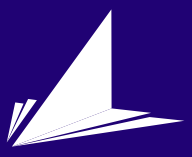


Escuela
Politécnica
Superior

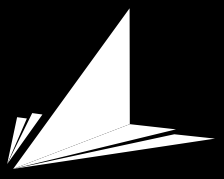
Título del Trabajo Fin de Grado/Máster

Grado en Ingeniería en Sonido e Imagen en Telecomunicación

Trabajo Fin de Grado

Autor:
Luis Pérez-Sala García-Plata
Tutores:
José Francisco Vicent Frances
Nombre Apellido1 Apellido2 (tutor2)

Mayo 2022



Escuela
Politécnica
Superior

Título del Trabajo Fin de Grado/Máster

Subtítulo del proyecto

Autor
Luis Pérez-Sala García-Plata

Tutores
José Francisco Vicent Frances
Ciencia de la Computacion e Inteligencia Artificial
Nombre Apellido1 Apellido2 (tutor2)
Departamento del cotutor

Grado en Ingeniería en Sonido e Imagen en Telecomunicación

ALICANTE, Mayo 2022

Preámbulo

Poner aquí un texto breve que debe incluir entre otras:

> "las razones que han llevado a la realización del estudio, el tema, la finalidad y el alcance y también los agradecimientos por las ayudas, por ejemplo apoyo económico (becas y subvenciones) y las consultas y discusiones con los tutores y colegas de trabajo. (**?**)"

Agradecimientos[1]

Este trabajo no habría sido posible sin el apoyo y el estímulo de mi colega y amigo, Doctor Rudolf Fliesning, bajo cuya supervisión escogí este tema y comencé la tesis. Sr. Quentin Travers, mi consejero en las etapas finales del trabajo, también ha sido generosamente servicial, y me ha ayudado de numerosos modos, incluyendo el resumen del contenido de los documentos que no estaban disponibles para mi examen, y en particular por permitirme leer, en cuanto estuvieron disponibles, las copias de los recientes extractos de los diarios de campaña del Vigilante Rupert Giles y la actual Cazadora la señorita Buffy Summers, que se encontraron con William the Bloody en 1998, y por facilitarme el pleno acceso a los diarios de anteriores Vigilantes relevantes a la carrera de William the Bloody.

También me gustaría agradecerle al Consejo la concesión de Wyndham-Pryce como Compañero, el cual me ha apoyado durante mis dos años de investigación, y la concesión de dos subvenciones de viajes, una para estudiar documentos en los Archivos de Vigilantes sellados en Munich, y otra para la investigación en campaña en Praga. Me gustaría agradecer a Sr. Travers, otra vez, por facilitarme la acreditación de seguridad para el trabajo en los Archivos de Munich, y al Doctor Fliesning por su apoyo colegial y ayuda en ambos viajes de investigación.

No puedo terminar sin agradecer a mi familia, en cuyo estímulo constante y amor he confiado a lo largo de mis años en la Academia. Estoy agradecida también a los ejemplos de mis difuntos hermano, Desmond Chalmers, Vigilante en Entrenamiento, y padre, Albert Chalmers, Vigilante. Su coraje resuelto y convicción siempre me inspirarán, y espero seguir, a mi propio y pequeño modo, la noble misión por la que dieron sus vidas.

Es a ellos a quien dedico este trabajo.

[1]Por si alguien tiene curiosidad, este "simpático" agradecimiento está tomado de la "Tesis de Lydia Chalmers" basada en el universo del programa de televisión Buffy, la Cazadora de Vampiros.http://www.buffy-cazavampiros.com/Spiketesis/tesis.inicio.htm

A mi esposa Marganit, y a mis hijos Ella Rose y Daniel Adams,
sin los cuales habría podido acabar este libro dos años antes [2]

[2]Dedicatoria de Joseph J. Roman en "An Introduction to Algebraic Topology"

Si consigo ver más lejos
es porque he conseguido auparme
a hombros de gigantes

Isaac Newton.

Índice general

Índice de figuras

Índice de tablas

Índice de Códigos

1. Introducción (Con ejemplos de contenido)

1.1. Motivación

Los accidentes de tráfico son uno de los principales problemas de Salud Pública en nuestros días. La multicausalidad, la variedad de las fuentes de datos y la poca cantidad de análisis específicos, apuntan a una gran complejidad en su tratamiento. Mas de 3.500 personas mueren, en las carreteras, diariamente en todo el mundo. Esto, significa casi 1,3 millones de muertes y aproximadamente 50 millones de lesiones cada año. Por lo tanto, la predicción de la gravedad de los accidentes de tráfico es un componente muy importante que se debe tener en cuenta, adoptando cualquier mejora en la predicción de la gravedad de los mismos. Además, el impacto económico y social asociado con los accidentes de tráfico lleva a las administraciones a buscar de manera activa mejoras. En el campo de la investigación sobre seguridad vial, el desarrollo de metodologías fiables para predecir y clasificar el nivel de gravedad, de los accidentes de tráfico, en función de diversas variables es un componente clave. La creciente disponibilidad de datos a gran escala de varias fuentes, incluidos los vehículos conectados y autónomos exige una comprensión profunda de las relaciones causales entre la seguridad y factores asociados con la ayuda de nuevas metodologías.

En las últimas dos décadas, los rápidos desarrollos en los métodos de aprendizaje automático (ML) y su regresión precisa y rendimiento de clasificación han atraído mucho la atención de los investigadores. Ha habido un número creciente de aplicaciones de métodos ML en la investigación de la gravedad de los accidentes. Aunque los métodos estadísticos tradicionales tienen formas funcionales rigurosas y claramente definidas, los métodos ML tienen una gran flexibilidad,requieren poca o ninguna suposición previa con respecto a los datos de gravedad de choques y son capaces de manejar valores perdidos, ruidos y valores atípicos.

A pesar de que la utilización de técnicas de inteligencia artificial (ML) en lugar de otras técnicas simples permite identificar las relaciones ocultas entre los factores de accidentes debido a la heterogeneidad de los datos de accidentes de tráfico, el principal inconveniente de estos modelos es que muy pocos estudios utilizan modelos de aprendizaje profundo para mejorar el rendimiento lo que se traduce en un bajo rendimiento en accidentes graves. Además, como los conjuntos de datos de accidentes de tráfico existentes, están muy desequilibrados, es difícil mejorar el rendimiento de las clases menos comunes como los accidentes graves. Basándose en lo expuesto anteriormente, en este trabajo se plantea estudiar y formalizar un modelo de aprendizaje profundo para predicción de la gravedad de los accidentes de tráfico. Con la finalidad de las características de los datos y preparar la entrada para el modelo de aprendizaje profundo, se ha utilizado una técnica para transformar números en matrices bidimensionales (Sharma et al., 2019) consistente en convertir un vector de características del accidente en una matriz de características, teniendo en cuenta que los vectores de características eran las variables de los datos del accidente y la matriz de características tenía las posiciones de las variables. Para fines de comparación, los resultados se han comparado con otros resultados

de modelos de aprendizaje automático.

1.2. Objetivos

1.2.1. Objetivos Generales

El objetivo principal de este trabajo final de Máster es desarrollar un sistema predictivo de la avedad de accidentes de tráfico utilizando características que se pueden identificar en los lugares del accidente, como el sexo del conductor, el tipo de vehículo, el entorno o los atributos de la carretera.

1.2.2. Objetivos Específicos

Con la finalidad de realizar este objetivo principal, se plantean los siguientes objetivos secundarios:

1. Utilización de técnicas para la transformación de características cualitativas de los accidentes de tráfico en matrices numéricas.
2. Utilización de algoritmos de aprendizaje automático basado en árboles de decisiones (XGBoost) para inferir pesos de las características de accidentes de tráfico.
3. Utilizar técnicas de algoritmia evolutiva para optimización de parámetros de entrada del algoritmo XGBoost.
4. Desarrollar un enfoque basado en el aprendizaje profundo, redes convolucionales, para predecir la gravedad de los accidentes de tráfico.
5. Comparar los resultados de los modelos de aprendizaje profundo con otros modelos utilizando indicadores de rendimiento.

1.2.3. Planificación

Diagrama de Grant
Antes de comenzar la lectura de este documento debo agradecer el trabajo realizado por Pedro Pernías Peco en su plantilla de "tfg" que se puede ver en `https://github.com/lcg51/tfg`. Gracias a esa plantilla me he lanzado a crear mi versión. Algunos contenidos aquí mostrados han sido extraídos de la plantilla de Pedro.

Esta plantilla se ha diseñado de 0 y por ello no utiliza la misma estructura que la plantilla de Pedro. Pero la estructura de contenido para un TFG/TFM es la misma y a continuación se muestran las diferentes partes que debe tener un TFG/TFM redactado por Pedro.

1.3. ¡Importante!, leer primero

Este texto está escrito pensando en orientar a los alumnos que usarán LaTeX para escribir su Trabajo Final de Grado (TFG) y Trabajo Final de Máster (TFM).

Contiene información útil para aquellos que no tengan experiencia previa en LaTeX así como algunos datos acerca de cómo escribir mejor su TFG. A continuación, se ofrece una copia de la información que hay en el libro de estilo para la realización de los TFG de la EPS de la Universidad de Alicante.

En los capítulos siguientes encontrarás ejemplos de muchas de las cosas que se pueden realizar con LaTeX. Con un poco de paciencia, estudia cómo se hacen estas cosas y luego aplícalas en tus documentos.

1.4. Estructura de un TFG

En caso de que el TFG/TFM tenga como finalidad la elaboración de un proyecto o un informe científico o técnico, deberá ajustarse a lo dispuesto en las normas UNE 157001:2002 y UNE 50135:1996 respectivamente.

Si el TFG/TFM tiene por finalidad la elaboración de un trabajo monográfico, el documento presentado deberá constar de las siguientes partes, teniendo como base la norma UNE 50136:1997.

Preámbulo: se describirán brevemente la motivación que ha originado la realización del TFG/TFM, así como una breve descripción de los objetivos generales que se quieren alcanzar con el trabajo presentado.

Agradecimientos: se podrán añadir las hojas necesarias para realizar los agradecimientos, a veces obligatorios, a las entidades y organismos colaboradores.

Dedicatoria: se podrá añadir una única hoja con dedicatorias, su alineación será derecha.

Citas: (frases célebres) se podrá añadir una única hoja con citas, su alineación será derecha.

