 2.7

```

1 \documentclass{article}
2
3 \begin{document}
4   Fora de l'entorn multicols el text es distribueix en una sola columna,
5   com es habitual. El que ve ara es troba distribuït en dues columnes:\

```

<sup>4</sup>El *Lorem Ipsum* és un text d'exemple en llatí que no té significat. Es fa servir per a ocupar espais on hauria d'haver text amb informació, és útil per a veure com s'acabaria veient algun format de text o com quedarien plantilles.

```

6 \begin{multicols}{2}
7   Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do
   eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim
   ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut
   aliquip ex ea commodo consequat. Duis aute irure dolor in
   reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla
   pariatur. Excepteur sint occaecat cupidatat non proident, sunt in
   culpa qui officia deserunt mollit anim id est laborum.
8 \end{multicols}
9 \vspace{4mm}
10 \noindent Tambe podem aplicar un format de tres columnes:\\
11
12 \begin{multicols}{3}
13   Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do
   eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim
   ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut
   aliquip ex ea commodo consequat. Duis aute irure dolor in
   reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla
   pariatur. Excepteur sint occaecat cupidatat non proident, sunt in
   culpa qui officia deserunt mollit anim id est laborum.
14 \end{multicols}
15 \end{document}

```

**Sortida:**

Fora de l'entorn multicols el text es distribueix en una sola columna, com es habitual. El que ve ara es troba distribuït en dues columnes:

<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo conse-</p>	<p>quat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>
--	--

Tambe podem aplicar un format de tres columnes:

<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud</p>	<p>exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Ex-</p>	<p>cepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</p>
--	---	---

De l'exemple 2.7 podem veure que per a indicar el nombre de columnes que volem a la nostra distribució del text només cal que posem el nombre a l'argument de l'entorn. No es pot fer explícitament en un exemple com l'anterior ja que dins d'una `tcolorbox`<sup>5</sup> no té sentit, però `multicols` accepta forçar la distribució del text a omplir primer la primera columna i, quan s'acabi l'espai, començar a omplir la segona (la distribució de text que potser esperavem en un principi, com hem comentat abans). Això s'indica escrivint `\begin{multicols*}{...}`. L'asterisc al declarar un entorn normalment significa ignorar alguna configuració específica de l'entorn, en aquest cas fa ignorar la redistribució del text en les columnes.

Naturalment la justificació i la configuració del format del text és compatible i s'aplica de la mateixa forma que amb el text distribuït en una sola columna. Com s'ha comentat abans, la justificació del text es fa respecte a l'amplada de línia, per tant si apliquem una justificació cap a la dreta, el text es distribueix en les columnes que haguem indicat i justificat al marge de columna dret més proper que trobi.

<sup>5</sup>Veure secció 6.2.1

## 2.5 Estructura per a un document científic

En la darrera secció d'aquest capítol farem servir totes les eines que hem après i aprendrem de noves amb l'objectiu de poder redactar el nostre primer document. Com la resta del curs aquestes eines i exemples són enfocades a un àmbit científic, però totes es poden fer servir en altres contextos molt diferents.

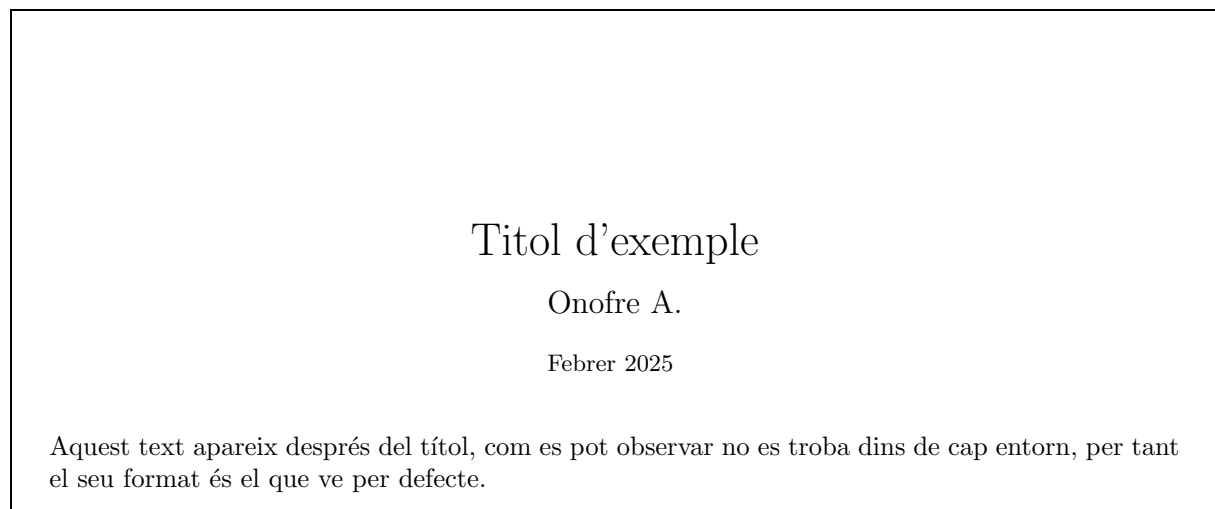
### 2.5.1 Pàgina de títol

En diverses ocasions ens veurem amb la necessitat de presentar un treball amb una portada i amb un índex on puguem veure el llistat de continguts del document. Jugant amb la justificació, l'espaiat i el format del text podem crear una pàgina inicial que servirà com a portada, la pàgina de títol. Posant el format del text amb negreta, canviant la mida i manualment posant números podem separar els diferents apartats en seccions. Normalment en un text científic separem la informació en cinc parts: resum o *abstract*, introducció i metodologia, resultats i discussió, conclusions, i bibliografia. Posem-ne primer un exemple de títol generat automàticament per LaTeX:

#### Exemple 2.8

```
1 \documentclass{article}
2
3 \title{Titol d'exemple}
4 \author{Onofre A.}
5 \date{Febrer 2025}
6
7 \begin{document}
8 \maketitle
9 Aquest text apareix després del títol, com es pot observar no es troba
10 dins de cap entorn, per tant el seu format es el que ve per defecte.
\end{document}
```

Sortida:



Veiem a l'exemple 2.8 que existeix el comandament `\maketitle`. Aquest comandament genera de forma automàtica un títol amb l'autor i la data agafant la informació que afegim al preàmbul amb els comandaments `\title`, `\author` i `\date`. Overleaf per defecte genera un document en blanc amb aquest títol agafant la informació que afegim al crear-lo. Com veiem el comandament genera un espaiat amb el marge superior, justifica al centre i separa la informació amb un salt de línia més gran de lo normal. Ens podem quedar amb aquest títol per defecte o crear els nostres propis títols amb les eines que us hem presentat abans.

En cas de voler separar el títol en una pàgina apartada de la resta del text, podem fer servir altres eines d'espaiat del text per a generar allò que realment volem o també un entorn específic per a generar una portada. Començem amb l'entorn. Podem indicar a LaTeX que el text personalitzat dins de l'entorn `\titlepage`. Aquest entorn aïlla el text que posem a dins a la primera pàgina, fent que la resta del text vingui a continuació a partir de la segona pàgina i deixant una “marca” que fa que el compilador entengui

que allò que hi ha a dins és una portada amb un títol<sup>6</sup>. Podem posar un exemple senzill de títol:

### Exemple 2.9

```

1  \documentclass{article}
2
3  \begin{document}
4  \begin{titlepage}
5      \centering
6      \vfill
7      {\LARGE \textsc{Titol personalitzat d'exemple}}\\
8      \vspace{4mm}
9      {\large \textbf{Autor:} Onofre A.}\\
10     {\large Optica't UAB}
11     \vfill
12 \end{titlepage}
13 Aquest text d'exemple comença a una nova pagina en blanc.
14 \end{document}

```

Sortida:

## TITOL PERSONALITZAT D'EXEMPLE

**Autor:** Onofre A.  
Optica't UAB

Aquest text d'exemple comença a una nova pagina en blanc.

LaTeX pot fer aquesta tasca de separar la informació al nostre gust de forma pràcticament automàtica segons el propòsit d'aquesta informació dins del text (com veurem a la secció 2.5.2). A més també és capaç de generar automàticament un índex ordenat i polit sense haver d'escriure res de forma explícita.

### 2.5.2 Capítols, seccions, subseccions i índex

Com hem dit a la secció anterior, podem separar el tipus d'informació segons el seu context o propòsit. Una de les formes més utilitzades per a separar la informació és l'ús dels comandaments que separen i jerarquitzen la informació. Us deixem una llista de tots els nivells en que es pot separar la informació, ordenats de major "importància" a menor dins del text:

- **part** - Pensat per a separar la informació en blocs molt grans, com per exemple per a separar un text en un bloc de teoria i un bloc d'exercicis. Les parts s'enumeren amb numeros romans, i el títol de cada part surt separat en una nova pàgina en blanc. Aquest comandament es pot fer servir en tipus de document com **book**, **report** o **article**.
- **chapter** - Separa la informació en blocs grans, per capítols. És possible configurar-ho per a que el títol del capítol estigui en una pàgina separada o que seguidament comenci el text del document. Cada capítol ve enumerat per un nombre natural, començant per l'u. Aquest comandament només es pot utilitzar en aquells tipus de document com **book** i **report**.
- **section** - Separa el text en seccions que, per exemple, poden separar el cos d'un capítol. En textos científics normalment serveix per a separar el document en les 5 parts que hem mencionat anteriorment. Quan una secció es troba en un tipus de document que no se separa per capítols (com en un **article**) la secció s'enumera amb un nombre natural, començant per l'u. Quan una

<sup>6</sup>Aquesta informació extra per al compilador sembla irrellevant però pot ser útil per a algunes automatitzacions com per exemple per a modificar de forma més senzilla com es contabilitzen les pàgines del document. Si fem entendre a LaTeX que existeix una portada, podem fer que entengui que aquella pàgina no s'ha d'indexar amb un nombre de pàgina.



secció es troba dins d'un codument que se separa per capítols, l'enumeració de la secció s'estructura com  $n.m$ , on  $n$  és el nombre del capítol on es troba la secció i  $m$  l'enumeració de la secció dins del capítol. Per exemple, la secció 3.7 d'un document és aquella que es troba al tercer capítol del document i ocupa la setena posició dins de totes les seccions que hi ha dins d'aquell capítol.

