

# Introducción a la escritura científica en $\text{\LaTeX}$

Fernando Oleo Blanco

Universidad ICAI Comillas  
Asociación de LinuxEC

*201507027@alu.comillas.edu*

11 de agosto de 2019

- 1 Recursos
- 2 Razones
- 3 Introducción general
- 4 Introducción cualitativa
- 5 Un poco de simbología y ejemplos
- 6 Tablas
  - Matrices
- 7 Ecuaciones
- 8 Recursos expresivos
- 9 Intercompatibilidad
  - Microsoft Word
  - Matlab
- 10 Fin

# Recursos recomendados

## Lectura

- *The not so Short Introduction to  $\LaTeX$*  por Tobias Oetiker.
- *$\LaTeX$  Wikibook*: Libro escrito por y para Wikipedia.
- *More Math Into  $\LaTeX$*  por George Grätznér (esta es una buena muestra).

## Internet

- **Editor colaborativo** herramienta moderna para el trabajo en grupo, multiedición y tabajo a tiempo real en la nube.
- **Overleaf**: escritura de  $\LaTeX$  en el navegador  $\leftarrow$  ya os estáis metiendo.
- Cualquier servicio con plantillas (**Latextemplates** por ejemplo).
- **Tug**: Centro de recursos *oficiales*.
- Foros, "puntos de información", etc.
- Google.

# ¿Por qué?

## Rasgos generales

- 1 Herramienta estándar de escritura científica. Prácticamente todos los servicios científicos tienen alguna herramienta de exportación.
- 2 Gran flexibilidad. Fácil uso (más indicaciones a continuación).
- 3 Muy buena integración con herramientas científicas: Matlab, Wolfram, R... y MS Word.
- 4 Saber más nunca fue malo bueno...

## Características propias

- 1 Fácil lectura.
- 2 Estructuración y formalidad absoluta.
- 3 Eficiente en la escritura.
- 4 Perfectamente integrado en el texto.
- 5 Requiere de un largo aprendizaje, pero prisa no hay.

# Otra introducción a $\text{\LaTeX}$

Ya que muchos no sabéis absolutamente nada de  $\text{\LaTeX}$ ... Vuelta a la introducción.

## TexStudio

- **TeXStudio:** Download  $\rightarrow$  busca tu plataforma. Instálalo como solo tú sabes.

## TexLive

- **Tug**

Windows: pincháis en download e instaláis, instaláis **todo** 4.5Gb

Mac: Link MacTeX distribution. Instaláis y listo.

Linux y BSD: ya sabéis.

# Consejos

Procedimiento que **siempre, siempre, siempre** seguiréis:

- Nunca empecéis desde un documento en blanco, usad plantillas (templates).
- Crearos vuestras propias plantillas.
- Usad comentarios, especialmente si acabáis de empezar  
`% comentario`.
- No compliques demasiado las cosas, como en programación, con for se hace mucho, solo hay que jugar con él.
- Google.
- Acordaros que funciona como un entorno de programación.

# Estructura básica de un comando

## Comando normal

Ejemplo: `\documentclass[12pt, landscape, a4paper]{article}`

Las partes son:

- El propio comando `\documentclass`
- Las opciones de uso de ese comando `[12pt, landscape, a4paper]`.
- Y los datos de ese comando (argumento) `{article}`
- De lo anterior puede que sean opcionales u obligatorias las opciones y/o el argumento.

## Comando de entorno

E.j: `\begin{columns}[T] ... \end{columns}` y

`\begin{block}{Argumento del entorno} ... \end{block}`

El mismo cuento. Pero hay que añadir el detalle de que el entorno se pone entre llaves después de `\begin`.