Índices: cada índice debe comenzar en una nueva página, se incluirán los índices que se estimen necesarios (conforme UNE 50111:1989), en este orden:

- **Índice de contenidos:** (obligatorio siempre) se incluirá un índice de las secciones de las que se componga el documento, la numeración de las divisiones y subdivisiones utilizarán cifras arábigas (según UNE 50132:1994) y harán mención a la página del documento donde se ubiquen.
- **Índice de figuras:** si el documento incluye figuras se podrá incluir también un índice con su relación, indicando la página donde se ubiquen.
- **Índice de tablas:** en caso de existir en el texto, ídem que el anterior.
- **Índice de abreviaturas, siglas, símbolos, etc.:** en caso de ser necesarios se podrán incluir cada uno de ellos.

Cuerpo del documento: en el contenido del documento se da flexibilidad para su organización y se puede estructurar en las secciones que se considere. En todo caso obligatoriamente se deberá, al menos, incluir los siguientes contenidos:

- **Introducción:** donde se hará énfasis a la importancia de la temática, su vigencia y actualidad; se planteará el problema a investigar, así como el propósito o finalidad de la investigación.

Marco teórico o Estado del arte: se hará mención a los elementos conceptuales que sirven de base para la investigación, estudios previos relacionados con el problema planteado, etc.

Objetivos: se establecerán el objetivo general y los específicos.

Metodología: se indicarán el tipo o tipos de investigación, las técnicas y los procedimientos que serán utilizados para llevarla a cabo; se identificarán la población y el tamaño de la muestra así como las técnicas e instrumentos de recolección de datos.

Resultados: incluirá los resultados de la investigación o trabajo, así como el análisis y la discusión de los mismos.

Conclusiones: obligatoriamente se incluirá una sección de conclusiones donde se realizará un resumen de los objetivos conseguidos así como de los resultados obtenidos si proceden.

Bibliografía y referencias: se incluirá también la relación de obras y materiales consultados y empleados en la elaboración de la memoria del TFG/TFM. La bibliografía y las referencias serán indexadas en orden alfabético (sistema nombre y fecha) o se numerará correlativamente según aparezca (sistema numérico). Se empleará la familia 1 como tipo de letra. Podrá utilizarse cualquier sistema bibliográfico normalizado predominante en la rama de conocimiento, estableciéndose como prioritarios el sistema ISO 690, sistema American Psychological Association (APA) o Harvard (no necesariamente en ese orden de preferencia). En esta plantilla Latex se propone usar el estilo APA indicándolo en la línea correspondiente como

```
\bibliographystyle{apacite}
```

Anexos: se podrán incluir los anexos que se consideren oportunos.

1.5. Apartados dentro de los capítulos

En LaTeX existen diferentes niveles de títulos para realizar secciones, subsecciones, etc. En esta web puedes ver más información al respecto `https://en.wikibooks.org/wiki/LaTeX/Document_Structure`

Para ello se utilizan los siguientes comandos;

```
\section{Esto es una sección}
Y este el contenido de la sección.
\subsection{Esto es una subsección}
Y este el contenido de la subsección.
\subsubsection{Esto es una subsubsección}
Y este el contenido de la subsubsección.
\paragraph{Esto es un paragraph}
Y este el contenido del paragraph. Que siempre se inicia en la misma línea que el título del mismo.
```

Y se genera lo siguiente:

1.6. Esto es una sección

Y este el contenido de la sección.

1.6.1. Esto es una subsección

Y este el contenido de la subsección.

1.6.1.1. Esto es una subsubsección

Y este el contenido de la subsubsección.

1.6.1.1.1. Esto es un paragraph Y este el contenido del paragraph. Que siempre se inicia en la misma línea que el título del mismo.

1.7. Citar bibliografía

Para citar la bibliografía tal como se define en el sistema APA (en esta web se indica como debe aparecer en el texto la cita: `http://guides.libraries.psu.edu/apaquickguide/intext`) se debe realizar con alguno de los comandos mostrados a continuación:

```
Esto es una cita estándar: \citet{Shaw1996}, que también puedes mostrar con paréntesis así: \citep{Shaw
    ↪ 1996}. También se puede realizar una cita indicando a qué parte te refieres \citep[ver][Cap. 2]{Shaw
    ↪ 1996} o \citep[Cap. 2]{Shaw1996} o \citep[ver][]{Shaw1996}.

También puedes mostrar todos los autores cuando hay más de 2 autores añadiendo un asterisco después del
    ↪ comando como: \citet*{Akyildiz2005}, sin el asterisco quedaría así: \citet{Akyildiz2005}.

O puedes citar dos o más fuentes al mismo tiempo: \citep{Barkan1995,Leighton2012}
```

Y LaTeX genera lo siguiente:

Esto es una cita estándar: **?**, que también puedes mostrar con paréntesis así: (**?**). También se puede realizar una cita indicando a qué parte te refieres (ver **?**, Cap. 2) o (**?**, Cap. 2) o (ver **?**).

También puedes mostrar todos los autores cuando hay más de 2 autores añadiendo un asterisco después del comando como: **?**, sin el asterisco quedaría así: **?**.

O puedes citar dos o más fuentes al mismo tiempo: (**??**)

1.8. Notas a pie de página

Para introducir notas a pie de página se debe escribir lo siguiente:

```
La plantilla necesita el motor XeLaTeX \footnote{Para más información sobre XeLaTeX visita \url{https
    ↪ ://es.sharelatex.com/learn/XeLaTeX}} (el más recomendable actualmente), por lo que si el
    ↪ programa que utilizas compila la plantilla con el motor pdfLaTeX \footnote{También puedes
    ↪ buscar más información en internet} (el más habitual pero menos potente) debes cambiarlo por
    ↪ XeLaTeX en las opciones del programa. Si no sabes como hacerlo busca en el manual del programa
    ↪  o en google.
```

LaTeX genera lo siguiente (observa las notas a pie de página):

La plantilla necesita el motor XeLaTeX[1] (el más recomendable actualmente), por lo que si el programa que utilizas compila la plantilla con el motor pdfLaTeX[2] (el más habitual pero menos potente) debes cambiarlo por XeLaTeX en las opciones del programa. Si no sabes como hacerlo busca en el manual del programa o en google.

1.9. Estilos de texto

A continuación se muestran ejemplos de distintos estilos de texto:

- \textit{Cursiva} → *Cursiva*
- \emph{Cursiva 2} → *Cursiva 2*
- \textbf{Negrita} → **Negrita**
- \texttt{Monoespacio} → `Monoespacio`
- \textsc{Mayúsculas capitales} → MAYÚSCULAS CAPITALES
- \uppercase{Todo mayúsculas} → TODO MAYÚSCULAS

1.10. Acrónimos

Ahora vamos a ver cómo se ponen los acrónimos.

La norma dice que la primera vez que aparece un acrónimo debe ponerse su fórmula completa, es decir lo que significa, al lado del acrónimo. Después de ello, podemos usar sólo el acrónimo salvo cuando consideremos que debemos volver a usar la fórmula completa por alguna razón de legibilidad.

¿Cómo llevar la cuenta de cuándo es la primera vez que ponemos el acrónimo? si hacemos cambios en el doc es fácil que perdamos esa información así que lo mejor es que sea el propio LaTeX el que lleve esa cuenta. Para ello tenemos que hacer dos cosas:

Primero: creamos la entrada del acrónimo en el fichero acronimos.tex. Revisa los comentarios de su cabecera para saber cómo crear esa entrada. Básicamente lo que hacemos allí es poner la "fórmula corta" y la "fórmula larga" del acrónimo es decir, el propio acrónimo y su significado

Segundo: escribimos en el texto el acrónimo SIEMPRE diciendo que es un acrónimo y el tipo de fórmula que queremos usar. Por ejemplo, si siempre que queremos hacer referencia al IEEE escribimos

```
\gls{ieee}
```

se consigue que la primera vez que aparezca el acrónimo ponga las fórmulas larga y corta y en las siguientes ocasiones sólo aparecerá la corta.

[1] Para más información sobre XeLaTeX visita `https://es.sharelatex.com/learn/XeLaTeX`

[2] También puedes buscar más información en internet

Aquí va un ejemplo:
Si escribimos:

```
El \gls{ieee} es una institución muy importante en el mundo de la
ingeniería. El \gls{ieee} lleva marcando normas y protocolos desde
hace mucho tiempo. Pero el \gls{ieee} no está solo en esta tarea.
Además del \gls{ieee} hay muchas otras instituciones para ello.
```

Obtendremos:
El Institute of Electrical and Electronics Engineers (IEEE) es una institución muy importante en el mundo de la ingeniería. El IEEE lleva marcando normas y protocolos desde hace mucho tiempo. Pero el IEEE no está solo en esta tarea. Además del IEEE hay muchas otras instituciones para ello.

1.11. Tareas por hacer

En esta plantilla se ha incluido un paquete para incluir notas/comentarios en el texto para recordar partes que hay que revisar o terminar de desarrollar. El uso es sencillo, el manual para conocer todos los comandos se encuentra en `http://osl.ugr.es/CTAN/macros/latex/contrib/todonotes/todonotes.pdf`, a continuación se muestran algunos ejemplos:

Para incluir un comentario sobre el texto:

```
Recomiendo utilizar programas LaTeX que permitan trabajar con sistema de archivos para poder editar el
↪ conjunto de capítulos en la misma ventana. Este tipo de función lo tienen programas como
↪ TexStudio, es multiplataforma. \todo{Incluir más ejemplos de programas}
```

LaTeX genera lo siguiente:

Recomiendo utilizar programas LaTeX que permitan trabajar con sistema de archivos para poder editar el conjunto de capítulos en la misma ventana. Este tipo de función lo tienen programas como TexStudio, es multiplataforma.

Incluir más ejemplos de programas

Para incluir un comentario sobre el texto pero dentro del texto:

```
Recomiendo utilizar programas LaTeX que permitan trabajar con sistema de archivos para poder editar el
↪ conjunto de capítulos en la misma ventana. Este tipo de función lo tienen programas como
↪ TexStudio, es multiplataforma. \todo[inline]{Incluir más ejemplos de programas}
```

LaTeX genera lo siguiente:

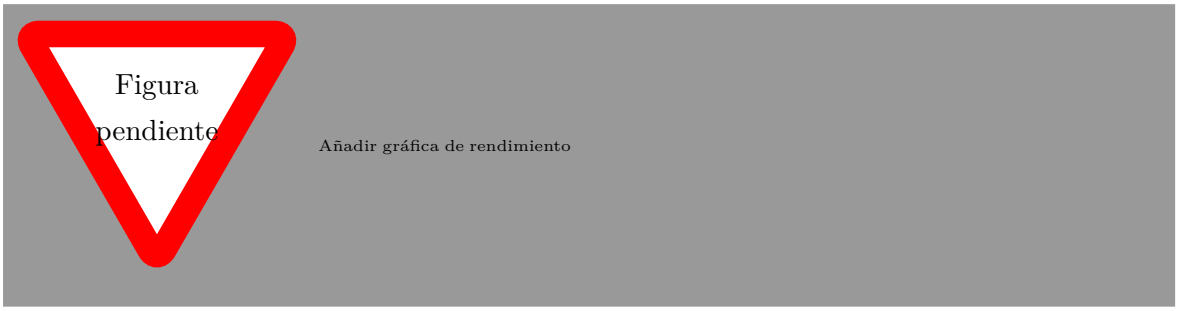
Recomiendo utilizar programas LaTeX que permitan trabajar con sistema de archivos para poder editar el conjunto de capítulos en la misma ventana. Este tipo de función lo tienen programas como TexStudio, es multiplataforma.