- **subsection** - Seguint la lògica que hem vist a **section**, aquesta separació d'un nivell més baix es comporta exactament igual però s'enumera amb el format  $m.k$  si es troba en un document sense capítols i amb el format  $n.m.k$  si es troba en un document separat per capítols. Anàlogament al punt anterior, cada índex correspon a un nivell de separació, el més alt es troba a l'esquerra i va baixant anant cap a la dreta. Per exemple, la subsecció 1.5.2 pertany al primer capítol, secció cinc i la segona de les subseccions que conté aquesta secció. Un altre exemple, si no hi ha capítols, la subsecció 2.3 pertany a la segona secció del document i ocupa la tercera posició de totes les subseccions que conté la secció on es troba.
- **subsubsection** - Aquesta darrera separació es comporta igual que les dues anteriors només que en un nivell més baix. La seva enumeració depèn de la configuració del document. Si treballem amb documents com **article** la subsubsecció apareixerà enumerada amb el format  $m.k.j$  (secció, subsecció i subsubsecció). Si treballem amb documents que permeten separar per capítols normalment les subsubseccions apareixen sense enumerar. El motiu d'això és la configuració per defecte d'aquest tipus de document, amb allò que podem anomenar “profunditat de nivell”. Si es vol podem canviar això i fer que les subsubseccions apareixin enumerades amb el format  $n.m.k.j$  (capítol, secció, subsecció i subsubsecció) només cal canviar la configuració dins del preàmbul afegint el comandament `\setcounter{secnumdepth}{3}`. El 3 indica el nivell màxim al que volem que aparegui enumerat, en aquest cas **subsubsection** es troba al tercer nivell. Les anteriors separacions també tenen enumerat el seu nivell, -1 (part), 0 (capítol), 1 (secció) i 2 (subsecció). Notem com la profunditat de nivell d'un **article** és per defecte 3 mentre que per a un **book** és 2
- **paragraph** - Els nivells que queden normalment no s'utilitzen ja que amb les separacions de línia que hem explicat abans ja és suficient. Igualment aquesta separació correspon al nivell 4, per tant ve sense enumerar per defecte. Una diferència amb la resta de separacions és que no separa el text que fem a l'argument del comandament amb la resta del text, només el posa en negreta. A la pràctica el funcionament d'aquesta separació és com posar el text en negreta.
- **subparagraph** - Aquesta és la darrera separació bàsica d'un text en LaTeX, es comporta pràcticament com un text normal en negreta. Es diferencia del paràgraf en moments concrets com per exemple al col·locar-se després d'un objecte que no és un paràgraf, per exemple una imatge o un llistat. Un paràgraf després d'un objecte apareix en negreta i no pateix indentació inicial, un subparagraph si que se li aplica aquesta indentació. El nivell del subparagraph, naturalment, és el 5.

LaTeX entén que l'enumeració de cada separació és la que correspon a l'ordre d'aparició en el codi. Per exemple, podem tenir dues seccions consecutives anomenades “Secció A” i “Secció B”. En un primer moment escrivim la “Secció A” i després la “Secció B” i quan compilem apareixen enumerades amb un 1 per a la A i un 2 per a la B. Si temps després volem canviar d'ordre, és a dir, que apareixi primer al text la “Secció B” i després la “Secció A” podem moure dins del codi cada secció sense cap problema. Ara bé, ara l'enumeració s'haurà actualitzat, ara amb el canvi d'ordre les seccions s'enumeren amb un 1 per a la B i un 2 per a la A. A més, llevat dels nivells -1 (part) i 0 (capítol)<sup>7</sup>, totes les separacions són compatibles amb el format de dos (o més) columnes.

Una altra característica de les separacions del text és que els nivells del -1 (part) al 2 (subsection) l'argument que escrivim es troba separat del text normal en forma de títol amb una mida de lletra diferent. Depèn del tipus de document que escollim, però la mida de cada separació normalment és: -1 (part) **huge**, 0 (capítol) **LARGE**, 1 (secció) **Large**, 2 (subsecció) **large** i la resta de nivells **normalsize**, sense separar-se com a un títol.

Aquestes separacions que acabem de descriure són molt útils per a poder-nos moure pel document de forma ràpida, fins i tot si és un document molt llarg. Overleaf té una part de la seva pantalla de treball dedicada a l'ordre i mobilitat dins del document. En aquesta part podrem observar en forma de llistat

<sup>7</sup>Aquests nivells són incompatibles amb l'entorn **multicols**, però si les múltiples columnes estan configurades al preàmbul i afecten a tot el text si que és pot fer servir **part** i **chapter**. Les incompatibilitats apareixen, per exemple, al intentar ficar un **chapter** dins d'un **multicols**.

desplegable totes les separacions que haguem posat al nostre document. Aquest desplegable ens permet navegar entre capítols, seccions i subseccions de forma ràpida.

Un del avantatges més notables de separar la informació d'aquesta forma és que LaTeX és capaç de crear un índex de forma automàtica amb només un comandament: `\tableofcontents`. Aquest comandament es posa dins del text, situat a la pàgina on vulguem situar l'índex. La posició de l'índex dins del text és indiferent, la informació que emmagatzema Overleaf per a poder generar el navegador que hem mencionat abans és la mateixa que ens crea l'índex. Com exemple d'índex generat d'aquesta manera podeu mirar l'índex d'aquestes notes del Curs de LaTeX, només que nosaltres hem ficat una característica més que explicarem més endavant, el nostre índex permet navegar pel mateix pdf seleccionant la part a on es vol anar (veure secció 6.2.4).

### 2.5.3 Footnotes i etiquetes

Com segurament haureu vist ja diverses vegades al llarg d'aquestes notes de tant en tant fem aclaracions al text que estem redactant<sup>8</sup>. Es tracta de notes a peu de pàgina o *footnotes* i aplicat a text normal és molt fàcil de fer servir. Només cal fer escriure el comandament `\footnote{...}`, que és un comandament doblement local, és a dir, que actua justament al lloc on el posam al codi. L'argument del comandament és directament el text que sortirà a peu de pàgina i, amb algunes limitacions, es pot emplenar com si estiguessim escrivint al document de forma normal. Podem fins i tot afegir objectes matemàtics (veure Capítol 3).

Pel fet de ser un comandament doblement local s'ha de tenir cura d'on situem una *footnote*, ja que és un comandament bastant sensible a incompatibilitats. Per exemple, com veurem més endavant, posar una *footnote* dins d'un títol d'una taula o figura (anomenat *caption*, veure Capítol 4) és una mica més complicat que simplement afegir allà el comandament de nota a peu de pàgina. Ho explicarem amb més detall més endavant al detallar l'ús de figures i taules, però fem un petit avenç per a explicar com solucionar aquest problema. Objectes com un *caption* es troben dins d'entorns anomenats entorns flotants. Aquests entorns pateixen unes normes especials a l'hora de posicionar els objectes que afecta dins del document, per qüestions d'optimització d'espai. El comandament `footnote` és sensible a entorns flotants, és per això que en aquest tipus d'espai dins del codi hem de separar el marcador i el text de la *footnote*. El marcador és el número que indexa a la *footnote*, és l'objecte que funciona de forma doblement local, el número es queda allà on l'hem posicionat dins del text. És després, quan es compila el document, que la informació d'aquest marcador (posició al text, número que li correspon...) es relaciona amb el text que volem que mostri (el cos de la *footnote*). Per a separar el marcador del text fem servir el comandament `\footnotemark`. Més endavant, podem declarar el text que anirà en aquesta *footnote* escrivint el comandament `\footnotetext{...}`. Aquest darrer comandament, com es pot esperar, s'ha d'escriure a fora de l'entorn flotant.

Les notes a peu de pàgina es poden configurar de diverses maneres, fins i tot utilitzant paquets. Però, LaTeX té algunes configuracions senzilles per defecte. Es pot observar al llarg d'aquestes notes com l'índex de les *footnotes* es reinicia a cada capítol que passa. Aquest comportament és per defecte i ajuda a no acumular un nombre molt gran de notes. Podem manipular l'índex que mostra el marcador, no només amb números, sinó també fent servir símbols.

Per a reiniciar el comptador dels índex que estem fent servir (números, lletres o símbols) només cal escriure allà on vulguem que passi el comandament

$$\text{\setcounter{footnote}{0}}$$

Aquest tipus de comandament és un de nou, s'encarrega de canviar la configuració d'una característica de LaTeX fora del preàmbul. Podem anomenar a aquest tipus de comandaments com a comandaments de configuració. Notem com aquest comandament té doble argument. `setcounter` en el seu primer argument selecciona el llistat d'índex que actualment fa servir l'objecte `footnote`, en el seu segon argument declara quina posició de la llista d'índex es vol seleccionar a partir d'aquell moment. En aquest cas el que fa és reiniciar el comptador de números a la posició 0 (o posició 1, és indiferent) de la llista de números naturals que fa servir LaTeX. La pròxima *footnote* que aparegui al text apareixerà amb un 1 com a marcador.

---

<sup>8</sup>Com per exemple aquesta aclaració que esteu llegint ara.

Per a poder fer servir marcadors que no siguin números hem de fer servir un altre comandament de configuració:

```
1 \renewcommand{\thefootnote}{\fnsymbol{footnote}}
```

per a que els marcadors siguin símbols,

```
1 \renewcommand{\thefootnote}{\alph{footnote}}
```

per a que siguin lletres minúscules,

```
1 \renewcommand{\thefootnote}{\roman{footnote}}
```

per a que siguin nombres romans.

Els comandaments de configuració del tipus `renewcommand` normalment el que fan és substituir alguna configuració per defecte de LaTeX o generar una nova ordre. En aquest cas el que fa és substituir la llista de marcadors que hi ha per defecte, passem de la llista de números naturals a una llista amb l'abecedari, una altra amb nombres romans i una altra amb 9 símbols diferents. S'ha de tenir en cura de que, per a llistes finites com l'abecedari o els símbols no fiquem més *footnotes* que elements hi hagi a la llista de marcadors. En cas d'haver posat 9 notes fent servir el llistat de 9 símbols com a marcadors, haurem de reiniciar manualment el comptador o començar a fer servir altres marcadors per a les següents *footnotes*.

Haver de preocupar-se d'un nombre limitat de *footnotes* pot ser una mica farragós. Per aquest motiu existeixen eines externes a LaTeX bàsic en forma de paquet que ens poden ser de molta utilitat. Us deixem un exemple d'un d'aquests paquets i una possible configuració útil. En cas de voler fer servir com a marcadors la llista de símbols i voler configurar el document per a que a cada nova pàgina el comptador de marcadors torni a zero, podem utilitzar el paquet `footmisc` amb la següent configuració al preàmbul:

```
1 \usepackage[perpage, symbol]{footmisc}
2 \renewcommand{\thefootnote}{\fnsymbol{footnote}}
```

Podem notar com el primer comandament crida un nou paquet amb una certa configuració (aquella que permet reiniciar el comptador per cada pàgina i aquell que ho fa compatible amb els símbols). El segon comandament canvia la configuració del document com hem vist abans.