# Documento básico en L<sup>A</sup>T<sub>E</sub>X

<code>\documentclass[] {article}</code>	Ha de estar siempre, define nuestro trabajo
<code>\title{}</code>	Nuestra información personal
<code>\author{}</code>	
<code>\begin{document}</code>	Aquí comienza nuestro documento
<code>\maketitle</code>	Nos hace nuestra portada automáticamente
<code>\section{}</code>	Una sección (parte principal del texto)
<code>\end{document}</code>	Finaliza nuestro trabajo



Los de habla española tenemos que configurar nuestro documento un poco. Aunque  $\text{\TeX}$  se diseñase hasta para aceptar chino, no lo usa por defecto. Además, tiene sus ventajas. A continuación de `\documentclass[]\{article}` vamos a poner las siguientes líneas:

- `\usepackage[utf8]{inputenc}`
- `\usepackage[spanish]{babel}`
- `\usepackage{graphicx}`
- `\usepackage{amsmath, amssymb}`
- `\usepackage{hyperref}`
- `\usepackage{geometry}`

Los dos primeros son para que nos deje poner la Ñ y parta las palabras con guión de manera correcta (99 % de los casos) de manera automática. ¿A que mola? `Graphicx` para poner fotos; `Amsmath` para símbolos matemáticos y mucho más. `Geometry` para cambiar las dimensiones de los márgenes. `Hyperref` para hacer referencias externas e internas.

# Comandos de estructura

- `\include{file}` Incluir otro texto escrito en `.tex`
- `\tableofcontents` Genera índices de manera automática con los comandos que vienen a continuación.
- `\chapter{title}` Solo disponible en book.
- `\section{title}` Parecido a `\chapter` pero utilizable en cualquier entorno (En Beamer funciona distinto).
- `\subsection{title}` y `\subsubsection{title}`.
- `\paragraph{text}` Párrafos especiales (no aparecen en el índice).

El uso y configuración de `\makeindex` queda fuera de esta charla

**Nota:** en este recuadro no he puesto título, fíjate en la diferencia.

# Márgenes

Los márgenes en  $\text{\LaTeX} 2_{\epsilon}$  son diseñados para escritura profesional, no son sencillos de manejar. Tendremos que usar las opciones del paquete `Geometry`.

Como se indico anteriormente usaremos el comando `\geometry{options}` con sus argumentos para definir los márgenes queridos. Unos márgenes sanos y de fácil modificación son los siguiente.

```
\geometry{ a4paper, total={170mm,257mm}, left=20mm,
top=20mm,}
```

Más información en [Share \$\text{\LaTeX}\$](#)

## Items

```
\begin{itemize}
\item[label] description
\end{itemize}
```

## Enumerate

```
\begin{enumerate}
\item[label] description
\end{enumerate}
```

## Notas a pie de página

Esto es un poco de texto.<sup>a</sup> Y este mejor<sup>1</sup>.

Esto es un poco de texto.\footnote{Y esto la anotación}  
Y este mejor\footnote[frame]{¿A que sí?}

---

<sup>a</sup>Y esto la anotación

---

<sup>1</sup>¿A que sí?

# Donald Ervin Knuth

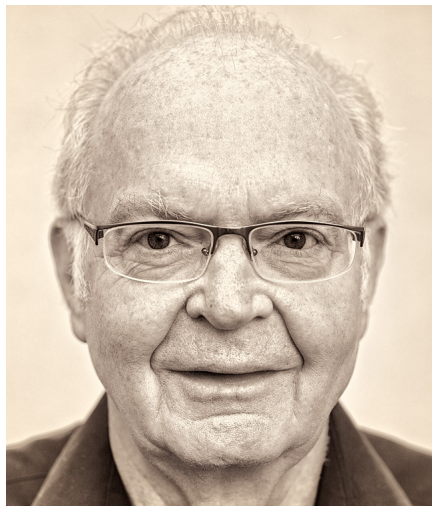


Figura: Donald Ervin Knuth. Creador de  $\text{\TeX}$

# Desarrollo y lógica del comando

Pudiendo daros un montón de datos no sería útil, así que continuemos con las herramientas.

## Creación y diseño

$\text{\LaTeX}$  fue creado por un matemático bien pragmático, que también es considerado uno de los padres de la informática moderna, siendo su especialidad el **diseño** de algoritmos.  $\text{\LaTeX}$  se creó para ser una herramienta eficiente y flexible que solucionara los problemas presentes de la tipografía y de la escritura científica; además de ser estable (seguimos con la misma versión desde 1994 de  $\text{\TeX}$ ).

## Mnemotecnia

Una herramienta que requiera más tiempo para poder ser utilizada que para trabajar no tiene ningún sentido,  $\text{\LaTeX}$  soluciona tal problema (sorprendente). Para facilitar su uso, su diseño es puramente mnemotécnico y gráfico, a la vez que expresivo. Aprender su diseño es cuestión de hora y media.