Incluir más ejemplos de programas

También se puede dejar indicado donde falta una imagen o figura, para incluirla más adelante del siguiente modo:

```
\missingfigure{Añadir gráfica de rendimiento}
```

LaTeX genera lo siguiente:



2. Marco Teórico (Con ejemplos de listas)

2.1. Listas

Hacer una lista es simple en LaTeX. Para ello has de crear un entorno (así se llama) itemize con

```
\begin{itemize}
...
\end{itemize}
```

Y dentro de esa estructura, añadir cada elemento de la lista precedido de

```
\item primer ítem de lista
\item segundo ítem de lista
...
\item ultimo ítem de lista
```

Es importante que revises este texto tal como aparece en la plantilla y relaciones el aspecto que tiene el PDF final con cómo está escrito el documento LaTeX.

Aquí va una lista con subtérminos:

```
    \begin{itemize}
    \item Ingeniería Informática.
    \item Ingeniería Sonido e Imagen en Telecomunicación.
    \item Ingeniería Multimedia.
        \subitem Mención: Creación y ocio digital.
        \subitem Mención: Gestión de Contenidos.
    \end{itemize}
```

El resultado es el siguiente:

- Ingeniería Informática.
- Ingeniería Sonido e Imagen en Telecomunicación.
- Ingeniería Multimedia.

    Mención: Creación y ocio digital.

    Mención: Gestión de Contenidos.

Aquí va una lista con subtérminos pero numerada:

```
\begin{enumerate}
    \item Ingeniería Informática.
    \item Ingeniería Sonido e Imagen en Telecomunicación.
```

```
    \item Ingeniería Multimedia.
    \begin{enumerate}
        \item Mención: Creación y ocio digital.
        \item Mención: Gestión de Contenidos.
    \end{enumerate}
\end{enumerate}
```

El resultado es el siguiente:

1. Ingeniería Informática.
2. Ingeniería Sonido e Imagen en Telecomunicación.
3. Ingeniería Multimedia.
    a) Mención: Creación y ocio digital.
    b) Mención: Gestión de Contenidos.

2.2. Listas de definición

Puedes realizar una lista de conceptos con su definición del siguiente modo:

```
\begin{description} % Inicio de la lista
    \item[MAPP XT:] Programa desarrollado por \textit{Meyer Sound} para el diseño y ajuste de sistemas ↩
        ↪ formados por altavoces de su marca.
    \begin{description} % Realiza una lista dentro de la lista
        \item[Ventajas:]~
        El programa permite realizar múltiples ajustes tal como se podría realizar en la realidad con un ↩
            ↪ procesador real.

        Permite analizar la fase recibida en cualquier punto y compararla con otras mediciones.

        Dispone de varios tipos de filtros, inversiones de fase, etc.
        \item[Inconvenientes:]~
        No existe una lista global de los altavoces ubicados en el plano, por lo tanto solo se pueden editar ↩
            ↪ seleccionándolos sobre el plano.

        Sólo permite diseñar en 2 dimensiones, principalmente sobre la vista lateral ya que los array de ↩
            ↪ altavoces no permite voltearlos.
    \end{description}
\end{description}
```

Y LaTeX genera lo siguiente:

MAPP XT: Programa desarrollado por *Meyer Sound* para el diseño y ajuste de sistemas formados por altavoces de su marca.

Ventajas: El programa permite realizar múltiples ajustes tal como se podría realizar en la realidad con un procesador real.

Permite analizar la fase recibida en cualquier punto y compararla con otras mediciones.

Dispone de varios tipos de filtros, inversiones de fase, etc.

Inconvenientes: No existe una lista global de los altavoces ubicados en el plano, por lo tanto solo se pueden editar seleccionándolos sobre el plano.

Sólo permite diseñar en 2 dimensiones, principalmente sobre la vista lateral ya que los array de altavoces no permite voltearlos.

3. Objetivos (Con ejemplos de tablas)

3.1. Tablas

Ahora veremos otra estructura más: las tablas.
Aquí va una tabla[1] para que se vea cómo insertar una tabla simple dentro del documento.

```
\begin{table}[h]
    \centering
    \begin{tabular}{lllll}
        &columna A&columna B&columna C\\
        \hline
        fila 1&fila 1, columna A & fila 1, columna B & fila 1, columna C\\
        fila 2&fila 2, columna A & fila 2, columna B & fila 2, columna C\\
        fila 3&fila 3, columna A & fila 3, columna B & fila 3, columna C\\ \hline
    \end{tabular}
    \caption{Ejemplo de tabla.}
    \label{tabladeejemplo}
\end{table}
```

	columna A	columna B	columna C
fila 1	fila 1, columna A	fila 1, columna B	fila 1, columna C
fila 2	fila 2, columna A	fila 2, columna B	fila 2, columna C
fila 3	fila 3, columna A	fila 3, columna B	fila 3, columna C

Tabla 3.1: Ejemplo de tabla.

LaTeX usa un sistema de parámetros para "decorar" las tablas. Puedes consultar estos parámetros en la tabla 3.2 de la página 14. La tabla se ubicará donde, a juicio de LaTeX, menos moleste por lo que puede no aparecer necesariamente donde se ha insertado en el texto original.

Existe la posibilidad de forzar que las tablas, figuras u otros objetos aparezcan en la zona del texto que se desea aunque en ocasiones puede dejar grandes espacios en blanco. El comando a utilizar es:

```
\FloatBarrier
```

Que introducido justo después de una tabla, figura, etc (despues del comando \end{...}) fuerza la aparición en el texto, empujando el contenido.

[1] En http://www.tablesgenerator.com/ se puede encontrar un generador On-Line de tablas para LaTeX

Parámetro	Significado
`h`	Situa el elemento flotante *preferentemente* (es decir, si es posible) en la situación exacta donde se incluye este
`t`	Sitúa el elemento en la parte de arriba de la página
`b`	Sitúa el elemento en la parte de abajo de la página
`p`	Sitúa el elemento en una página aparte dedicada sólo a elementos flotantes; en el caso del formato `article`, ésta se sitúa al final del documento, mientras que para al book es colocada al final de cada capítulo

Tabla 3.2: Parámetros optativos de los entornos flotantes

También es posible elegir el ancho de cada columna y la orientación del texto en cada una. Por ejemplo:

```
\begin{table}[ht]
    \centering
    \begin{tabular}{|C{2cm}|C{2cm}|C{2cm}|C{2cm}|} % 4 columnas de 2cm - texto centrado y con bordes
        \hline
        \multicolumn{4}{|c|}{\textbf{\begin{tabular}[c]{@{}c@{}}FUENTE: TRÁFICO RODADO\\ ↩
            ↪ HORARIO: TARDE\end{tabular}}} \\ \hline
        \textbf{dB(A)} & \textbf{Población expuesta tarde} & \textbf{\%} & \textbf{\scriptsize{↩
            ↪ CENTENAS}} \\ \hline
        \textbf{\textgreater70} & 0 & 0,000 & 0 \\ \hline
        \textbf{65 - 70} & 348,9 & 9,792 & 3 \\ \hline
        \textbf{60 - 65} & 1594,7 & 44,757 & 16 \\ \hline
        \textbf{55 - 60} & 322,1 & 9,040 & 3 \\ \hline
        \textbf{50 - 55} & 0 & 0,000 & 0 \\ \hline
        \textbf{\textgreater50} & 1297,3 & 36,410 & 13 \\ \hline
        \textbf{TOTAL} & 3563 & 100 & 35 \\ \hline
    \end{tabular}
    \label{my-label}
\end{table}
```

LaTeX genera esto:

FUENTE: TRÁFICO RODADO HORARIO: TARDE			
dB(A)	**Población expuesta tarde**	**%**	**CENTENAS**
>70	0	0,000	0
65 - 70	348,9	9,792	3
60 - 65	1594,7	44,757	16
55 - 60	322,1	9,040	3
50 - 55	0	0,000	0
>50	1297,3	36,410	13
TOTAL	3563	100	35

Donde C{2cm} indica que la columna tiene el texto centrado y un ancho de 2 cm. Tambien

se puede utilizar L{} o R{} para poner el texto a la izquierda o derecha y definir un ancho concreto.

Páginas como `https://www.tablesgenerator.com/` ayudan a realizar tablas fácilmente, es lo más recomendado, ahorra mucho tiempo de trabajo y luego si falta algún detalle se puede retocar en el documento.

El formato estándar de las columnas es c, l o r, así lo genera la web mencionada antes, pero una vez generada puedes cambiar ese formato por el definido anteriormente para ajustar el ancho de las columnas, o mantenerlo así si el resultado ya es el deseado.

Para conocer más sobre las tablas puedes leer manuales como este: `https://latexlive.files.wordpress.com/2009/04/tablas.pdf` que contiene muchos ejemplos y explicaciones.

3.2. Otros diseños de tablas

Modelo	15LEX1600Nd	15P1000Fe V2
fs (Hz)	41	45
Re (ohm)	5.5	5.2
Le (μH)	1600	1500
Bl (N/A)	25.7	27.4
M_{MS} (g)	175	157
C_{MS} ($\mu m/N$)	84	78
R_{MS} (kg/s)	6.8	7.6
d (cm)	33.5	33
Vas (dm^3)	91	80.7
Q_{TS}	0.36	0.30
Q_{MS}	6.6	5.9
Q_{ES}	0.38	0.31
Sens (dB @ 2.83V/1m)	96	98
η	1.7%	2.4%
Sd (cm^2)	880	855

Tabla 3.3: Parámetros de los altavoces elegidos de la marca Beyma®.

		140PU				50PU			
		Phase II		Phase I		Phase II		Phase I	
# BJet		≥ 4	2 or 3	≥ 4	2 or 3	≥ 4	2 or 3	≥ 4	2 or 3
# Bkg		123	76	12	7	84	35	7	3
Asimov	NM1	13	6	9	3	15	9	11	4
	NM2	6	2	4	1	7	3	5	1
	NM3	3	1	2	0	4	1	2	0
	STC	6	3	4	1	7	5	5	2

Tabla 3.4: Ejemplo 2

4. Tecnologías

4.1. Modelos

A continuación se presentarán los modelos con los que se ha experimentado para resolver este problema.