Per últim, també podem forçar a que aparegui un marcador concret a una footnote específica. Només cal aplicar un segon argument al comandament `footnote`, indicant la posició a la llista de marcadors d'aquell que vulguem fer servir. Per exemple, si vull fer servir la lletra b com a marcador de la primera *footnote* del meu text, he d'escriure primer el comandament que canvia la llista per defecte a la llista de lletres i després fer servir, allà on toqui, el comandament: `\footnote[2]{Text exemple}`.

## Etiquetes i referències

Novament dins del context d'un document científic normalment és necessari referenciar parts del document al mateix text. LaTeX per defecte té una eina que ens facilita la tasca d'indexar correctament aquestes referències creuades. El comandament `\label{...}` és un comandament doblement local que es fa servir per a marcar objectes del nostre document i els hi col·loca una etiqueta. Tenim llibertat completa per a escollir quina serà la nostra etiqueta, és a dir que l'argument de `label` és completament lliure ja que s'omple amb text pla. Aquestes etiquetes el que fan és:

1. Reconèixer quin és l'objecte del seu voltant al qual volem aplicar una referència.
2. Cercar quina és la posició relativa d'aquest objecte dins del document i indexar-li un número natural.
3. Assignar com a etiqueta o nom el text que fiquem a l'argument.

Fixem-nos que el primer ítem de la llista és crític a l'hora de fer servir aquest comandament. Això és deu a que `label` funciona correctament amb objectes de LaTeX, no amb qualsevol troç de codi on el fiquem. LaTeX ha de reconèixer que existeix un objecte predefinit al preàmbul per a poder categoritzar-lo. Això es pot veure millor amb els exemples que venen per defecte a LaTeX. Sense que calgui declarar-los a la configuració del preàmbul existeixen per defecte alguns objectes com taules, imatges, equacions, etc. Aquests objectes són diferents i tenen un paper particular dins del text que estem redactant, per tant té sentit que cada tipus d'objecte estigui enumerat de forma independent. LaTeX té en compte aquesta caracterització dels objectes i enumera per separat les taules o les equacions que hi ha al document.

El comandament `label` aprofita aquesta informació per a seguir els tres punts que hem indicat abans. Naturalment si canviem un objecte de lloc els índex s'actualitzen i no cal que estiguem pendents de quin numero li toca a cada objecte.

A més de considerar l'enumeració, el comandament `label` afegeix més informació a l'objecte sobre el que actua i li annexa el nom que nosaltres posem a l'argument. Dins del flux de treball pensat per a LaTeX existeix un esquema útil per a identificar fàcilment els noms que anem posant. És senzill, només cal separar per objectes els noms que posem a les etiquetes, e.g. “tau” quan sigui una taula, “fig” per a una figura, “sec” per a una secció... Posem un exemple de com etiquetem un capítol d'aquestes notes:

```
1 \chapter{L'entorn matemàtic}\label{cap:entorn_matematic}
```

Ara que ja sabem com posar etiquetes com cal podem veure quina és la seva utilitat. Un com col·locada una etiqueta a un objecte podem referenciar-la a qualsevol part del text escrivint `\ref{...}`. Aquest comandament també es doblement local i el seu argument s'omple amb el nom de l'etiqueta que volem referenciar. El fet que puguem posar les referències en qualsevol lloc del text (ja sigui abans o després d'on es troba col·locat el `label`) fa que puguem parlar de referències creuades. Aquesta característica de LaTeX, un cop dominada, és de lo més útil que podem trobar ja que ens treu de sobre la preocupació d'indexar correctament les referències. Només cal tenir una norma clara, senzilla i fàcil de recordar per a posar nom a les etiquetes. Més endavant veurem un comandament similar per a poder ficar cites creuades de la bibliografia (veure Capítol 5).

Com a darrera observació d'aquesta secció, en aquestes notes fem servir de forma recurrent les etiquetes per a referenciar parts del text, és a dir, seccions, capítols, etc. Els separadors que hem vist a la subsecció anterior també són objectes independents dels altres, amb la característica que la seva enumeració prové de la propia jerarquia que ja hem explicat. A més, les referències que nosaltres posem en aquestes notes es comporten com a enllaços (com vam veure amb les *footnotes*). Per defecte això no passa, més endavant al text explicarem com implementar aquests hiperenllaços. De fet existeixen paquets que permeten modificar de diverses maneres el comportament de les etiquetes i de les referències, al igual que també existeixen incompatibilitats entre les referències per defecte i alguns paquets, però això és millor trobar-s'ho un mateix.

## Capítol 3

# L'entorn matemàtic

En aquest capítol ens dedicarem a aprofundir una mica en la integració d'expressions matemàtiques en els nostres documents. Com la resta de capítols, el tractament que farem d'aquest tema no serà exhaustiu, ja que de ser-ho el document seria interminable. Com es pot deduir, aquest capítol és un dels més enfocats en redactar documents de caire científic (sobre tot en branques com les matemàtiques o la física). Tot i això, es tracta d'un capítol molt interessant per a aprendre a fer servir amb més comoditat els comandaments i entendre com es relacionen els diferents tipus d'objectes que ofereix LaTeX. Una de les parts més interessants per a aquelles persones que no vulguin fer documents científics és la part on expliquem com funcionen les matrius, ja que tenen un comportament molt semblant al de les taules.

### 3.1 Formes de cridar un entorn matemàtic

Una de les funcionalitats més polides que té LaTeX és la de poder escriure expressions matemàtiques de forma elegant i senzilla. A diferència d'altres editors de textos, LaTeX no requereix d'un lloc on cercar els símbols que necessitem copiar i enganxar continuament al document. Com es pot esperar a aquesta alçada del document, els símbols matemàtics es criden amb comandaments, de fet ja hem pogut veure algun exemple en l'anterior capítol. Per a que LaTeX pugui entendre que el que escrivim és una expressió matemàtica i no text pla podem cridar l'entorn matemàtic. De fet, podem cridar aquest entorn de més d'una forma i amb característiques diferents. Tractem ara les formes més bàsiques de cridar l'entorn matemàtic.

Com a primera aproximació introduïm el delimitador `$`. Aquest símbol serveix per a ficar expressions matemàtiques dins del paràgraf on estiguem escrivint<sup>1</sup>. Veiem-ne un exemple:

#### Exemple 3.1

1 Aquest text es un text d'exemple que serveix per a comprovar un entorn matemàtic. Una de les expressions mes utilitzades en matèries com la física es al teorema de pitagores, que s'expressa com  $a^2 + b^2 = c^2$ .

#### Sortida:

Aquest text es un text d'exemple que serveix per a comprovar un entorn matemàtic. Una de les expressions mes utilitzades en matèries com la física es al teorema de pitagores, que s'expressa com  $a^2 + b^2 = c^2$ .

Com es pot observar, el text que es troba entre `$` acaba adoptant un format diferent al del text pla. Les lletres acaban imprimint-se en cursiva i hi ha símbols reservats per a expressar la sintaxi matemàtica que correspon a la fórmula de Pitàgores. De la sintaxi d'aquest entorn ens encarregarem en la següent secció.

Una altra forma de poder cridar l'entorn matemàtic dins d'un paràgraf és escrivint els delimitadors `\( i \)`. Aquests dos delimitadors actuen d'igual forma que els delimitadors `$`. Són delimitadors intercanviables, però, no es poden combinar. Començar un entorn matemàtic amb un `$` i acabar-ho amb un `\)` és un error de sintaxi per a LaTeX, per tant en Overleaf no escriurà res (ho ignorarà) i en altres entorns

<sup>1</sup>Aquesta característica, com veurem d'aquí uns moments és rellevant.

de treball directament no es compilarà el document.

Aquestes dues formes de poder ficar fòrmules són molt convenients quan per exemple volem mencionar variables que apareguin en el nostre treball. Però, si intentem ficar una expressió molt gran el paràgraf s'acaba veient molt atapeït i no queda tan polit com pot quedar fer servir altres entorns matemàtics.

### Exemple 3.2

```
1 Aquest paragraf serveix com a exemple de quan es posa una expressio molt
   gran dins d'un paragraf. L'exemple escollit es la generalitzacio del
   calcul de l'hipervolum d'una $n$-esfera en una geometria de norma $p$:
   $V_{\{n\}}^{\{p\}}(R) = (2R)^{\{n\}} \frac{\{\Gamma(1+1/p)\}^{\{n\}}{\{\Gamma(1+n/p)\}}$
```

#### Sortida:

Aquest paragraf serveix com a exemple de quan es posa una expressio molt gran dins d'un paragraf. L'exemple escollit es la generalitzacio del calcul de l'hipervolum d'una  $n$ -esfera en una geometria de norma  $p$ :  $V_n^p(R) = (2R)^n \frac{\Gamma(1+1/p)^n}{\Gamma(1+n/p)}$

Es pot veure de l'Exemple 3.2 com l'expressió matemàtica s'acaba imprimint correctament, però es veu petita i encaixada dins del paràgraf. Existeixen expressions matemàtiques encara més farragoses que si les posem dins d'un paràgraf poden espatllar com es llegeix el text. És per això que us introduïm dues formes més per a cridar l'entorn matemàtic fora dels paràgrafs.