# Comenzamos

Notas importantes en la lógica de la escritura científica en  $\text{\LaTeX}$ :

## Desarrollo

- $\text{\LaTeX}$  se creó para permitir una fácil y rápida creación de textos, aunque parezca poco intuitivo al principio.

**Regla de la mano derecha:** si algo es muy utilizado y básico en el mundo de las matemáticas y de las ciencias, está acortado, simplificado. El resto son los nombres descriptivos. **Ejemplo:** la integral cerrada se usa mucho  $\rightarrow$  está simplificada:

$$\oint = \text{\oint}$$

La doble integral cerrada sigue su desarrollo, pero no viene en Amsmath:  $\oint$ . La flecha a la derecha no es un símbolo matemático muy querido  $\rightarrow$  no se abrevia  $\rightarrow$

## Y continuamos

En  $\text{\LaTeX}$  existen dos formas de escribir fórmulas matemáticas. La razón es simple, estilo y formato. Estos dos modos son **Inline** y **Display**.

### Inline

Traducido al español: en línea. Se usa para meter símbolos y fórmulas **dentro del texto**. Este modo respetará el formato que posea el texto. Se accede con el signo del dólar, todo lo que caiga dentro, estará en modo **Inline**. Ejemplo:  $\frac{2^2}{4} = 1$   
`\frac{2^2}{4} = 1`, `\(...\)` también se permite.



# Y continuamos

En  $\text{\LaTeX}$  existen dos formas de escribir fórmulas matemáticas. La razón es simple, estilo y formato. Estos dos modos son **Inline** y **Display**.

## Inline

Traducido al español: en línea. Se usa para meter símbolos y fórmulas **dentro del texto**. Este modo respetará el formato que posea el texto. Se accede con el signo del dólar, todo lo que caiga dentro, estará en modo **Inline**. Ejemplo:  $\frac{2^2}{4} = 1$   
`\frac{2^2}{4} = 1`, `\(...\)` también se permite.

## Display

Traducción: demostración. Se utiliza para la escritura a parte de la expresión matemática, por ejemplo, para grandes ecuaciones, fórmulas importantes o elementos grandes (matrices). Genera un espacio nuevo para la fórmula. Se accede desde entornos o con la siguiente secuencia: `\[...\]`. Como ejemplo:

$$\frac{2^{23}}{4} = 2^{21}$$

`\[\frac{2^{23}}{4} = 2^{21}\]`.

## Alfabeto griego

Es especialmente expresivo, lo cual ayuda bastante a la hora de leer el código:

$\alpha$	<code>\alpha</code>	$\Gamma$	<code>\Gamma</code>
$\beta$	<code>\beta</code>	$\Delta$	<code>\Delta</code>
$\gamma$	<code>\gamma</code>	$\triangle$	<code>\varDelta</code>
$\varpi$	<code>\varpi</code>	$\Sigma$	<code>\Sigma</code>

## Y algún símbolo matemático

$\pm$	$\neq$	$\rightarrow$	
$\circ$ <code>\pm</code>	$\equiv$ <code>\neq</code>	$\Rightarrow$	<code>\rightarrow</code>
$\iiint$ <code>\circ</code>	$\approx$ <code>\equiv</code>	$\Leftrightarrow$	<code>\Longrightarrow</code>
$\sum$ <code>\iiint</code>	$\geq$ <code>\approx</code>	$\overbrace{abc}$	<code>\Longleftarrow</code>
$\prod$ <code>\sum</code>	$\propto$ <code>\geq</code>	$\overrightarrow{abc}$	<code>\overbrace{abc}</code>
			<code>\overrightarrow{abc}</code>

# Un par de ecuaciones para que os familiaricéis

## Ecuación de Bernoulli

$$\left( \frac{p}{\rho g} + \frac{\alpha}{2g} V^2 + z \right)_{ent} = \left( \frac{p}{\rho g} + \frac{\alpha}{2g} V^2 + z \right)_{sal} + h_{tur} + h_{fr} - h_{bom}$$

$$\left[ \left( \frac{p}{\rho g} + \frac{\alpha}{2g} V^2 + z \right)_{ent} = \left( \frac{p}{\rho g} + \frac{\alpha}{2g} V^2 + z \right)_{sal} + h_{tur} + h_{fr} - h_{bom} \right]$$