4.1.1. Algoritmos Genéticos

Un *Algoritmo Genético*, o *Genetic Algorithm (GA)*, es un modelo que imita la evolución de las especies en el ámbito de la biología, con el objetivo de encontrar una solución potencialmente óptima a un problema mediante la evolución de un conjunto de soluciones denominadas individuos.

El planteamiento de estos modelos consiste en evaluar los individuos pertenecientes a la población de la generación correspondiente para tomar el subconjunto de aquellas soluciones que mejor calidad ofrezcan (adaptación al medio) mediante una función heurística.

Las funciones heurísticas son funciones aproximadas a una solución ideal del problema, debido a que todas las posibles combinaciones de los parámetros a optimizar genera un espacio de búsqueda de búsqueda exponencial (problema NP-hard), es necesario crear una aproximación mediante la función heurística para acotar el espacio de búsqueda.

Para comprender el funcionamiento de los algoritmos genéticos es necesario analizar cada una de las fases que lo constituyen:

1. Inicialización de la Población:

   En primer lugar es necesario generar los n individuos de la población aleatoriamente. Cada uno de estos individuos está formado por el conjunto de parámetros que son objeto de optimización 4.1.

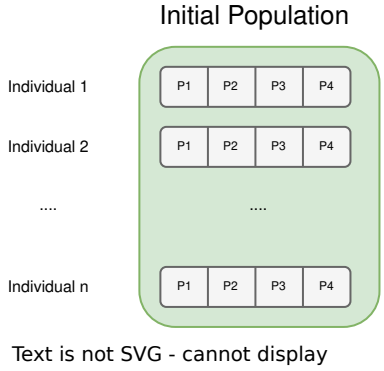


Figura 4.1: Inicialización de la población.

2. Evaluación de Población:

   Es la primera fase del proceso iterativo del algoritmo genético. En él, cada uno de los individuos pertenecientes a la población son evaluados mediante la función *fitness* para obtener el *score* de cada individuo de la población actual.

3. Selección de Padres:

   Una vez evaluados los individuos de la población, se procede a seleccionar aquellos padres que mejor *score* han obtenido para crear nuevas soluciones a partir de éstos. Existen distintas técnicas de selección, como por ejemplo escoger aquellos n mejores padres de la población o técnicas basadas en métodos probabilísticos, para que aquellos individuos que menos puntuación logren, tengan alguna probabilidad de ser elegidos para la generación de nuevas soluciones. Este tipo de técnicas se utilizan para aumentar la diversidad en la poblĺación y evitar caer en soluciones acotadas a mínimos locales del problema. Se puede ver el proceso de cómo son seleccionados los padres en función de su *fitness* en la figura 4.2.

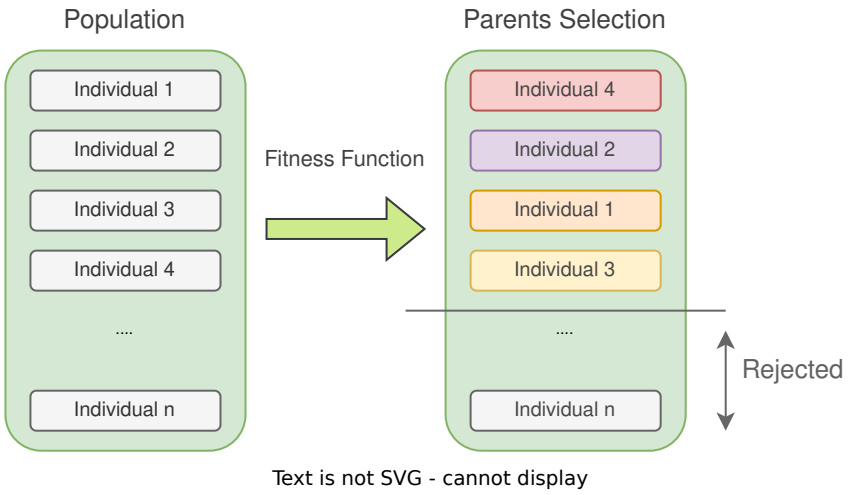


Figura 4.2: Selección de padres candidatos.

4. Cruzamiento:

   Una vez seleccionados los padres es necesario que intercambien información entre ellos para dar lugar a nuevos individuos, este proceso de herencia de información es posible realizarlo aplicando distintas filosofías; seleccionar un punto o posición de cruzamiento a lo largo del vector de los padres y combinar ambos padres, o escoger aleatoriamente las posiciones de los parámetros de ambos padres y se combinen para dar lugar a la solución generada 4.3.

5. Mutación:

   Cuando los padres seleccionados son combinados para dar lugar a los candidatos de la nueva generación, es importante aplicar un proceso de mutación entre éstos debido a la necesidad de generar diversidad en la población. Si se combina la misma información

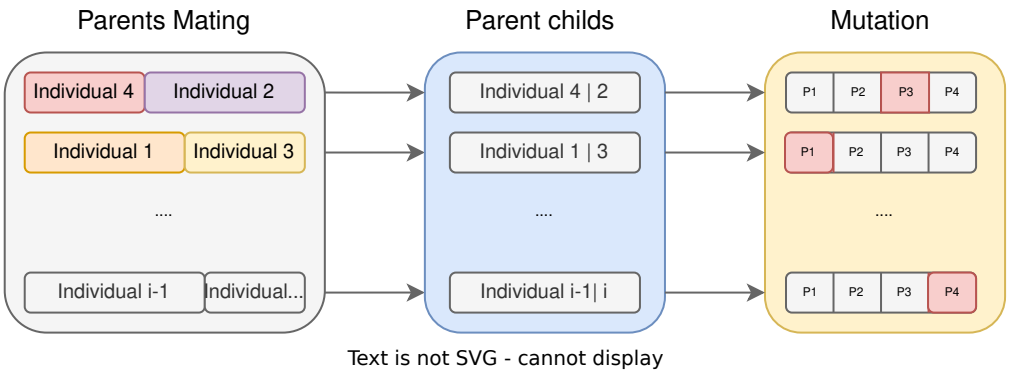


Figura 4.3: Mutación de hijos.

entre generaciones, se corre el riesgo de converger prematuramente a un mínimo local del espacio de búsqueda. Por este motivo se introduce el concepto de mutación, que consiste en aplicar una modificación aleatoria sobre alguno de los parámetros de los individuos generados para poder ampliar el espacio de soluciones.

Cada una de las fases enumeradas anteriormente se repite de forma iterativa entre generaciones hasta que se cumpla alguna condición de parada, como establecer un umbral mínimo del *score* de las soluciones, o fijar un número específico de iteraciones que se ejecutarán en el algoritmo.

A continuación enumeraremos los parámetros con los que configurar un algoritmo genético. Dependiendo del tipo de algoritmo utilizado es posible encontrar distintos parámetros con los que inicializar el algoritmo, sin embargo estos son los más extendidos:

- Tamaño de la población:

  Ees el número de individuos que sobrevivirán entre iteraciones, es decir, los individuos que se tratarán de mejorar a lo largo de las ejecuciones. En los experimentos de este trabajo Sección 5 se han inicializado 100 individuos, cada uno de ellos con 4 hiperparámetros del algoritmoo *XGBoost* a optimizar.

- Número de iteraciones:

  Es el número generaciones que evolucionarán en el algoritmo. Habitualmente un mayor número de iteraciones permite una mayor precisión de los resultados a costa de invertir recursos computacionales, debido a que existen mayores probabilidades de que los individuos evolucionen. En los experimentos se han usado el mismo número de iteraciones para cada uno de los algoritmos. Esto provoca que el tiempo de ejecución entre algoritmos varíe debido a su propia naturaleza.

- Probabilidad de cruce:

  Probabilidad con la que dos padres intercambian su información entre sí para dar lugar a un nuevo individuo hijo.

- Probabilidad de mutación:

Indica la probabilidad con la que cada hiperparámetro puede variar su valor, es decir, con una probabilidad de 0.5, cada hiperparámetro de un individuo hijo tiene un 50 por ciento de probabilidades de modificar su valor. Se establece además un rango de posibles valores en los cuales un párámetro de un individuo puede mutar, es decir, se definen unos valores máximos y mínimos en los que el parámetro mutará y se escoge un valor aleatorio dentro del rango. De esta forma si muta un mismo parámetro para varios individuos de la generación, esta condición se asegurará de que las variaciones aplicadas a cada uno de ellos estén controladas y sean distintas para cada uno de ellos.

4.1.2. XGBoost

Extreme Gradient Boosting Algortihm o *XGBoost* es una libería open-source que implementa . Se trata de uno de los algoritmos más importantes de los últimos años, logrando un rendimiento diez veces mayor a otras soluciones populares en una misma máquina y llegando a las primeras posiciones en numerosos retos propuestos en *Kaggle* (**?**).

XGBoost se basa en los modelos *Ensemble Boosting* de los algoritmos de *ML*, de ahí su nombre *Gradient Boosting*. La técnica *Boosting* se basa en construir un modelo robusto (*strong learner*) combinando una serie de modelos débiles (*weak learners*).

https://www.nvidia.com/en-us/glossary/data-science/xgboost/

Estos construyen un modelo predictivo mediante la combinación de otros modelos, la salida al predecir una muestra por el Ensemble será la combinación de la predicción de cada uno de los modelos individuales, y sobre sus resultados se pueden aplicar distintas estrategias para escoger la predicción agregada de éstos.

XGBoost implementa en su arquitectura *Ensemble Boosting* una serie de Árboles de Decisión, cada uno de los cuales

---

Además, *XGBoost* aplica técnicas de regularización en su función de pérdida, de tal forma que reduce la influencia individual de cada árbol generado y de sus hojas para poder dar lugar a posteriores árboles que consigan mejorar el modelo, evitando de esta manera el sobreajuste u *overfitting*.

4.1.3. Redes Neuronales Convolucionales (CNN)

Las Redes Neuronales Convolucionales tratan de imitar el comportamiento del sistema nervioso para identificar patrones en los datos de entrada.

Para entrenar a una *CNN* es necesario hacerlo mediante ejemplos etiquetados, ya que este tipo de redes se basa en el aprendizaje supervisado. Es decir, durante el proceso de entrenamiento se comparará la salida de las predicciones de la última capa con la salida conocida, y se calculará la entropía cruzada entre las distribuciones para optimizar los pesos de las capas mediante el método *Descenso por Gradiente (Backward Propagation)*

Antes de entrar en el detalle de la implementación de las *CNN* es necesario introducir los siguientes conceptos comunes a todas las *NN* para la posterior comprensión de las arquitecturas:

- Entropía:

La entropía sobre una varible aleatoria X es el grado de incertidumbre producido en base a los posibles valores que puede tomar como respuesta, es decir, a mayor entropía de una variable aleatoria X, existe mayor grado de incertidumbre sobre ella. La entropia $H(X)$ se calcula en base a la siguiente fórmula:

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i)$$

Donde x_i es cada uno de los posibles valores que puede tomar la variable como respuesta y $p(x_i)$ es la probabilidad de obtener dicho valor. Por lo tanto, es deseable mantener distribuciones con un grado de entropía bajo, ya que existirá menos incertidumbre en ella.