Per a fer això, només cal cridar l'entorn `equation`. Aquest entorn s'encarrega de que el que fiquem a dins segueixi la sintaxi matemàtica, que surti centrat i enumerat (amb enumeració dinàmica, com és natural és una indexació que permet fer servir referències creuades). Posem-ne un exemple per a escriure la mateixa expressió que a l'Exemple 3.2:

### Exemple 3.3

```
1 Aquest paragraf serveix com a exemple de text que va abans d'una
   expressio matematica. Tornem a presentar el calcul de l'hipervolum d'una
   $n$-esfera en una geometria de norma $p$:
2 \begin{equation}
3   V_{\{n\}}^{\{p\}}(R) = (2R)^{\{n\}} \frac{\{\Gamma(1+1/p)\}^{\{n\}}{\{\Gamma(1+n/p)\}}
4 \end{equation}
```

#### Sortida:

Aquest paragraf serveix com a exemple de text que va abans d'una expressio matematica. Tornem a presentar el calcul de l'hipervolum d'una  $n$ -esfera en una geometria de norma  $p$ :

$$V_n^p(R) = (2R)^n \frac{\Gamma(1 + 1/p)^n}{\Gamma(1 + n/p)} \quad (3.1)$$

Observem com de forma automàtica l'equació s'enumera amb el nombre del capítol seguit del nombre de l'equació (com és la primera que surt indexada, el seu nombre és un u). Aquesta forma de presentar equacions és més polida i permet més marge de maniobra a l'hora d'espaiar els símbols com vulguem. És notable com l'expressió es veu menys atapeïda escrita de la segona manera. En cas que vulguem escriure així una expressió matemàtica però aquesta no tingui massa rellevància al text com per a ser enumerada podem aplicar un al comandament `equation*`. Com hem comentat en capítols anteriors, un asterisc usualment s'encarrega d'anul·lar alguna característica d'un comandament. Veiem ara com queda el mateix exemple sense enumerar:

### Exemple 3.4

```
1 Aquest paragraf serveix com a exemple de text que va abans d'una
   expressio matematica. Tornem a presentar el calcul de l'hipervolum d'una
   $n$-esfera en una geometria de norma $p$ pero ara sense enumerar l'
   equacio:
2 \begin{equation*}
3   V_{\{n\}}^{\{p\}}(R) = (2R)^{\{n\}} \frac{\{\Gamma(1+1/p)\}^{\{n\}}{\{\Gamma(1+n/p)\}}
```

```
\end{equation*}
```

**Sortida:**

Aquest paràgraf serveix com a exemple de text que va abans d'una expressió matemàtica. Tornem a presentar el càlcul de l'hipervolum d'una  $n$ -esfera en una geometria de norma  $p$  però ara sense enumerar l'equació:

$$V_n^p(R) = (2R)^n \frac{\Gamma(1 + 1/p)^n}{\Gamma(1 + n/p)}$$

Dues formes alternatives per a col·locar una equació sense enumerar és aplicar els delimitadors `\[` i `\]` o bé cridar l'entorn `displaymath`. L'elecció d'una o altra forma de cridar un entorn matemàtic sense enumerar és la compatibilitat amb els objectes o paquets que fem servir. Per exemple, pot passar que posem algun paquet que afecti a tot el document i imposi un tipus de font i justificació a tot el text. És possible que aquest paquet per si sol (o si ha estat mal configurat) afecti al cridar `equation*`, fent que doni error. En aquest cas exemple hauríem de fer servir alguna de les alternatives per a fer el que volíem des d'un principi.

La col·locació d'un `label` en un entorn `equation` (enumerat) ha de ser justament després d'escriure el primer delimitador de l'entorn `equation`, és a dir, s'ha d'escriure `\{equation\}\label{...}`. L'etiqueta de l'entorn matemàtic mostrarà el nombre que marca l'equació al seu costat. En cas que sigui una equació sense enumerar com a l'Exemple 3.3 el `label` no tindrà informació sobre el posicióament relatiu de l'equació i no mostrarà cap referència.

En cas de voler treballar amb més d'una columna, els entorns que hem presentat es fan servir de la mateixa forma que en una sola columna. L'únic problema és l'espai, hi ha expressions matemàtiques llargues que potser no tenen espai suficient al treballar amb, per exemple, dues columnes. Veiem-ne un exemple de l'error i una possible solució:

**Exemple 3.5**

```
1 \begin{multicols}{2}
2   Com sabem, si treballem dins d'aquest entorn podem treballar amb
   multiples columnes. Com a exemple d'expressió llarga posarem la
   Segona Llei de Newton de forma explícita i fent servir diverses
   notacions:
3   \begin{equation*}
4     \vec{F}(\vec{r},t) = m \vec{a}(\vec{r},t) = m \frac{\text{d}}{\text{d}t} \vec{v}(\vec{r},t) = m \frac{\text{d}^2}{\text{d}t^2} \vec{r}(t) = m \ddot{\vec{r}}(t)
5   \end{equation*}
6 \end{multicols}
```

**Sortida:**

Com sabem, si treballem dins d'aquest entorn podem treballar amb múltiples columnes. Com a exemple d'expressió llarga posarem la Segona Llei de Newton de forma explícita i fent servir diverses notacions:

$$\vec{F}(\vec{r}, t) = m\vec{a}(\vec{r}, t) = m \frac{d\vec{v}}{dt}(\vec{r}, t) = m \frac{d^2\vec{r}}{dt^2}(t) = m\dot{\vec{v}}(t) = m\ddot{\vec{r}}(t)$$

És evident que l'expressió matemàtica que apareix a l'Exemple 3.5 no té espai dins de la columna on s'ha escrit, de fet, surt fora del requadre en que enmarquem els exemples. Per a poder arreglar aquest inconvenient podem fer servir el paquet `amsmath`. Aquest paquet és molt útil per a treballar amb expressions matemàtiques. Entre altres coses, permet fer una cosa que LaTeX pur no deixa fer directament dins de l'entorn `equation`, permet fer servir `\[` com a salt de línia:

**Exemple 3.6**

```
1 \begin{multicols}{2}
2   Tornem a escriure exactament el mateix exemple d'abans però amb l'
   ajuda del paquet \texttt{amsmath}. Com a exemple d'expressió llarga
   posarem, novament, la Segona Llei de Newton de forma explícita i
   fent servir diverses notacions:
```



```

3      \begin{multline}
4      \vec{F}(\vec{r},t) = m \vec{a}(\vec{r},t) = m \frac{\text{d}}{\text{d}t} \vec{v}(\vec{r},t) = m \frac{\text{d}^2}{\text{d}t^2} \vec{r}(t) = m \dot{\vec{v}}(t) = m \ddot{\vec{r}}(t)
5      \end{multline}
6      \end{multicols}

```

**Sortida:**

Tornem a escriure exactament el mateix exemple d'abans però amb l'ajuda del paquet **amsmath**. Com a exemple d'expressió llarga posarem, novament, la Segona Llei de Newton de

$$\begin{aligned} \vec{F}(\vec{r},t) &= m\vec{a}(\vec{r},t) = m\frac{d\vec{v}}{dt}(\vec{r},t) = \\ &= m\frac{d^2\vec{r}}{dt^2}(t) = m\dot{\vec{v}}(t) = m\ddot{\vec{r}}(t) \quad (3.2) \end{aligned}$$

Fixant-nos en el codi que hi ha a l'Exemple 3.6, enmig de les expressions matemàtiques<sup>2</sup> apareix escrit `= \frac{d}{dt} =`. A més, s'ha substituït l'entorn `equation` per l'entorn `multline`. Com a resultat, l'expressió llarga que se sortia dels marges de l'Exemple 3.5 es divideix justament al lloc on hem col·locat les dues contrabarres i fa un salt de línia. La primera meitat està justificada a l'esquerra i la segona a la dreta, però, si hi haguessin més salts de línia, les expressions intermedies tindrien una alineació central. Hem col·locat dos símbols d'igual per a que apareguin escrits, però no afecten en res a l'hora de fer el salt de línia, aquest es pot posar allà on es vulgui sempre i que no trenquem algun objecte que no sigui compatible amb un d'aquests salts (per exemple dins de l'argument d'algun comandament o símbol matemàtic). En cas que vulguem una equació amb `multline` sense enumerar només cal posar el comandament amb un asterisc.

Aquesta solució és una bàsica que us presentem en aquesta secció en forma d'exemple. Però, el paquet **amsmath** conté entorns on es pot treballar fent diverses línies d'expressions matemàtiques amb diferent alineació cada una. Aprofitarem una part de la següent secció per a explicar-vos ho en detall.

## 3.2 Sintaxi de l'entorn matemàtic

Com ja hem pogut observar en alguns exemples, les expressions matemàtiques escrites en LaTeX poden arribar a confondre molt per a les persones que mai han vist una sintaxi semblant. Potser no ho sembla, però escriure aquestes expressions matemàtiques acaba sent bastant repetitiu i, un cop us acostumeu, saber o no saber com escriure certa expressió matemàtica serà el menor dels vostres problemes a l'hora d'escriure un document en LaTeX.

Començem aquest apartat indicant algunes característiques bàsiques de la sintaxi matemàtica de LaTeX. El primer que hem de considerar al escriure en un entorn matemàtic és que no existeixen els espais, és a dir, escriure `$a + b$` i escriure `$a+b$` és completament equivalent. Qualsevol de les dues expressions anteriors imprimeix  $a + b$  per pantalla. Com hem dit, les lletres que fem servir per a escriure en text pla, dins d'un entorn matemàtic surten en cursiva. Intentar escriure una frase normal en aquests entorns fa que quedi visualment malament:

`$Frase exemple dins d'un entorn matematic$`  $\longrightarrow$  *Fraseexempldinsd'unentornmatematic*

Un altra canvi respecte a escriure en text pla és que les lletres no accepten accents, en cas d'escriure una lletra accentuada LaTeX dona error de sintaxi. Aquest tractament de l'alfabet està pensat per a tractar a cada lletra com a una variable matemàtica.

Al principi pot semblar que aquest entorn és molt restrictiu, però està pensat per a facilitar la feina per a escriure equacions. En cas de voler aplicar més personalització espacial podem fer servir els comandaments `\hspace` i `\vspace` que ja coneixem<sup>3</sup>. Si, com hem vist en els Exemples 3.5 i 3.6, ens quedem sense espai al escriure una expressió matemàtica podem fer ús de dos entorns molt convenients que provenen

<sup>2</sup>No us preocupeu ara de si es veu complicada la sintaxi d'aquestes fórmules, això ho veurem al següent apartat i us acostumareu molt ràpid.