## Hermosas mates

$$\zeta(s) = \sum_{n=1}^{\infty} \frac{1}{n^s} \quad \Gamma(z) = \int_0^{\infty} t^{z-1} e^{-t} dt$$

$$\left[ \zeta(s) = \sum_{n=1}^{\infty} \frac{1}{n^s} \quad \Gamma(z) = \int_0^{\infty} t^{z-1} e^{-t} dt \right]$$

# Tablas

## Entorno tabular/array básico

Esto es la introducción básica general. **Nota:** las opciones de alineación son obligatorias.

```
\begin{tabular}{alineaciones} ... \end{tabular}
```

Ejemplo:

11	12	13
hola	hola	hola
adiós querida	adiós	Sayonara Baby

```
\begin{tabular}{l||c|r}
11          & 12 & 13 \\
\hline \hline
hola        & hola & hola \\
\hline
adiós querida & adiós & Sayonara Baby
\end{tabular}
```

# Matrices

Como las tablas, son fácilmente modificables:

## Arrays

Funciona igual que el entorno tabular pero se usa dentro del entorno de escritura matemática. Juntando esto con el `\left(...\right)` o cualquier otro símbolo podemos hacer matrices.

# Matrices

Como las tablas, son fácilmente modificables:

## Arrays

Funciona igual que el entorno tabular pero se usa dentro del entorno de escritura matemática. Juntando esto con el `\left(...\right)` o cualquier otro símbolo podemos hacer matrices.

## Pero $\text{\LaTeX}$ es bien eficiente

Las matrices son una herramienta bien usada, por lo que hay una forma sencilla. `\begin{*matrix}...\end{*matrix}`. No requiere de opciones de alineación. **\***: significa el tipo de puntuación a usar: **p**: paréntesis; **v**: vertical; **b**: corchetes; **B**: llaves. **Ejemplo:**

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \rightarrow \begin{vmatrix} 1 & 2 \\ 3 & 4 \end{vmatrix} = -2 \Rightarrow \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}^2 = \begin{bmatrix} 7 & 10 \\ 15 & 22 \end{bmatrix}$$

# El ejemplo de las matrices

```

\[\begin{pmatrix}
1 & 2 \\
3 & 4
\end{pmatrix} \rightarrow
\begin{vmatrix}
1 & 2 \\
3 & 4
\end{vmatrix} = -2 \rightarrow
\begin{bmatrix}
1 & 2 \\
3 & 4
\end{bmatrix}^2 =
\begin{bmatrix}
7 & 10 \\
15 & 22
\end{bmatrix}

```

## Entorno equation(\*)

### La base de la estructuración

`\begin{equation}...\end{equation}` tiene los mismos efectos que `\[...\]` sin embargo, para ecuaciones formales, importantes y largas es preferido. Da mayor claridad al código, otros entornos se pueden usar dentro suyo y permite la **referencia y numeración**. Para evitar la numeración se le pone un `*` al final de su declaración. La **numeración** puede modificarse para que tenga en cuenta el capítulo, sección, etc.



## Entorno equation(\*)

### La base de la estructuración

`\begin{equation}...\end{equation}` tiene los mismos efectos que `\[...\]` sin embargo, para ecuaciones formales, importantes y largas es preferido. Da mayor claridad al código, otros entornos se pueden usar dentro suyo y permite la **referencia y numeración**. Para evitar la numeración se le pone un `*` al final de su declaración. La **numeración** puede modificarse para que tenga en cuenta el capítulo, sección, etc.

### Ejemplo

$$f(x) = (x + a)(x + b) \tag{1}$$

Como se puede ver en 1, escribir en  $\text{\LaTeX}$  es bien sencillo.

```
\begin{equation} \label{ec:ejemplo1}
f(x)=(x+a)(x+b)
\end{equation}
```

# Desarrollando un poco más el tema...

## Subequations

Permite numerar distintas ecuaciones dentro de un mismo entorno y separación de las mismas. También permite cierta alineación.