- Softmax:

  Se trata de la generalización en forma multidimensional de la función logística o *logistic function*. Softmax es una función que permite la transformación de un vector de números reales a una distribución de probabilidad. Esta función se suele aplicar como capa de activación en el último nivel en las *NN*, representando la probabilidad de que la salida de la red pertenezca a cada una de las K clases posibles en el modelo **?**.

  Esto se debe a que normalmente los componentes de la salida de la última capa de una red neuronal (denominados como *logits*) son el resultado de la predicción en forma de números reales no normalizados, por lo tanto pueden ser mayores que 1 e incluso negativos. La función permite calcular la distribución de probabilidad en base a estos *logits* y calcular la probabilidad de pertenencia a cada una de las K clases que componen el modelo mediante la siguiente fórmula:

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad for\ i = 1, 2, \ldots, K$$

  Esta función es continua y diferenciable, por lo que es posible aplicar el método *GD* para actualizar los valores de los pesos de la red neuronal en las capas anteriores gracias a estas propiedades que permiten la derivabilidad de la función.

- Cross-Entropy:

  El objetivo de cross-entropy es minimizar la función de pérdida (*log loss*) comparando la probabilidad de la clase predicha con respecto a su valor verdadero. Al aplicar una penalización logarítmica, las probabilidades de las clases que más disten respecto a su valor verdadero se verán más acentuadas **?**.

$$L_{CE} = -\sum_{i=1}^{n} y_i \log_2(p_i)$$

  Donde n es el número de clases a predecir, el valor y_i es la etiqueta de la clase verdadera, y p_i es la probabilidad de que la muestra actual pertenezca a la clase i.

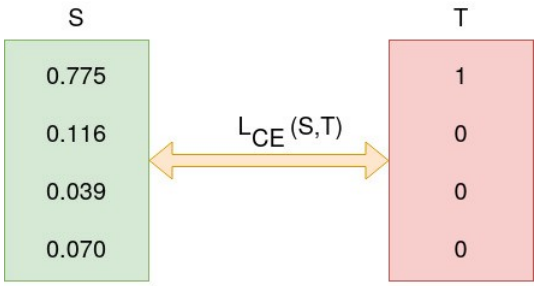


Figura 4.4: https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e

- BackPropagation:

  En el proceso de entrenamiento de la *CNN* es necesario actualizar los pesos de las capas ocultas con el objetivo de minimizar la cross-entropy total del modelo. Para minimizar este valor se utiliza el método de *Descenso por Gradiente* o *Gradient Descent (GD)*, algoritmo de optimización que maximiza el valor de una función siempre y cuando ésta sea continua diferenciable.

  Gracias a las propiedades de la función *Softmax* es posible utilizar *GD* para calcular la derivada de la función de pérdida respecto de cada uno de los pesos de las capas intermedias de la red y actualizar cada uno de ellos con el objetivo de minimizar la función de pérdida.

- Batch Normalization:

  El proceso de entrenamiento de una red neuronal puede ser costoso, ya que existen una gran cantidad de parámetros para cada capa que son optimizados durante el proceso de entrenamiento. Esto hace que este proceso sea demasiado sensible a los parámetros inicializados en el inicio del entrenamiento de la red, como es el ejemplo de establecer una tasa de aprendizaje *learning rate* muy baja para poder converger. Esto hace que el entrenamiento sea muy lento, por lo que es necesario transformar los datos de entrada a la entrada de la capa, o lo que es lo mismo, normalizar las entradas para cada mini-batch (subconjuntos de datos de entrenamiento con el que se entrena la red en cada época *epoch*).

  El método *Batch Normalization* **?** normaliza los datos de tal forma que permite utilizar tasas de aprendizaje mayores además de actuar como regularizador, reduciendo así el número de capas *Dropout* que existan en la arquitectura.

ESPECÍFICOS DE CNN:

- Kernels:

  Son las estructuras que se encargan de realizar la convolución. Los kernels son matrices de pesos unidimensionales $(1 \times n)$ o bidimensionales $(n \times m)$dependiendo del tipo de convolución que se esté aplicando (Convolución 1-D o Convolución 2-D), y se encargan de proyectar las posiciones de la matriz que correspondan en instante de la convolución en un mapa de características (*feature map*) que será pasado a la siguiente capa de la red.

Un kernel realiza el producto escalar de sus pesos con respecto a las posiciones de la matriz de entrada (imagen o *feature map*) que colisionen en ese momento con el kernel, generando un nuevo *feature map* de menor dimensión, a menos que se utilice la técnica de *Padding* **?**.

- Filtros:

  Un filtro no es más que una agrupación de kernels, siempre tendrán una dimensión mayor que la definida para el kernel. Por ejemplo, si disponemos de un kernel unidimensional $(1 \times n)$, el filtro contendrá k kernels de $(1 \times n)$ en la capa convolucional definida **?**

- Strides:

  Son los saltos producidos por un kernel en cada etapa de la convolución, dependiendo si la convolución es dimensional o bidimensional, el stride constará de una dimensión (i), es decir, los pasos hacia la derecha en la imagen, o de dos dimensiones (i, j), los pasos hacia la derecha y hacia abajo en la imagen.

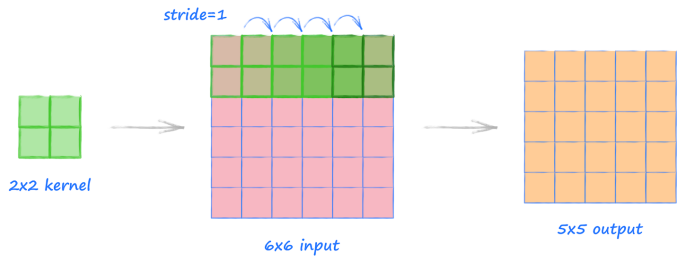


Figura 4.5: http://makeyourownneuralnetwork.blogspot.com/2020/02/calculating-output-size-of-convolutions.html

- Padding:

  El padding es un método utilizado para incrementar la dimensionalidad de la entrada de la capa debido el desvanecimiento de los valores posicionados en zonas comprometedoras una vez aplicada la convolución. Estas posiciones son aquellas en las que el filtro de la convolución pasaría una única vez en caso de no aplicar *padding*.

  El *padding* es el proceso de generar celdas artificiales inicializadas con valor *0* que permiten que el filtro convolucional sea aplicado en su totalidad en la posición de estas zonas comprometedoras, manteniendo la información de los límites de la imagen. De otra forma los valores de estos extremos se desvanecerían a medida se incrementa la profundidad de la red (número de capas) con respecto al resto de valores de la matriz.

En el proceso de entrenamiento de la *CNN* cada uno de los valores de los kernels son optimizados mediante el método *GD*
!!!!!?!!!!!
Una vez definidos los pasos que sigue el entrenamiento de una *NN*, podemos hacer distinción entre dos enfoques distintos que se han aplicado en este proyecto:

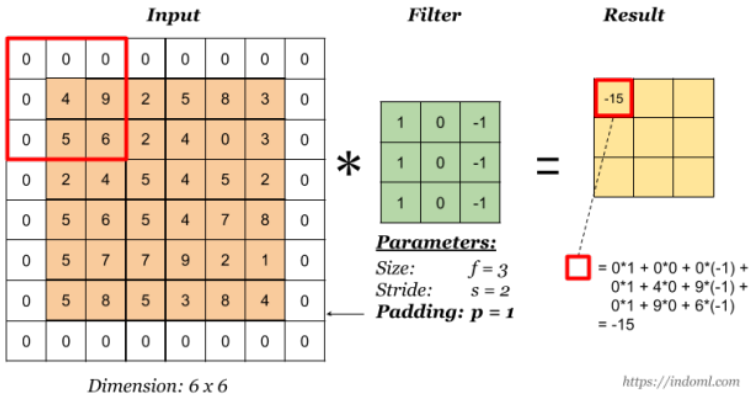


Figura 4.6: https://indoml.com/2018/03/07/student-notes-convolutional-neural-networks-cnn-introduction/

4.1.3.1. Unidimensionales (Conv-1D)

Una *Red Neuronal Convolucional de 1 Dimensión*, o *Convolutional Neural Network 1-Dimensional (CNN-1D)*, es un tipo de red convolucional que utiliza kernels *(filters)* que son aplicados a las imágenes de entrada con el objetivo encontrar ciertas características en ellas.

Las complejidad computacional de las *CNN-1D* es mucho más simple que las *CNN-2D*, además, estas redes se han demostrado ser más efectivas en distintos campos con respecto a las *CNN-2D*, como en técnicas de análisis de señales. La baja carga computacional de esta arquitectura la hace atractiva para aplicarla en tiempo real en dispositivos con pocos recursos como teléfonos móviles **?**.

El resultado de este proceso es la proyección del filtro sobre un espacio dimensional denominado mapa de caracerísticas *(feature maps)*. Este filtro se utiliza para convolucionar los feature maps de la capa anterior **?** o de la imagen de entrada si se trata de la primera capa de la red.

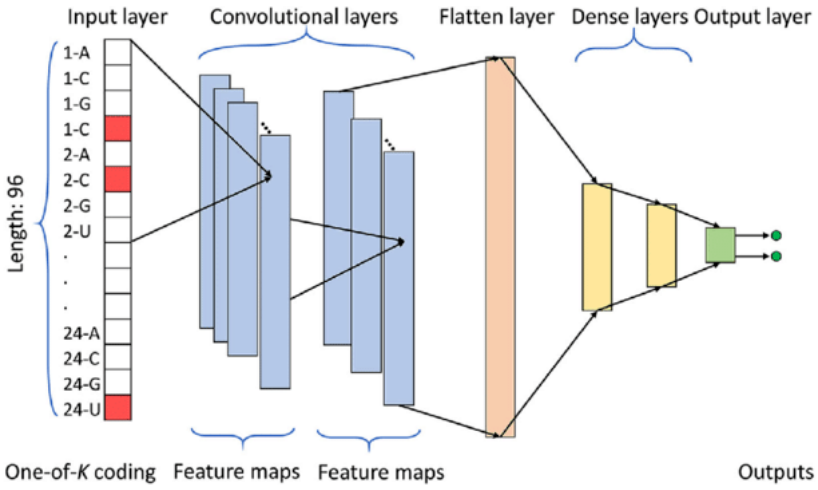


Figura 4.7: https://www.researchgate.net/publication/329201018/figure/fig2/AS:697400008138753@1543284527161/Th architecture-of-our-1D-CNN-model-This-model-consists-of-two-convolutional-layers.png

Al ser una arquitectura unidimensional, el *kernel* tendrá que tener un tamaño $(1 \times n)$.