<sup>3</sup>Tot i que el segon (`\vspace`) no se sol fer servir en expresions que ocupen una o dues línies. Un possible ús d'aquest comandament pot ser quan es construeixen matrius, però igualment és poc usual.



del paquet `amsmath`. El primer entorn que presentem, `split`, serveix per a quan tenim una única equació o expressió matemàtica molt llarga i la volem partir i alinear com cal; el segon entorn, `align`, serveix per a quan tenim més d'una equació i les volem col·locar ben alineades en el mateix bloc al document. Veiem-ne primer exemples d'ús dels dos entorns i posteriorment expliquem que fa cada un:

### Exemple 3.7

```

1  En aquest primer bloc matemàtic apliquem l'entorn \texttt{split} per a
    una expressió matemàtica molt llarga. Posem com a exemple una expressió
    intermitja (sense simplificar) del càlcul d'una component del tensor d'
    inèrcia d'un triangle rectangle que rota sobre el seu vertex on te l'
    angle recte:
2  \begin{equation}
3      \begin{split}
4          J_{11} = & \frac{\sigma}{3} \left( a^4 \sin^3 \alpha \cos \alpha - \frac{3}{2} a^4 \tan \alpha \sin^2 \alpha \cos^2 \alpha + \right. \\
& \left. a^4 \cos^3 \alpha \sin \alpha \tan^2 \alpha - \frac{1}{4} a^4 \tan^3 \alpha \cos^4 \alpha \right) \\
5      \end{split}
6  \end{equation}
7  En aquest segon bloc matemàtic farem servir l'entorn \texttt{align}.
    Aquest comandament normalment es fa servir per a alinear més d'una
    equació. L'exemple anterior es pot simplificar si sabem el valor de  $\sigma$ .
    Podem aprofitar-ho com es veu al bloc que ve a continuació:
8  \begin{align}
9      J_{11} &= \frac{\sigma}{3} \left( a^4 \sin^3 \alpha \cos \alpha - \frac{3}{2} a^4 \tan \alpha \sin^2 \alpha \cos^2 \alpha + a^4 \cos^3 \alpha \sin \alpha \tan^2 \alpha - \frac{1}{4} a^4 \tan^3 \alpha \cos^4 \alpha \right) \\
10     J_{11} &= \frac{1}{6} M a^2 \sin^2 \alpha
11 \end{align}

```

### Sortida:

En aquest primer bloc matemàtic apliquem l'entorn `split` per a una expressió matemàtica molt llarga. Posem com a exemple una expressió intermitja (sense simplificar) del càlcul d'una component del tensor d'inèrcia d'un triangle rectangle que rota sobre el seu vertex on te l'angle recte:

$$J_{11} = \frac{\sigma}{3} \left( a^4 \sin^3 \alpha \cos \alpha - \frac{3}{2} a^4 \tan \alpha \sin^2 \alpha \cos^2 \alpha + a^4 \cos^3 \alpha \sin \alpha \tan^2 \alpha - \frac{1}{4} a^4 \tan^3 \alpha \cos^4 \alpha \right) \quad (3.3)$$

En aquest segon bloc matemàtic farem servir l'entorn `align`. Aquest comandament normalment es fa servir per a alinear més d'una equació. L'exemple anterior es pot simplificar si sabem el valor de  $\sigma$ . Podem aprofitar-ho com es veu al bloc que ve a continuació:

$$J_{11} = \frac{\sigma}{3} \left( a^4 \sin^3 \alpha \cos \alpha - \frac{3}{2} a^4 \tan \alpha \sin^2 \alpha \cos^2 \alpha + a^4 \cos^3 \alpha \sin \alpha \tan^2 \alpha - \frac{1}{4} a^4 \tan^3 \alpha \cos^4 \alpha \right) \quad (3.4)$$

$$J_{11} = \frac{1}{6} M a^2 \sin^2 \alpha \quad (3.5)$$

L'entorn `split` és un entorn que per a poder cridar-lo és necessari que se situï dins d'un entorn `equation` o `equation*`. Serveix per a poder tallar una expressió llarga per on vulguem (igual que l'entorn `multline`) però a més permet alinear els troços que anem separant per allà on vulguem. Cada troç de l'expressió que separem accepta un `&`, un símbol que marca en quin lloc del troç d'expressió s'alinearan. Com només s'encarrega de tallar, tota l'expressió matemàtica que fiquem quedarà enumerada pel mateix índex.

En canvi, l'entorn `align` és un entorn matemàtic per si sol, no necessita que es col·loquem dins de cap altre entorn. Aquest serveix per a escriure diferents equacions o expressions de forma separada pero amb la capacitat de ser alineades allà on vulguem. Es fa servir de forma molt similar a `split`, l'alineació funciona de la mateixa manera. Però, una diferència clau és que `align` separa cada expressió com a una equació diferent, per tant, enumera cada troç separat amb un índex diferent<sup>4</sup>.

### 3.2.1 Operacions i relacions matemàtiques

### 3.2.2 Parèntesis i claudàtors

### 3.2.3 Alguns accents, símbols i formats útils

Expressió d'unitats

### 3.2.4 Límits, sumatoris, productoris i integrals

## 3.3 Matrius i alineació

## 3.4 Exemples

---

<sup>4</sup>Notem com l'expressió és massa llarga com per a que l'enumeració entri dins de l'espai que li toca. Quan això passa l'enumeració passa a la següent línia com es pot veure al segon cas de l'Exemple 3.7. Per a solucionar aquest (possible) inconvenient podem fer servir `split` dins de l'entorn `align`.

## Capítol 4

# Figures i taules

Un document qualsevol és molt probable que necessiti d'elements gràfics o visuals que ajudin a comprendre la informació que s'està exponant. En aquest capítol us donarem algunes eines bàsiques que haurien de ser suficients per a poder col·locar imatges i taules, la seva configuració i alguns tips per a poder obtenir un document endreçat.<sup>1</sup>

Tant les figures com les taules es treballen amb entorns i comandaments que generen objectes o entorns de tipus flotant. Ja ho hem comentat en capítols anteriors, aquests objectes flotants són elements que LaTeX entén que s'han de col·locar dins del document a la posició més òptima possible. El compilador de LaTeX obté informació de la configuració de l'objecte flotant i el col·loca segons la jerarquia que vulguem. S'ha de tenir en compte que el lloc on posicionem l'objecte flotant ha de ser compatible amb objectes flotants. Si no treballem en entorns que canviïn molt el posicionament dels paràgrafs en text pla o altres objectes comuns com els entorns matemàtics podem fer servir objectes flotants sense cap problema. En aquest capítol veurem un entorn incompatible amb objectes flotants i com solucionar-ho.

Per a poder entendre en més profunditat com funcionen els entorns **figure** i **table** primer hem de fer un cop d'ull a l'entorn **minipage** i a l'estructura d'un document de LaTeX per defecte. Podeu aprofundir sobre aquest entorn i l'estructura d'un document a la documentació de la bibliografia [1][2].

### 4.1 Format de pàgina

Cada pàgina d'un document de LaTeX se separa en tres parts: capçalera, cos i peu. La configuració de la mida de les separacions d'aquestes tres parts de cada pàgina modifica el posicionament dels objectes que cridem amb el codi. Aquesta configuració la podem modificar al preàmbul amb el comandament de configuració `\setlength{...}{...}`. Veiem com aquest comandament té dos espais on col·locar arguments. El primer espai serveix per a indicar quina és la distància concreta que volem modificar i el segon per a indicar la modificació que volem aplicar a aquesta distància. Per a aclarir aquesta descripció podem redactar un exemple:

El comandament `\textwidth` és aquell comandament que conté la informació de la mida de l'amplada del text al document que estem escrivint. Notem que hem dit l'amplada del text al document, és a dir, entre els marges que haguem configurat al preàmbul. Existeix un altre comandament similar (que se sol fer servir de forma indiferent però sense saber exactament les diferències que tenen) anomenat `\linewidth`. Aquest segon comandament conté la informació de la mida de l'amplada de text a l'entorn on estem redactant. Veiem doncs que, mentre que `linewidth` és dinàmic, `textwidth` és fixe<sup>2</sup>. En cas que vulguem modificar la mida per defecte del `textwidth` podem aplicar el comandament

$$\setlength{\textwidth}{mida + unitats}$$

---

<sup>1</sup>Aquestes eines funcionen sempre en tipus de document com **article** o **book**. Però, en altres tipus de document, amb documents amb una configuració específica o al carregar un paquet que canviï molt LaTeX bàsic és possible que no funcionin. En aquests casos s'ha de veure cada cas de forma individual, intentant comprendre com funcionen el tipus de document, la configuració imposada o el paquet instal·lat.

<sup>2</sup>Això es pot notar molt en el cas de escriure en més d'una columna, mentre que `textwidth` es manté constant `linewidth` es modifica adaptant-se a la mida de l'amplada de la columna on estem treballant.

on al darrer comandament podem col·locar la mida que vulguem amb les seves corresponents unitats (com si empleguem un `hspace` o `vspace`).

També podem aplicar la mida d'una distància a una altra amb el mateix comandament. Si per exemple estem treballant a doble columna i volem (per algun motiu excepcional) que `linewidth` tingui la mateixa mida que `textwidth` podem escriure

```
\setlength{\linewidth}{\textwidth}
```

En aquest cas no cal que posem unitats ja que no estem especificant una distància amb números. Us deixem una breu llista amb algunes de les distàncies que es poden modificar per a adaptar el format de pàgina del nostre document:

- `textwidth` - Amplada completa del text dins del cos
- `textheight` - Altura completa del text dins del cos
- `headheight` - Altura de la capçalera del document
- `topmargin` - Marge superior fins a la frontera superior de la capçalera
- `footheight` - Altura del peu de pàgina
- `footskip` - Distància des de la base del cos fins a la frontera inferior del peu de pàgina

També us deixem un recurs visual per a poder observar aquestes distàncies per al format de pàgina d'aquestes notes, Figura 4.1, i per a un document en blanc de LaTeX, Figura 4.2.<sup>3</sup>

## 4.2 L'entorn minipage

L'entorn `minipage` genera capses independents de la resta del text on es pot escriure com si fos un paràgraf normal. Al seu interior podem escriure text pla o incloure imatges. L'entorn `minipage` no és un entorn flotant, genera les capses allà con el col·loquem al codi. Posem un exemple senzill per a comprendre a què ens referim quan diem que genera capses:

### Exemple 4.1

```
1 Aquest es un text d'exemple fora del minipage,
2 \begin{minipage}{3cm}
3     un cop el text que escrivim es troba dins del minipage es co\lgem
4     oca dins del document en forma de caps a amb una amplada de 3
5     centimetres.
6 \end{minipage}
7 Fora del minipage podem continuar escrivint de forma normal, respectant
8 el salt de linia for\c{c}at per l'al\c{c}ada que s'ha generat pel \
9 texttt{minipage}.
```

Sortida:

<p>Aquest es un text d'exemple fora del minipage,</p> <p>continuar escrivint de forma normal, respectant el salt de linia forçat per l'alçada que s'ha generat pel minipage.</p>	<p>un cop el text que escrivim es troba dins del minipage es col·loca dins del document en forma de caps a amb una amplada de 3 centimetres.</p> <p>Fora del minipage podem</p>
--	---

Veiem com a l'Exemple 4.1 en mig de la línia es genera un requadre que correspon a l'entorn `minipage`. Aquest requadre té l'amplada que indiquem a l'argument del `minipage`. Aquest argument és obligatori

<sup>3</sup>Aquests esquemes mostren més distàncies de les que hem posat al llistat. Podeu obtenir-los al vostre document amb el paquet `layout` i col·locant `\layout` allà on vulgueu que s'imprimeixi la imatge.

per a poder fer servir-ho, si no es col·loca LaTeX ho ignora i pot donar a error. Dins de l'espai generat pel `minipage` el text es comporta com a text pla només que es troba restringit al nou format imposat pel `minipage`. De fet, el nom del comandament mateix ho indica, el que fa és com si es generés una mini pàgina amb un format en concret. Dins d'aquest espai generat podem editar pràcticament com si fos un altre document dins del nostre document. Els entorns `minipage` es poden anidar, iguala quan feiem `itemize` anidats, un dins de l'altre. Només cal ser coherent amb l'espai disponible al nostre document abans i després de col·locar els `minipage`.