## Desarrollando un poco más el tema...

### Subequations

Permite numerar distintas ecuaciones dentro de un mismo entorno y separación de las mismas. También permite cierta alineación.

### Align

**Entorno muy querido y usado.** Permite partir largas ecuaciones en diferentes líneas. Se usa mucho para demostraciones, ya que estas requieren un gran número de pasos e igualdades, por lo que se suelen partir en diferentes líneas. Numera cada nueva ecuación, por lo que se suele usar (\*). Si solo se quiere un número, se suele usar *split* o *aligned* o la opción `\nonumber` **Ejemplo:**

$$\begin{array}{ll}
 f(x) = (x + a)(x + b) & (2) \\
 = x^2 + (a + b)x + ab & (3)
 \end{array}$$

```

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

```

# Más flexibilidad en la escritura

$\text{\LaTeX}$  es conocido por ser muy formal, lo que puede echar atrás a entusiastas de las anotaciones sobre el texto, correcciones anotadas etc.  $\text{\LaTeX}$  no trae herramientas sencillas que hagan esta labor de manera bonita, pero podemos jugar con lo que tenemos.

## Anotaciones sobre fórmulas

- `\overset{}{} y \underset{}{} Nos permiten poner unos`  
 símbolos encima de otros. Ejemplo:  $A \stackrel{!}{=} B; A \stackrel{!}{=} B \rightarrow$   
`$A \overset{!}{=} B; A \stackrel{!}{=} B$.`

# Más flexibilidad en la escritura

$\text{\LaTeX}$  es conocido por ser muy formal, lo que puede echar atrás a entusiastas de las anotaciones sobre el texto, correcciones anotadas etc.  $\text{\LaTeX}$  no trae herramientas sencillas que hagan esta labor de manera bonita, pero podemos jugar con lo que tenemos.

## Anotaciones sobre fórmulas

- `\overset{}{}{}` y `\underset{}{}{}` Nos permiten poner unos símbolos encima de otros. Ejemplo:  $A \stackrel{!}{=} B$ ;  $A \stackrel{!}{=} B \rightarrow$   
`$A \overset{!}{=} B`; `A \stackrel{!}{=} B`.
- `\overbrace{}{}` y `\underbrace{}{}` Nos permiten coger trozos de

ecuaciones. Ejemplo:  $z = \overbrace{\underbrace{x}_{\text{real}} + \underbrace{iy}_{\text{imaginario}}}^{\text{número complejo}} \rightarrow$

`$z = \overbrace{\underbrace{x}_{\text{real}} + \underbrace{iy}_{\text{imaginario}}}^{\text{número complejo}}$`

# Tipografía

## Fuentes

- `\mathbb{}`: SOLO MAYÚSCULAS
- `\mathbf{}` **1234** **text**
- `\mathfrak{}` 1234 text  $\Re, \Im, \mathfrak{L}, \mathfrak{F}, \mathfrak{N}$
- `\mathrm{}` 1234 text
- `\mathcal{}` *SOLO MAYÚSCULAS*
- `\mathrm{}` 1234 text

# Cuadros

## Solo ecuación

```
\begin{equation} \boxed{f(x)=(x+a)(x+b)} \end{equation}
```

$$\boxed{f(x) = (x + a)(x + b)} \quad (4)$$

```
\fbox{\begin{equation} f(x)=(x+a)(x+b) \end{equation}}
```

## Toda la ecuación

$$f(x) = (x + a)(x + b) \quad (5)$$

```
\fbox{\begin{minipage}{\linewidth}  
\begin{equation}  
f(x)=(x+a)(x+b)  
\end{equation}  
\end{minipage}}
```

# Referencias

Aunque no es un procedimiento único al apartado científico, hacer referencias es prácticamente una necesidad.

Aquí es donde  $\text{\LaTeX}$  brilla de nuevo, ya que automatiza el proceso **independientemente del contenido**. Las referencias tienen dos partes.

## Label

`\label{key}` Es el comando con el que etiquetamos un contenido, ya sean párrafos, ecuaciones, imágenes, etc. `key` será el *código*, la etiqueta que nosotros le pondremos a nuestra referencia para luego llamarla cuando sea necesario.