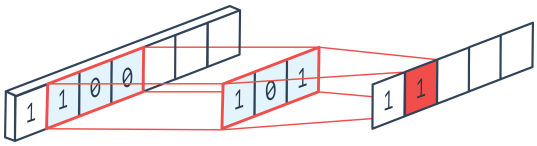


Figura 4.8: https://peltarion.com/knowledge-center/documentation/modeling-view/build-an-ai-model/blocks/1d-convolution Observamos la entrada de la red (izquierda) a la que se le aplica un kernel (en el centro) que resulta en la convolución de un mapa de características (derecha).

4.1.3.2. Bidimensionales (Conv-2D)

Otra de las técnicas convolucionales utizadas en este documento son las *Redes Neuronales Convolucionales de 2 Dimensiones*, o *Convolutional Neural Networks 2-Dimensional (CNN-2D)*. Al igual que en el caso de las *CNN-1D*, el filtro se aplica sobre el input de la capa, resultando en *feature maps*, sin embargo en este caso *kernel* será de bidimensional por ser una convolución de dos dimensiones, es necesario especificar el tamaño de la matriz que recorrerá la matriz pasada como input a la capa correspondiente.

4.1.4. KNN

El algoritmo *K vecinos más próximos*, o *K-Nearest Neighbors (kNN)* **?**, es una técnica de aprendizaje supervisado ampliamente utilizada para tareas de clasificación y regresión, es uno de los algoritmos más básicos de *ML* y tiene una gran importancia en el campo de la *Ciencia de Datos* debido a su simplicidad de implementación y a su interpretabilidad.

El algoritmo *KNN* asume que muestras con características parecidas deben pertenecer a la misma clase. O lo que es lo mismo, muestras parecidas deben estar cercanas en el espacio dimensional.

Esta técnica se basa en representar las muestras de un conjunto de datos en base a sus características en un espacio n-dimensional y clasificar la muestra correspondiente como la clase mayoritaria al calcular la distancia de dicha muestra a los k vecinos más próximos. Normalmente la distancia empleada es la *Euclídea*, aunque es posible calcular otras distancias como *Manhattan* o *Chebyshev*.

A medida que se incrementan las dimensiones del espacio resulta más costoso computacionalmente calcular distancias entre los puntos.

Por lo tanto, el único parámetro a configurar es el número de vecinos más cercanos que se calculará en base al punto que queramos clasificar, en la figura 4.9 se muestra un ejemplo de un problema de clasificacíón aplicando distintos valores al parámetros k.

4.2. Especificaciones técnicas

En esta sección detallaremos las herramientas que han sido utilizadas para poder realizar este proyecto.

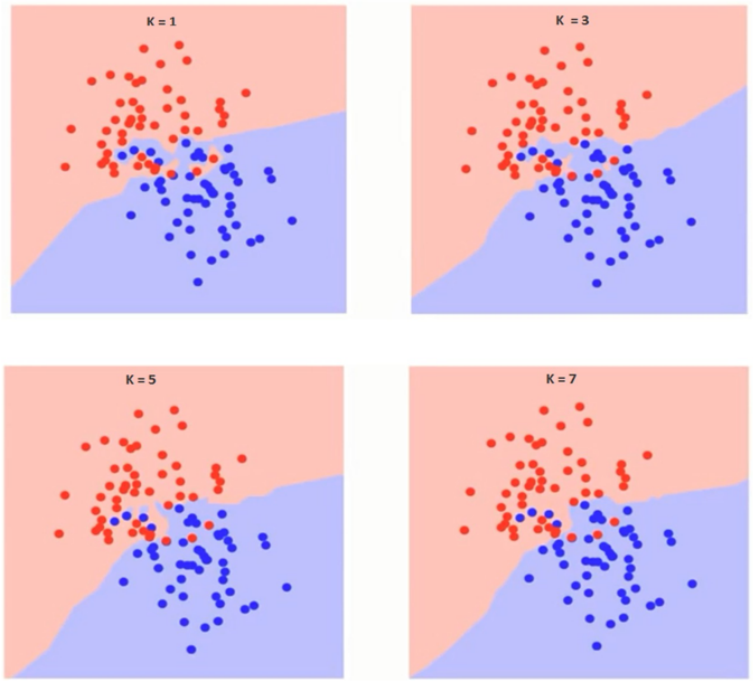


Figura 4.9: https://forum.huawei.com/enterprise/es/data/attachment/forum/202108/20/150033de3l4nyvmcbozh14.png KNN aplicado con distintos valores de k

4.2.1. Herramientas utilizadas

Para el desarrollo de este proyecto se ha hecho uso de los siguientes programas:

- Lenguaje de programación:

  Python: lenguaje de programación interpretado de nivel alto multiparadigma. La versión empleada para el desarrollo del proyecto ha sido 3.9.11.

- Liberías: Se enumerarán las librerías

  Pandas: libería que provee de herramientaas que permiten el análisis y la manipulación de datos. La versión utilizada para este proyecto es la 1.3.5 **?**.

  Tensorflow: libería utilizada para la implementación de redes neuronales permitiendo su ejecución. Este proyecto se basa en la versión 2.8.0 **?**.

  Sklearn: librería que contiene múltiples modelos predictivos implementados, basado en NumPy, SciPy y Matplotlib. La versión configurada para este proyecto es 1.0.2 **?**.

  XGboost: paquete que contiene la implementación del algoritmo XGBoost, además de múltiples configuraciones como la ejecución en *CPU, GPU y GPU paralelizada*. La importación de esta librería ha sido en base a la versión 1.5.0 **?**.

- Software:

  CUDA: plataforma de computación paralelizada que permite la ejecución de código en *GPU*, esto permite que las redes neuronales puedan entrenarse con mayor rapidez

que en *CPU* debido a la velocidad con la que se realizan las operaciones orientadas a datos en las tarjetas gráficas. Se ha utilizado la versión 11.6 **?**.

Anaconda: distribución open-source que ofrece la flexibilidad de mantener varios entornos con distintas configuraciones y versiones de distinas librerías, facilitando además la migración entre equipos. La versión utilizada ha sido la 4.12.0 **?**.

Jupyter Notebook: entorno interactivo que permite la creación, edición y ejecución de notebooks de forma local y remota. La versión utilizada para el desarrollo de este proyecto ha sido la 6.4.10 **?**.

Jupyter Lab: interfaz de nueva generación que convive con el entorno Jupyter Notebook, ofrece numerosas funcionalidades como es la navegación entre distintos repositorios dentro de la interfaz. La versión instalada para la realización del proyecto ha sido la 3.3.2 **?**.

MiKTeX:

DiagramsNet: plataforma utilizada para la confección de figuras mostradas en este documento **?**.

Google Meets: plataforma utilizada para realizar reuniones semanales con el tutor **?**.

- Sistemas de control de versiones:

  Github: repositorio donde tener un control de las versiones del desarrollo **?**.

4.2.2. Especificaciones del servidor

Los experimentos de este artículo se han realizado bajo un servidor con CPU *Dual AMD Rome 7742* (128 cores) y contando con una GPU *DGX NVIDIA A100* de 40 GigaBytes (*GB*).

5. Metodología (Con ejemplos de figuras)

El desarrollo de este proyecto respeta tanto el proceso de desarrollo *Software* como el ciclo de vida de proyectos de *Ciencia de Datos*, por lo que se ha dividido en varias fases acontecidas que definiremos a continuación en base a la planificación mediante un *Diagrama de Gantt*:

5.1. Diagrama de flujo

5.2. Inserción de figuras

5.3. Diagrama de Gantt

Las figuras son un caso un poco especial ya que LaTeX busca el mejor lugar para ponerlas, no siendo necesariamente el lugar donde está la referencia. Por ello es importante añadirle un "caption" y un "label" para poder hacer referencia a ellas en el párrafo correspondiente. Nosotros ponemos la referencia a la figura 5.1 que está en la página 30, justo aquí debajo, pero LaTeX puede que la ubique en otro lugar. (observa el código LaTeX de este párrafo para observar como se realizan las referencias. Estos detalles también se aplican a tablas y otros objetos).

Existe también la posibilidad de realizarlo sin tablas, con subfiguras:

```
\begin{figure}[h]
    \centering
    \begin{subfigure}[b]{0.4\textwidth} % Espacio horizontal ocupado por la subfigura
        \centering
        \includegraphics[width=4cm]{archivos/subs-sin} % Tamaño de la imagen
        \caption{Sin procesado.}
        \label{fig:gull}
    \end{subfigure}
    ~ % Añadir el espacio deseado, si se deja la linea en blanco la siguiente subfigura ira en una nueva linea
    \begin{subfigure}[b]{0.4\textwidth} % Espacio horizontal ocupado por la subfigura
        \centering
        \includegraphics[width=4cm]{archivos/subs-con} % Tamaño de la imagen
        \caption{Con procesado.}
        \label{fig:tiger}
    \end{subfigure}
    \caption{Ejemplo de subfiguras}\label{sistemass}
\end{figure}
```

Si eliminas la línea '\caption' de las subfiguras, tendrás las imágenes sin la información individual, aunque sí con la principal. Y obviamente, si eliminas el de la figura no se mostrará ninguna información.