Podem notar com per defecte la capsa on s'imprimeix el text que va al `minipage` queda aliniada verticalment amb la resta del text pel centre. Podem controlar aquesta aliniació aplicant un paràmetre de configuració secundària que, com hem pogut observar anteriorment, s'aplica entre claudators. Les possibles configuracions són tres:

- c - Per a aliniació pel centre
- b - Per a aliniació per la vorera inferior de la capsa
- t - Per a aliniació per la vorera superior de la capsa

Observem millor aquesta aliniació amb un exemple.

#### Exemple 4.2

```

1  Aquesta linia mostra un exemple de
2  \begin{minipage}[b]{1.5cm}
3      paragraf aliniat per baix
4  \end{minipage},
5  \begin{minipage}[c]{1.5cm}
6      paragraf aliniat pel centre
7  \end{minipage} i
8  \begin{minipage}[t]{1.5cm}
9      paragraf aliniat per dalt
10 \end{minipage}
```

Sortida:

	paragraf aliniat	paragraf	
Aquesta linia mostra un exemple de	per baix	, aliniat	i paragraf
	pel centre	aliniat	
		per dalt	

Sabent aquest funcionament del `minipage` i tenint cura de quant espai ocupa el nostre `textwidth`<sup>4</sup> podem recrear al nostre document el funcionament de la doble columna. Aquesta forma d'aplicar-ho és interessant en alguns formats comuns de presentació d'imatges en un text qualsevol. Ara posarem un exemple d'això, però abans expliquem breument com podem afegir imatges fent servir LaTeX.

#### 4.2.1 Incloure imatges al document

Ho hem comentat a l'inici d'aquestes notes i ho tornem a remarcar en aquesta subsecció: podem treballar amb LaTeX utilitzant diferents entorns de treball. Podriem dir que només cal un editor de textos per a poder començar a escriure en LaTeX i que la resta de la feina seria poder passar el nostre arxiu en format `.tex` pel compilador per a poder obtenir un arxiu pdf. Pensar que per a treballar amb LaTeX només ens calen aquestes dues coses és una limitació, disposem de moltes eines fora d'un arxiu de text!

El que intentem motivar en aquesta introducció és a ser conscient que els arxius de LaTeX existeixen dins d'un directori on hi han altres arxius, de qualsevol tipus, i a més que podem relacionar-los i fer-los servir com a eines per a millorar els nostres documents. Si es treballa en local potser aquesta

<sup>4</sup>El motiu d'aquesta precaució és una possible i no desitjada recol·locació dels objectes a la pàgina que estem editant. Depenent de que s'intenti imprimir al document si dos objectes ocupen completament la mida del `textwidth` LaTeX automàticament fa un salt de línia per a optimitzar l'espai. Per a evitar això, com veurem en algun exemple, podem forçar a que els objectes en la seva totalitat ocupin una mica menys que el `textwidth` del que disposen.

motivació no cal ja que és un fet força obvi. Però, si treballes en local al teu dia a dia vol dir que has hagut de moure't ja una mica entre directoris i, per tant, la teva experiència fa que les coses es vegin òbvies. Aquestes notes són introductories, per tant intentarem seguir un aprenentatge natural envers aquests conceptes, començant per indicar que per a algú que s'està iniciant no veu òbvies aquestes coses. Fem primer un esboç general i després ho traslladarem a Overleaf, que és el nostre entorn de treball. En aquesta part només explicarem lo necessari per a poder utilitzar imatges, però si es vol aprofundir una mica més podeu anar a la Secció 6.1.

### Funcionament dels directoris

Els arxius amb els que treballem o interactuem dins d'un ordinador s'ordenen en directoris o carpetes. En els sistemes operatius d'ús comú és òbvia aquesta forma d'ordenar la informació la nostre dispositiu, es pot observar fins i tot de forma totalment visual amb, per exemple, l'explorador d'arxius de Windows. Navegar entre els arxius és molt intuïtiu, només cal entrar a la carpeta on volem veure els arxius que ens interessa. Aquesta acció d'entrar a un directori ve determinada per la seva ruta d'accés, i la ruta d'accés no és més que els noms dels directoris que s'han anat seleccionant, de més general a més concret llegint la ruta d'esquerra a dreta. És notable que la forma en que els directoris ordenen la informació és de forma anidada, existeix un directori que conté la resta de directoris que podem anomenar unitat principal.

Un dels conceptes clau per a entendre el funcionament dels directoris és saber com es delimita l'accés d'un arxiu a la resta de directoris que existeixen. Si no s'indica cap ruta d'un directori en concret, la informació a la que pot accedir un arxiu situat a un directori és només la que se situa al mateix directori, al mateix nivell. És a dir, si el nostre arxiu es troba a la carpeta *A* i aquesta conté, a part del nostre arxiu *X*, dues carpetes més anomenades *B* i *C* i un arxiu de text *Y*, llavors només podrem tenir accés a la informació de l'arxiu de text *Y* que ens acompanya. La informació de les carpetes *B*, *C* i aquelles que es trobin abans de la *A* no seràn accessibles si no s'especifica la ruta d'accés.

Per a poder accedir a informació a partir de les rutes d'accés primer hem de començar per un punt de referència, és a dir, seleccionar el directori des d'on aplicarem les rutes per a cercar la informació que volem<sup>5</sup>. El punt de referència pot canviar segons l'arxiu des del qual busquem accedir a altres directoris, però normalment per defecte és el mateix directori a on es troba el nostre arxiu. Per a poder utilitzar la navegació per rutes hem de fer servir comandaments que ens permetin pujar i baixar de nivell dins dels directoris (de no fer-ho no hi hauria manera d'accedir a la informació que volem). En terminals tant de Windows com Mac o Linux el comandament que permet pujar de nivell s'escriu amb dos punts. Veiem les diverses situacions comunes i com s'accedeix a la informació en cada una d'elles. Als següents exemples situem la referència a on es troba el nostre arxiu *X* (marcat amb un triangle negre) i volem fer servir la informació que es troba a l'arxiu *Y*. La sintaxi de les rutes és la que es fa servir Linux, però, com hem dit, és fa d'una forma molt similar amb Windows o Mac.

#### Exemple 4.3

##### VISUALITZACIÓ DELS DIRECTORIS

Primera situació, punt de referència situat al mateix nivell en el mateix directori:

```
► Directori_A
    Arxiu_X.tex
    Arxiu_Y.tex
  ► Directori_B
  ► Directori_C
```

<sup>5</sup>A partir d'aquest moment hauriem de començar a parlar de rutes relatives d'accés, però les seguirem dient simplement rutes d'accés. Aquesta diferència prové de que amb una referència ja no és necessari accedir al nostre directori desde la unitat més general del nostre ordinador, només cal seleccionar la posició relativa dels nostres directoris al nostre dispositiu i la resta de la informació vindrà donada mitjançant el punt de referència. Això és útil per a compartir projectes, ja que tothom té un ordre i noms únics dels seus propis directoris. Amb aquesta eina, si algú descarrega el directori més general d'un programa o conjunt d'arxius, ja pot navegar per tot el codi sense haver d'adaptar totes les rutes d'accés a com té els directoris al seu ordinador.

```

1 # Ruta d'accés
2 Arxiu_Y.tex

```

Aquesta situació és la més senzilla, passa quan els dos arxius es troben al mateix directori. Fariem servir la mateixa ruta d'accés si els dos arxius estiguessin a *B* o a *C*. Notem que és necessari especificar el tipus d'arxiu i no només el nom, en aquest cas és un arxiu de codi de LaTeX, format `.tex`.

#### Exemple 4.4

##### VISUALITZACIÓ DELS DIRECTORIS

Segona situació, punt de referència situat a un nivell superior:

```

► Directori_A
    Arxiu_X.tex
  ▷ Directori_B
    Arxiu_Y.tex
  ▷ Directori_C

```

```

1 # Ruta d'accés
2 Directori_B/Arxiu_Y.tex

```

En aquest segon cas de l'Exemple 4.4, com la informació es troba en un nivell inferior a la referència podem accedir de forma senzilla escrivint la ruta només amb el nom del directori i l'arxiu. És anàleg a entrar a la carpeta on es troba l'arxiu que ens interessa.

#### Exemple 4.5

##### VISUALITZACIÓ DELS DIRECTORIS

Tercera situació, punt de referència situat a un nivell inferior:

```

▷ Directori_A
    Arxiu_Y.tex
  ► Directori_B
    Arxiu_X.tex
  ▷ Directori_C

```

```

1 # Ruta d'accés
2 ../Arxiu_Y.tex

```

La ruta d'aquest tercer cas comença amb dos punts. Aquest comandament serveix per a pujar de nivell dins dels directoris, si pensem que és com moure un cursor, el comandament `../` ens fa canviar el cursor de la carpeta *B* a la carpeta *A*. Con ja ens trobem al directori on es troba l'arxiu que ens interessa només cal continuar posant el nou de l'arxiu *Y*. Continuant amb l'analogia anterior, podem pensar que el que fem es anar cap a enrera entre els directoris i després accedir a l'arxiu que ens interessa. Veiem finalment un darrer cas que és una mescla dels Exemples 4.4 i 4.5.

#### Exemple 4.6

##### VISUALITZACIÓ DELS DIRECTORIS

Quarta situació, punt de referència situat al mateix nivell però en directoris diferents:

```

▷ Directori_A
    ▷ Directori_B
      Arxiu_X.tex
  ► Directori_C
    Arxiu_X.tex

```

```

1 # Ruta d'accés
2 ../Directori_B/Arxiu_Y.tex


```

Observem que hem hagut de primer pujar un nivell (directori *A*) i posteriorment tornar a baixar-lo pero entrant en el directori corresponent. Podem pensar que l'acció que fem és anar cap a enrere entre els directoris i després accedir a la carpeta on es troba l'arxiu que ens interessa.