## Reference

`\ref{text}` Nos permitirá llamar a nuestra referencia donde sea necesario. Solo tenemos que introducir el *código* para que la referencia sea creada.

**Ejemplo:** Véase la primera ecuación 1.



# Imágenes

## Imágenes

```
\begin{figure}
\centering
\vspace*{10px}
\includegraphics[height=0.6\linewidth]{images/Donald-Knuth-Sta
\caption{Donald Ervin Knuth. Creador de \TeX}
\label{fig:donald-knuth-stanford-computer-science}
\end{figure}
```

- `\includegraphics[keyvals]{imagefile}` [Keyvals] son los valores de tamaño; se está haciendo aritmética, se está cogiendo el 0.6 de todo el tamaño de línea. {Dirección relativa a la imagen desde nuestro archivo}
- `\caption{text}` Nota a pie de imagen.

# Microsoft Word

- Posible demostración de trabajo típico.

## La realidad de Word

- Fácil de usar, sencillo e intuitivo; gráfico; rápido.
  - Integrado en nuestra herramienta de trabajo. ¿Eficiente?
- 
- Ineficiente e inestable. **No es fácilmente modificable.**
  - Muy limitado (mayor expresividad con las herramientas de dibujo). Más cómodo  $\text{\LaTeX} 2_{\epsilon}$  en algunas áreas.

# Microsoft Word

- Posible demostración de trabajo típico.

## La realidad de Word

- Fácil de usar, sencillo e intuitivo; gráfico; rápido.
  - Integrado en nuestra herramienta de trabajo. ¿Eficiente?
- 
- Ineficiente e inestable. **No es fácilmente modificable.**
  - Muy limitado (mayor expresividad con las herramientas de dibujo). Más cómodo  $\text{\LaTeX} 2_{\epsilon}$  en algunas áreas.

## Soluciones

- Usa  $\text{\LaTeX}$  ya que sabes. Lo hace (casi)todo mejor.
- Trabaja en Word como si fuera  $\text{\LaTeX}$  (ahora más).

# L<sup>A</sup>T<sub>E</sub>X en Microsoft Word y amigos

- Word es un *phraser*, lo que significa que no procesa L<sup>A</sup>T<sub>E</sub>X sino que busca patrones y los traduce a su propia escritura. Lo que nos presenta grandes problemas.
- **Por suerte:** lo que funciona es lo que más solemos necesitar.

## Lo que funciona

Los sistemas más sencillos funcionan sin ningún problema. **Ejemplo:** símbolos, *exponentes*, las letras griegas: `\alpha` y nos la escribe.

**Procedimiento:** escribimos lo que queremos en L<sup>A</sup>T<sub>E</sub>X, pulsamos espacio y nos lo escribe.

## Lo que no funciona

Cualquier comando u estructura que utilice argumentos, por ejemplo los entornos. Para ellos tendremos que usar las herramientas que Word nos da.

# L<sup>A</sup>T<sub>E</sub>X en Matlab

## Meter L<sup>A</sup>T<sub>E</sub>X en Matlab

Sirve principalmente para escribir textos dentro de gráficos, como sus títulos u anotaciones.

- 1 Creamos una string como si estuviéramos en L<sup>A</sup>T<sub>E</sub>X. **Ejemplo:**  
`str = '$$ \int_{0}^{2} x^2 \sin(x) dx $$';`
- 2 Se le indica que nos la dibuje:  
`text(0.25,2.5,str,'Interpreter','latex')`

## Sacar L<sup>A</sup>T<sub>E</sub>X desde Matlab

Matlab posee un comando bien agradable para sacar sus expresiones en escritura L<sup>A</sup>T<sub>E</sub>X.

- 1 Localizamos el nombre nuestra estructura a escribir (matrices, fórmulas, funciones, etc).
- 2 `latex(estructura)` nos sacará lo que deseamos en formato L<sup>A</sup>T<sub>E</sub>X listo para ser copiado.

## Recursos y links

# Dudas

**Overleaf:** escritura de  $\text{\LaTeX}$  on-line.

**$\text{\TeX}$ Studio:** editor usado y recomendado.

**Libro:** *The not so Short Introduction to  $\text{\LaTeX}$ .*

**Correo:** 201507027@alu.comillas.edu