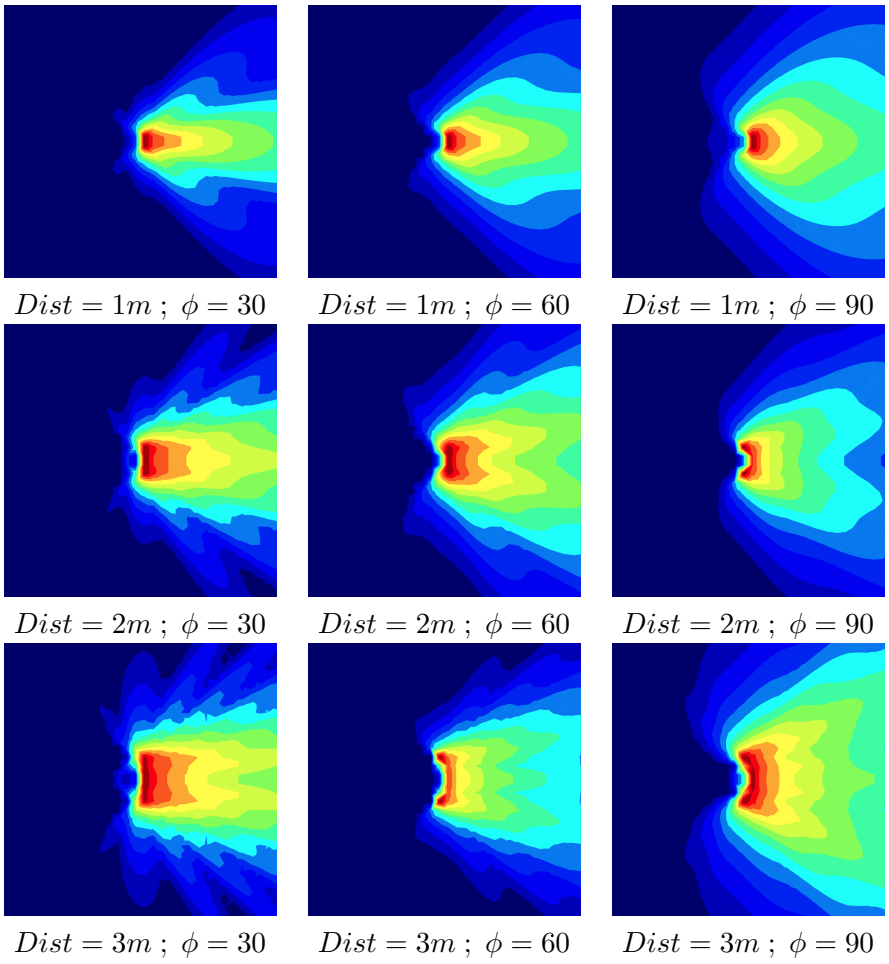


Tabla 5.1: Esta es una tabla con múltiples imágenes. Útil cuando se deben mostrar varias juntas.

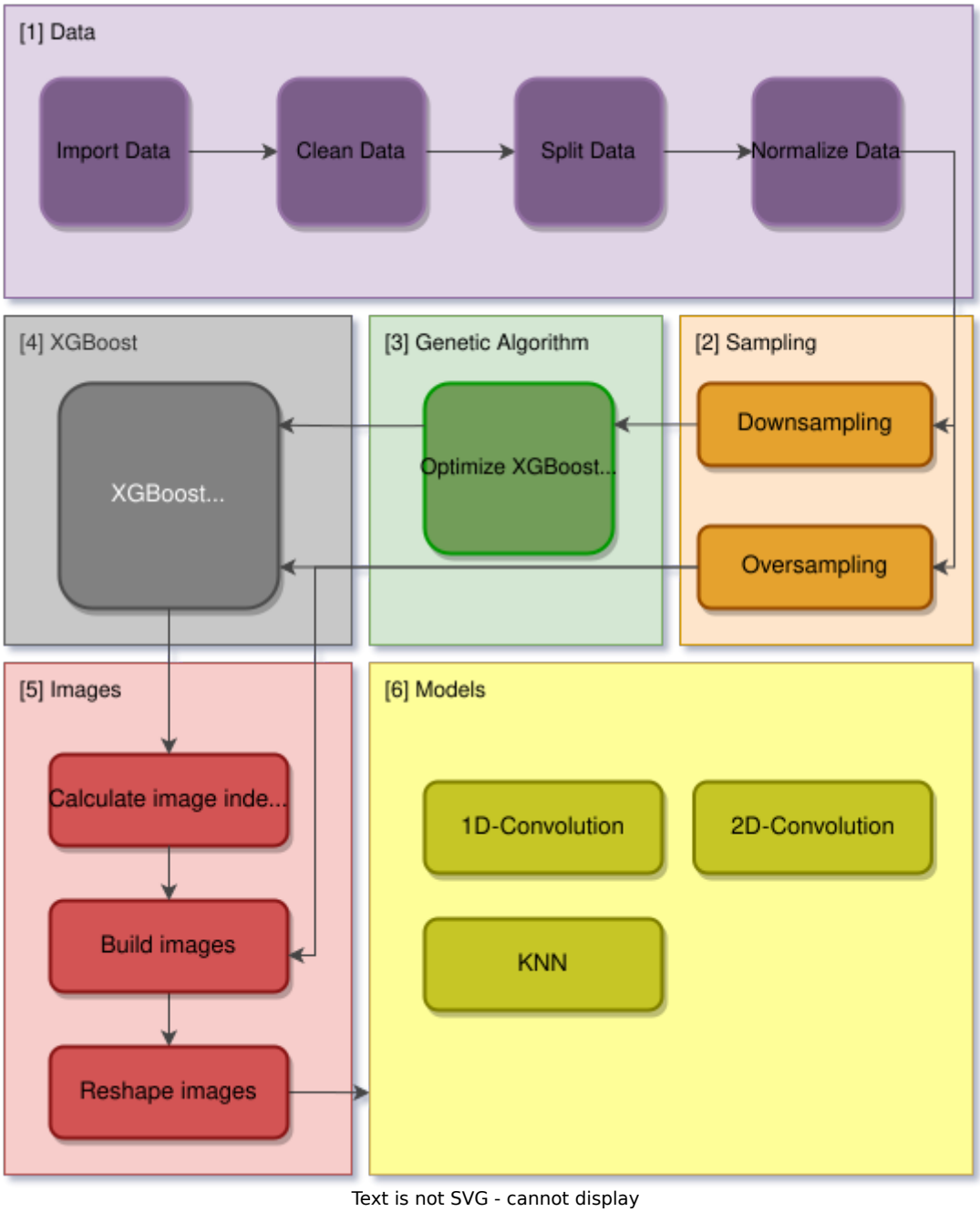


Figura 5.1: Flujo de datos del proceso del proyecto.

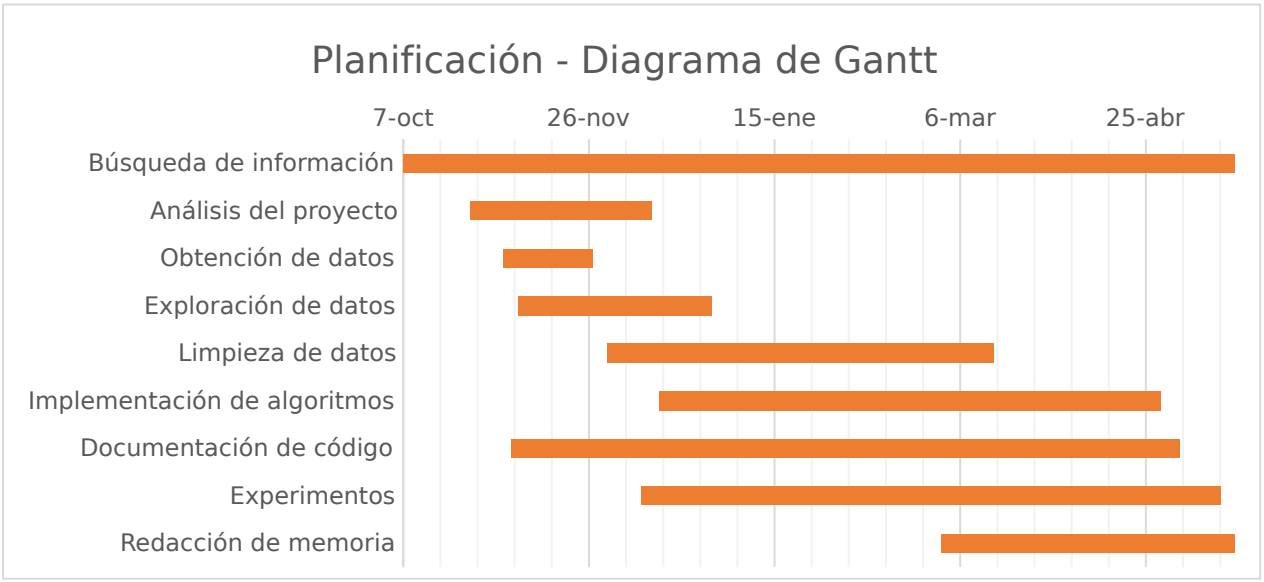
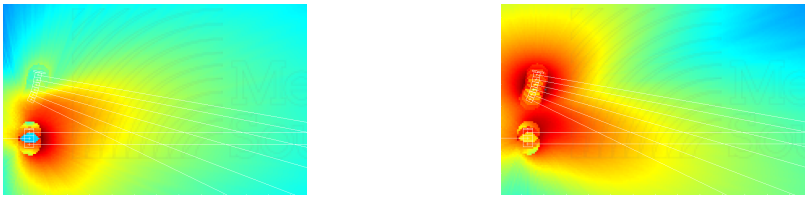
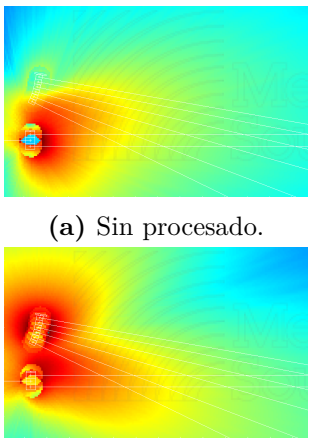


Figura 5.2: Diagrama de Gantt de la planificación del proyecto.



(a) Sin procesado. **(b)** Con procesado.

Figura 5.3: Ejemplo de subfiguras



(a) Sin procesado.

(b) Con procesado.

Figura 5.4: Ejemplo de subfiguras vertical

6. Desarrollo (Con ejemplos de código)

6.1. Inserción de código

A veces tendrás que insertar algún pedazo de código fuente para explicar algo relacionado con él. No sustituyas explicaciones con códigos enormes. Si pones algo de código en tu TFG que sea para demostrar algo o explicar alguna solución.

LaTeX te ayuda a escribir código de manera que su presentación tenga las marcas y tabulaciones propias de este tipo de texto. Para ello, debes poner el código que escribas DENTRO de un entorno que se llama "listings". La plantilla ya tiene una serie de instrucciones para incluir el paquete "listings" y añadirle algunos modificadores por lo que no tienes que incluirlo tú. Simplemente, mete tu código en el entorno "lstlisting" y ya está. Puedes indicar el lenguaje en el que está escrito el código y así LaTeX lo mostrará mejor.

En el archivo *estiloscodigoprogramacion.tex* están definidos algunos lenguajes para mostrarlos con un diseño concreto, se pueden modificar para cambiar el coloreado del código, qué términos se ponen en negrita, etc. Si se quiere profundizar más en la función "listings" se puede consultar su manual en `http://osl.ugr.es/CTAN/macros/latex/contrib/listings/listings.pdf`, aunque hay mucha información en foros y blog's que es más fácil de comprender.