Coneixent el funcionament bàsic dels directoris podem tractar sense complicacions la introducció d'imatges dins de LaTeX escrivint en un entorn de treball qualsevol. Observem el cas concret d'Overleaf. Com ja sabem Overleaf és un editor de textos que funciona en línia. Cada cop que obrim un nou arxiu es genera un directori principal amb un arxiu `.tex` omplert amb la informació predeterminada d'un document nou per defecte. La navegació entre els directoris a Overleaf succeeix a la part esquerra de l'overlay, a l'explorador d'arxius. Com usualment treballem en l'arxiu `.tex` principal (anomenat `main` per defecte) del document, sempre tindrem que el nostre punt de referència serà la carpeta més general del nostre nou document.

En Overleaf podem pujar a l'explorador els arxius que vulguem i ordenar-los en carpetes. El funcionament dels directoris en aquest explorador és igual a com hem explicat més a dalt.

### Selecció d'imatges per a un document

Usualment per a treballar de forma ordenada el que es fa en un document d'Overleaf (o en el directori principal del nostre projecte de LaTeX en local) és mantenir el nostre arxiu `.tex` dins del directori principal i en una carpeta a part les imatges que volem fer servir al document. Pujar les imatges a l'explorador és fàcil, només cal prémer  i buscar o arrastrar els arxius que vulguem pujar.

Un cop pujem els arxius que vulguem necessitem cridar una funció que els imprimeixi al document. Per a aquesta tasca farem servir el paquet `graphicx`. En Overleaf aquest paquet es carrega al preàmbul de forma automàtica. Principalment `graphicx` permet imprimir i configurar imatges allà on les col·loquem al text fent servir el comandament `includegraphics`. Posem un exemple.

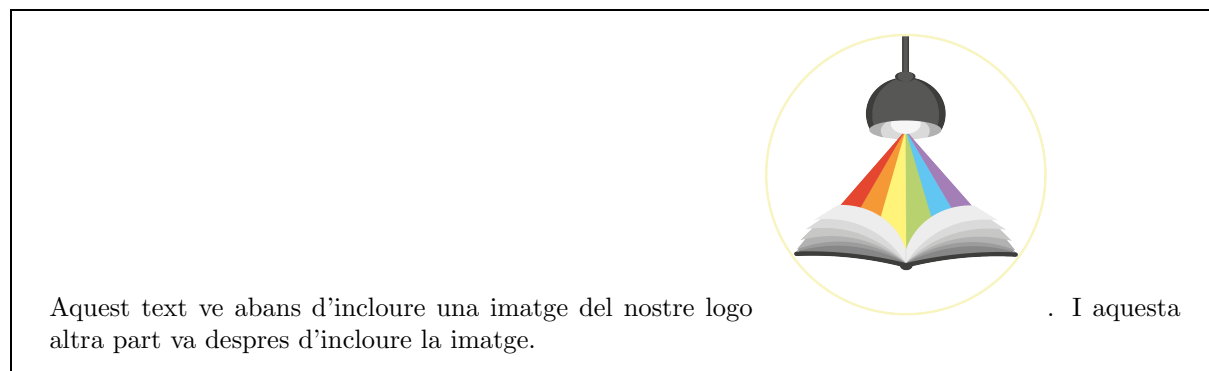
#### Exemple 4.7

```

1 Aquest text ve abans d'incloure una imatge del nostre logo \
  includegraphics[width=0.25\linewidth]{Imatges/logo_opticat.png}. I
  aquesta altra part va despres d'incloure la imatge.

```

Sortida:



Com veiem a l'Exemple 4.7 la imatge es col·loca en una capsula de forma similar a com es col·loca un `minipage` alineat per la part inferior. L'objecte que inclou la imatge no és un objecte flotant, apraixerà justament allà on l'haguem cridat al codi i amb la mida que haguem escollit. Com en altres casos, la configuració adicional del comandament `includegraphics` permet modificar més d'un paràmetre, per tant cal especificar quin és el paràmetre que configurem, col·locar un igual (el símbol “=”) i la configuració del paràmetre. En el cas de l'Exemple 4.7 el que es modifica és l'amplada de la imatge. Aquesta mida passa de la mida original (que és més ample que el document on escrivim les notes) a una mida equivalent al 25% del l'amplada de línia del colorbox.

Us deixem un llistat amb els paràmetres que podeu modificar d'una imatge amb `includegraphics`:



- Modificar mida i escala:
  - **width** - Amplada de la imatge
  - **height** - Altura de la imatge<sup>6</sup>
  - **scale** - Escala respectant la relació d'aspecte original de la imatge.
- Rotar imatge:
  - **angle** - Les unitats són els graus i només cal afegir el número. El sentit de gir és antihorari.
- Retallar la imatge:
  - **trim + clip** - El paràmetre **trim** emmagatzema les distàncies que es volen retallar i **clip** fa l'acció de retallar la imatge. L'argument de **trim** ha de contenir quatre números separats. Aquests números marquen la distància que es retalla (mesurada en punts, **pt**) de cada banda de la imatge, respectant l'ordre **left**, **bottom**, **right** i **top**. Posem un exemple de com cridar una imatge retallada:

```
\includegraphics[trim= 10 20 10 20, clip, width=\linewidth]{ruta/document.tex}
```

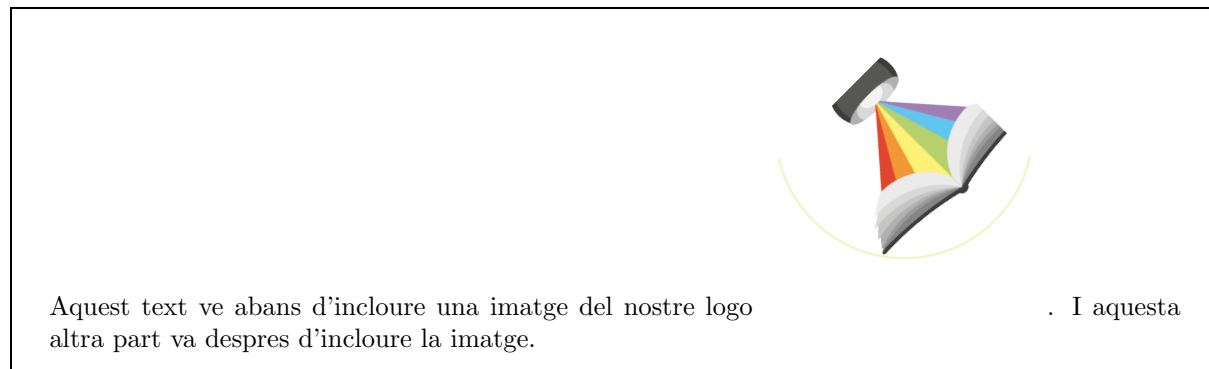
El que ens mostraria aquest comandament és una imatge que ocupa l'espai de línia disponible i que, respecte a la original que es troba als directoris, aquesta es veu retallada horitzontal i verticalment.

Podem posar també un exemple similar a l'Exemple 4.7 però amb els paràmetres modificats:

#### Exemple 4.8

```
1 Aquest text ve abans d'incloure una imatge del nostre logo \
  includegraphics[angle=45,trim= 0 0 200 200, clip, width=0.25\linewidth]{
  Imatges/logo_opticat.png}. I aquesta altra part va després d'incloure la
  imatge.
```

Sortida:



Veiem a l'Exemple 4.8 com hem rotat la imatge del nostre logo amb un angle de  $45^\circ$  i la hem retallat per la part superior i la part dreta uns 200 punts. L'aliniació de la imatge amb el text respecta que la col·locació sigui respecte al punt més baix de la capsa que conté la imatge. Al estar la imatge rotada podem veure com sembla que la imatge hagi pujat, però això es deu a que es troba aliniada amb el text per l'esquina inferior de la capsa. Com a darrer comentari d'aquest exemple veiem que el comandament **trim** retalla respecte els marges de la imatge original. rotar la imatge no crea una nova capsa amb la imatge inclinada, sinó que inclina tota la capsa i la resta de configuracions es fan respecte a l'estat de la capsa actual.

Finalment podem mostrar l'exemple que hem comentat al final de la Secció 4.2. Hi ha llibres on s'acostuma a posar una figura amb un peu de figura i una breu descripció o comentaris a un lateral. Aquest format el podem aplicar de la següent manera: **Exemple 4.9**

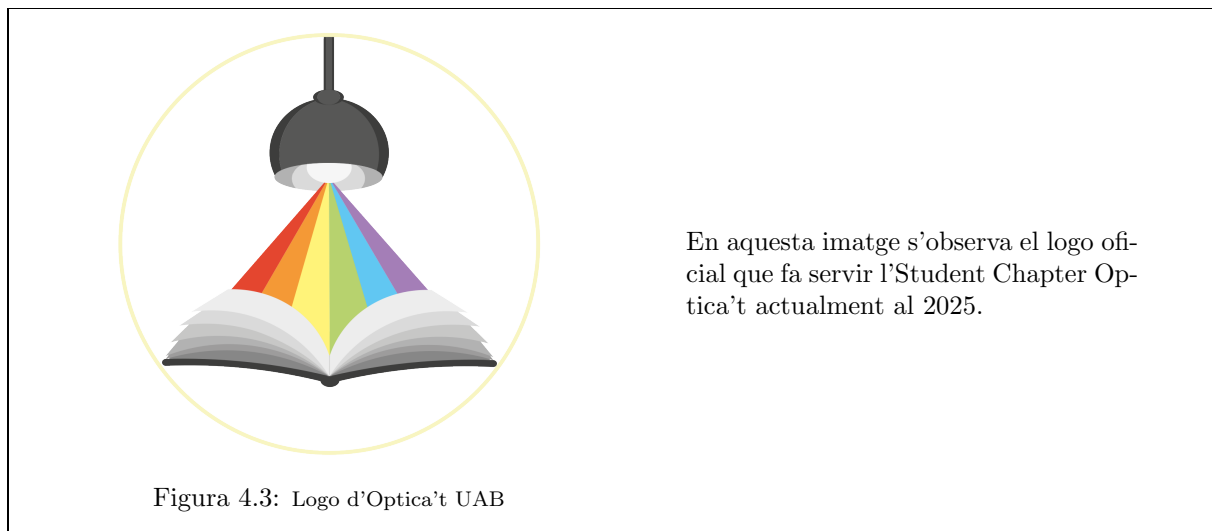
<sup>6</sup>Aquestes dues opcions permeten modificar la relació d'aspecte original de la imatge. Si només apliquem un d'ells normalment funciona com un reescalament. En cas que es modifiqui la relació d'aspecte, podem posar seguidament d'una coma **keepaspectratio** per a forçar-la a mantenir-se com la original.

```

1 \begin{minipage}{0.5\linewidth}
2   \centering
3   \includegraphics[width=0.75\linewidth]{Imatges/logo_opticat.png}
4   \captionof{figure}{\footnotesize Logo d'Optica't UAB}
5 \end{minipage}
6 \hspace{0.05\linewidth}
7 \begin{minipage}{0.4\linewidth}
8   En aquesta imatge s'observa el logo oficial que fa servir l'Student
9   Chapter Optica't actualment al 2025.
10 \end{minipage}

```

Sortida:



L'Exemple 4.9 ens mostra un format útil de presentació d'imatges dins dels nostres documents. És una forma de simular el format de doble columna sense entrar en incompatibilitats innecessàries. Com es pot observar per a afegir un peu de figura fem servir el comandament `\captionof`. Aquest comandament l'expliquem en la següent secció.

### 4.3 L'entorn figure

Ja hem pogut observar a l'inici d'aquestes notes i a la secció anterior com LaTeX permet afegir imatges sense cap problema. Tot i que podem afegir allà on vulguem el comandament `\includegraphics` és interessant conèixer l'entorn `figure`. Aquest entorn, com ja hem comentat abans, és un entorn flotant que acabarà optimitzant l'espai dins del document per a que hi càpiga la imatge que afegim al seu interior. A part d'això, l'entorn `figure` permet indexar com a objectes independents a les figures i permet afegir títols a les imatges. La forma més quotidiana de fer servir un entorn `figure` és la següent:

#### Exemple 4.10

```

1 \begin{figure}[h]
2   \centering
3   \includegraphics{Imatges/logo_opticat.png}
4   \caption{Aixo es un titol de figura (o caption) d'exemple}
5 \end{figure}

```

Sortida:



Figura 4.4: Això és un títol de figura (o caption) d'exemple

A l'exemple 4.10 podem veure com afegim tres comandaments dins de l'entorn **figure**. El primer s'encarrega de mantenir al centre els objectes que afegim a dins de l'entorn. El segon crida a la imatge que tenim al directori a partir de la seva ruta d'accés. El tercer genera i col·loca un nou objecte anomenat **caption**. Aquest objecte identifica que es troba dins d'un entorn **figure** i explicita la indexació i possible etiquetatge de la figura al document. Podem fer servir referències creuades afegint un **label** a dins o després de posar el **caption**. La generació de figures permet crear un llistat que LaTeX emmagatzema com a informació adicional. Podem crear un índex de figures aplicant el comandament **listoffigures**, que funciona igual que cridar l'índex de continguts.

## 4.4 Com col·locar subfigures

## 4.5 L'entorn table

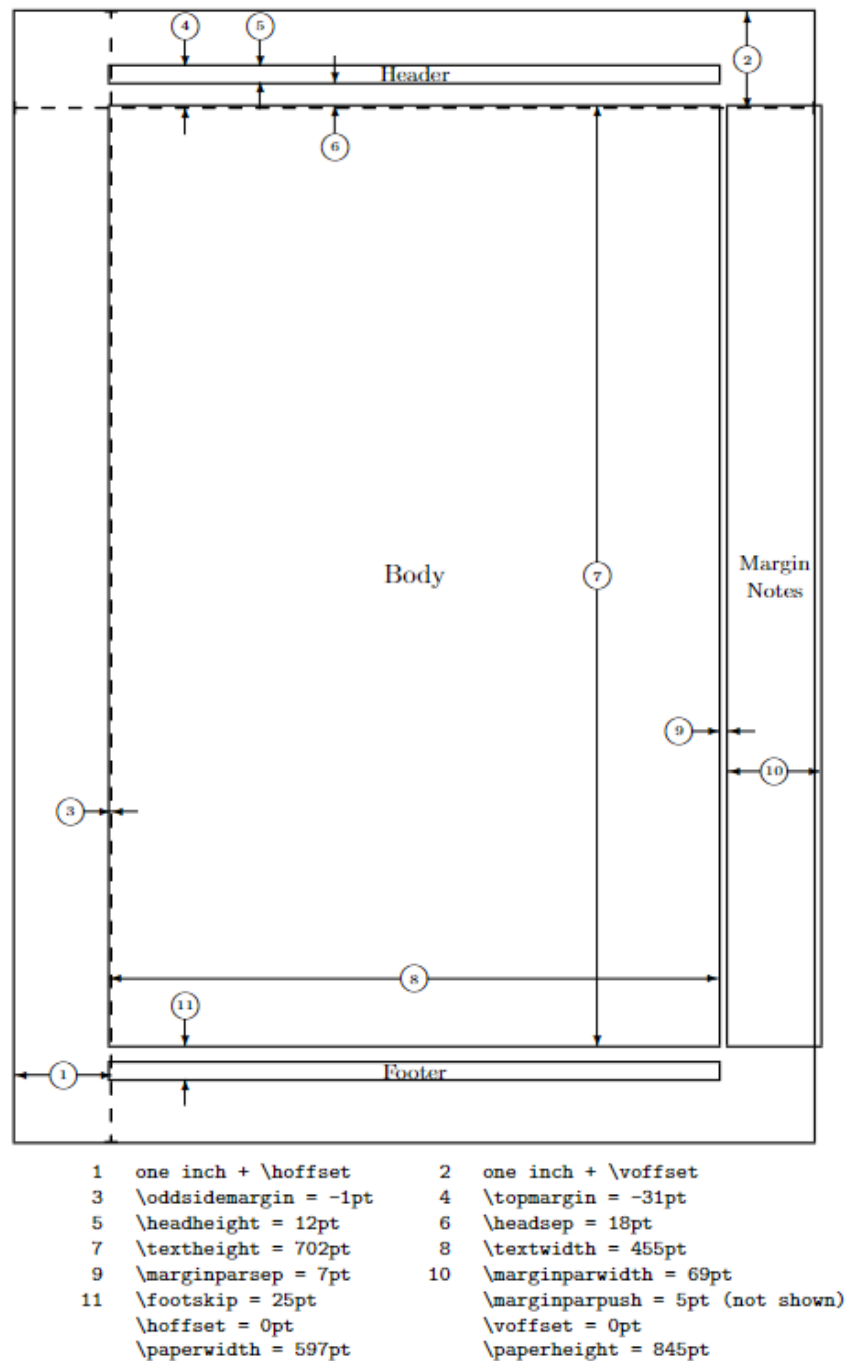


Figura 4.1: Esquema de distàncies que determinen el format de pàgina d'un document de LaTeX on s'han forçat els marges amb el paquet `geometry`.

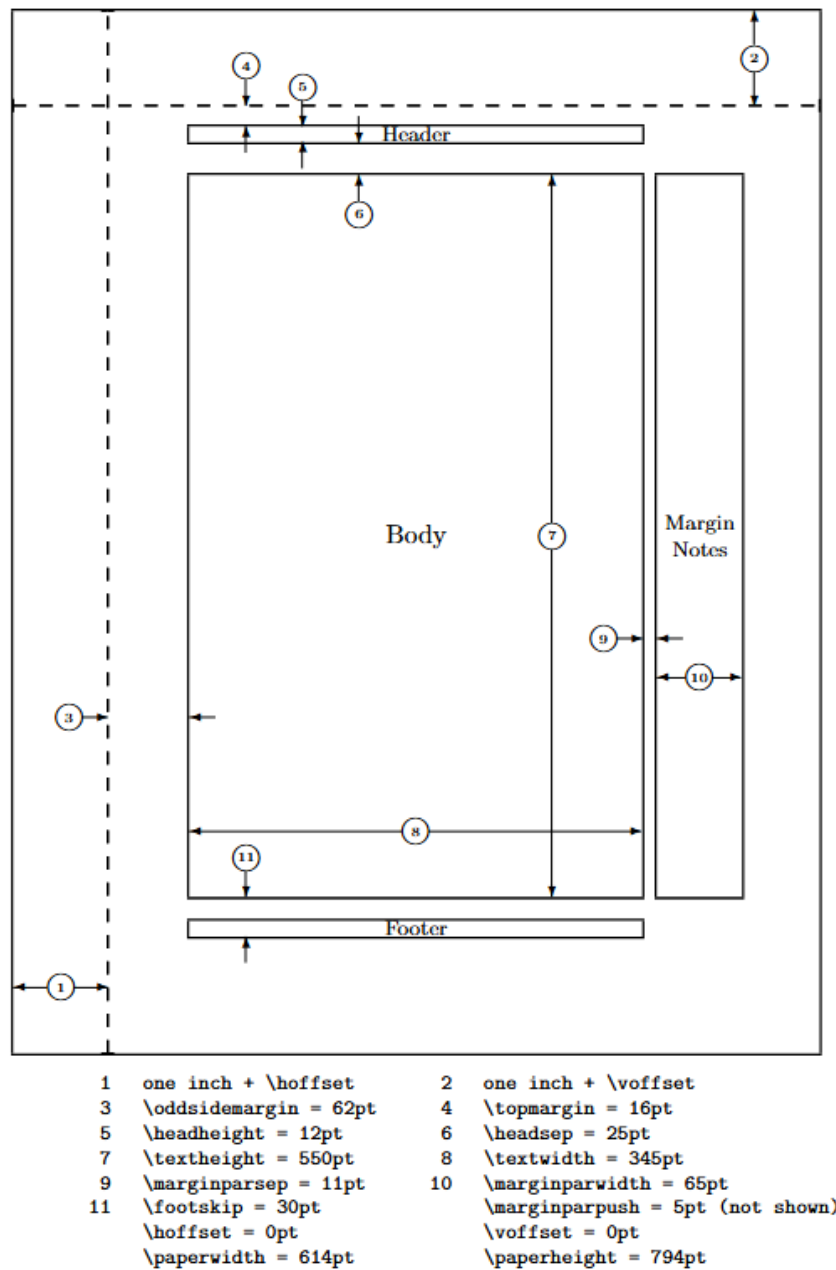


Figura 4.2: Esquema de distàncies que determinen el format de pàgina d'un document de LaTeX per defecte.

## Capítol 5

# La bibliografia

### 5.1 L'arxiu .bib

### 5.2 Referenciar i imprimir la bibliografia

## Capítol 6

# Personalització avançada

### 6.1 Ús d'arxius externs

#### 6.1.1 Arxiu externs com a preàmbul (comandament input)

### 6.2 Paquets útils

#### 6.2.1 colorbox

#### 6.2.2 babel

#### 6.2.3 fancyhdr

#### 6.2.4 hyperref

#### 6.2.5 wrapfig

#### 6.2.6 mhchem

# Bibliografia

- [1] Helmut Kopka and Patrick W. Daly. *A Guide to LaTeX. Document Preparation for Beginners and Advanced Users*. Addison-Wesley, 1993.
- [2] Latex-Tutorial. The minipage environment, 2021. Disponibile a: <https://latex-tutorial.com/minipage-latex/>.