Veamos un ejemplo en la figura 6.1:

```
\begin{lstlisting}[style=C, caption={ejemplo código C},label=C_code]
    #include <stdio.h>
    int main(int argc, char* argv[]) {
    puts("Hola mundo!");
    }
\ end{lstlisting}
```

El resultado será:

Código 6.1: ejemplo código C

```
1 #include <stdio.h>
2 // Comentario
3 int main(int argc, char* argv[]) {
4   puts("Hola mundo!");
5 }
```

Si lo quieres en color, está definido el estilo C-color en el archivo *estiloscodigoprogramacion.tex*, con algunos parámetros para mejorar la visualización:

```
\begin{lstlisting}[style=C-color, caption={ejemplo código C en color},label=C_code-color]
#include <stdio.h>
// Comentario
int main(int argc, char* argv[]) {
```

```
  puts(”Hola mundo!”);
}
\ end{lstlisting}
```

Código 6.2: ejemplo código C en color

```
1   #include <stdio.h>
2   // Comentario
3   int main(int argc, char* argv[]) {
4   puts(”Hola mundo!”);
5   }
```

Por supuesto, puedes mejorar esta presentación utilizando más modificadores. En la sección 6.2 se indican algunos detalles.

Otro ejemplo, ahora para mostrar código PHP, sería escribir en tu fichero LaTeX lo siguiente:

```
 \begin{lstlisting}[style=PHP, caption={ejemplo código PHP},label=PHP_code]
 /*
Ejemplo de código en PHP para escribir tu primer programa en este lenguaje
Copia este código en tu ordenador y ejecútalo
*/
<html>
  <head>
    <title>Prueba de PHP</title>
  </head>
  <body>
    <?php echo '<p>Hola Mundo</p>'; ?> //esto lo escribe TODO el mundo
  </body>
</html>
 \ end{lstlisting}
```

y el resultado es el siguiente:

Código 6.3: ejemplo código PHP

```
100 /*
101 Ejemplo de código en PHP para escribir tu primer programa en este lenguaje. Copia este código en tu ↩
        ↪ ordenador y ejecútalo
102 */
103  <html>
104   <head>
105     <title>Prueba de PHP</title>
106   </head>
107   <body>
108     <?php echo '<p>Hola Mundo</p>'; ?> //esto lo escribe TODO el mundo
109   </body>
110 </html>
```

O también en color:

Código 6.4: ejemplo código PHP

```
1 /*
2 Ejemplo de código en PHP para escribir tu primer programa en este lenguaje. Copia este código en tu ↩
        ↪ ordenador y ejecútalo
3 */
4  <html>
5   <head>
6     <title>Prueba de PHP</title>
```

```
  </head>
  <body>
    <?php echo '<p>Hola Mundo</p>'; ?> //esto lo escribe TODO el mundo
  </body>
</html>
```

Observa cómo LaTeX ha puesto los comentarios en gris y ajustado el código para que se muestre más claro.

A continuación se muestran otros ejemplos:

Código 6.5: ejemplo código Matlab en color

```
%% Code sections are highlighted.
% System command are supported...
!touch testFile.txt
A = [1, 2, 3;... %... as is line continuation.
     4, 5, 6];
fid = fopen('testFile.text', 'w');
for k=1:10
  fprintf(fid, '%6.2f \n', k)
end
x=1; %% this is just a comment, not the start of a section
% Context-sensitive keywords get highlighted correctly...
p = properties(person); %(here, properties is a function)
x = linspace(0,1,101);
y = x(end:-1:1);
% ... even in nonsensical code.
]end()()(((end while { end )end ))))end (end
%{
    block comments are supported
%} even
runaway block comments are
```

Código 6.6: ejemplo código Matlab en blanco y negro

```
%% Code sections are highlighted.
% System command are supported...
!touch testFile.txt
A = [1, 2, 3;... %... as is line continuation.
     4, 5, 6];
fid = fopen('testFile.text', 'w');
for k=1:10
  fprintf(fid, '%6.2f \n', k)
end
x=1; %% this is just a comment, not the start of a section
% Context-sensitive keywords get highlighted correctly...
p = properties(person); %(here, properties is a function)
x = linspace(0,1,101);
y = x(end:-1:1);
% ... even in nonsensical code.
]end()()(((end while { end )end ))))end (end
%{
    block comments are supported
%} even
runaway block comments are
```

Código 6.7: ejemplo código Python en color

```
class Example (object):
    def ___init___ (self, account, password):
        """e.g. account = 'bob@example.com/test'
                password = 'bigbob'
        """

        reg = telepathy.client.ManagerRegistry()
        reg.LoadManagers()

        # get the gabble Connection Manager
        self.cm = cm = reg.GetManager('gabble')

        # get the parameters required to make a Jabber connection
        # begin ex.basics.dbus.language-bindings.python.methods.call
        cm[CONNECTION_MANAGER].RequestConnection('jabber',
            {
                'account': account,
                'password': password,
            },
            reply_handler = self.request_connection_cb,
            error_handler = self.error_cb)
        # end ex.basics.dbus.language-bindings.python.methods.call
```

Código 6.8: ejemplo código Python en blanco y negro

```
class Example (object):
    def ___init___ (self, account, password):
        """e.g. account = 'bob@example.com/test'
                password = 'bigbob'
        """

        reg = telepathy.client.ManagerRegistry()
        reg.LoadManagers()

        # get the gabble Connection Manager
        self.cm = cm = reg.GetManager('gabble')

        # get the parameters required to make a Jabber connection
        # begin ex.basics.dbus.language-bindings.python.methods.call
        cm[CONNECTION_MANAGER].RequestConnection('jabber',
            {
                'account': account,
                'password': password,
            },
            reply_handler = self.request_connection_cb,
            error_handler = self.error_cb)
        # end ex.basics.dbus.language-bindings.python.methods.call
```

6.2. Usos y personalización

El texto que acompaña al código puedes incluirlo o no, también puedes decidir si el texto va numerado o no. A continuación se muestra como:

```
% Con esta línea el código no tendrá título
\begin{lstlisting}[style=Python]
    micodigo
\ end{lstlisting}
```

```
1    micodigo
```

```
    % Con esta línea el código tendrá el título abajo
    \begin{lstlisting}[style=Python, caption={Ejemplo de título abajo},captionpos=b]
        micodigo
    \ end{lstlisting}
```

```
1    micodigo
```

Código 6.9: Ejemplo de título abajo

```
    % Con esta línea el código tendrá título no numerado
    \begin{lstlisting}[style=Python, title={Ejemplo de título no numerado}]
        micodigo
    \ end{lstlisting}
```

Ejemplo de título no numerado

```
1    micodigo
```

```
    % Con esta línea el código no tendrá las líneas numeradas
\begin{lstlisting}[style=Python,numbers=none, title={Ejemplo de código sin número de líneas}]
    micodigo
    sin
    número
    de
    líneas
\ end{lstlisting}
```

Ejemplo de código sin número de líneas

```
micodigo
sin
número
de
líneas
```

6.3. Importar archivos fuente

Existe la posibilidad de importar un archivo de código en lugar de copiar su contenido y pegarlo en LaTeX.

Para realizarlo debes escribir:

```
\lstinputlisting[style=C++-color,caption={Archivo C++ importado}]{archivos/ejemplos/holamundo.cpp}
```

Y se importará con el formato establecido entre los '[ ]':

Código 6.10: Archivo C++ importado

```
#include <stdio.h>
int main()
{
   // printf() displays the string inside quotation
   printf("Hello, World!");
   return 0;
}
```

A continuación se muestran otros ejemplos

```
\lstinputlisting[style=Python-color,caption={Archivo Py importado},label=importado_py]{archivos/↩
    ↪ ejemplos/holamundo.py}
```

Código 6.11: Archivo Py importado

```
#------------------------------------- ↩
    ↪ -------------------------------------
# helloworld.py
#------------------------------------- ↩
    ↪ -------------------------------------

import stdio

# Write 'Hello, World' to standard output.
stdio.writeln('Hello, World')

#------------------------------------- ↩
    ↪ -------------------------------------

# python helloworld.py
# Hello, World
```

```
\lstinputlisting[style=Matlab-color,caption={Archivo Matlab importado},label=importado_m]{archivos/↩
    ↪ ejemplos/holamundo.m}
```

Código 6.12: Archivo Matlab importado

```
function y = hello_world %#codegen

y = 'Hello World!';

end
% Copyright 2010 The MathWorks, Inc.
```

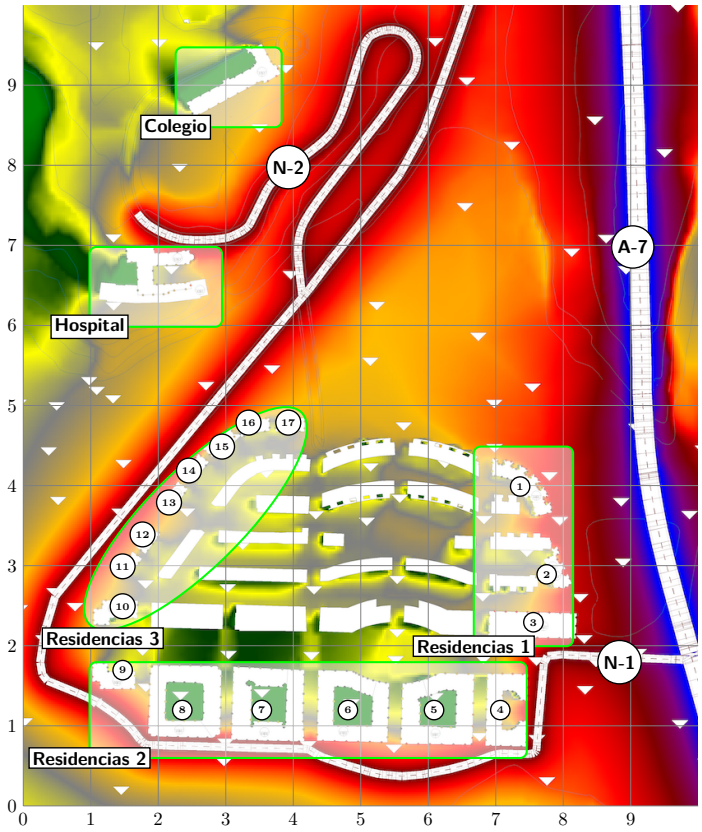
7. Resultados (Con ejemplos de gráficos)

7.1. Diagramas

Gracias al paquete *Tikz* se pueden incluir multitud de medios gráficos, diagramas, capas sobre imágenes, etc. Existen múltiples formas de realizarlo, para ello es recomendable consultar la guía de iniciación disponible aquí: `http://cremeronline.com/LaTeX/minimaltikz.pdf` y también el manual completo disponible aquí: `http://osl.ugr.es/CTAN/graphics/pgf/base/doc/pgfmanual.pdf`.

A continuación se muestran algunos ejemplos. Revisa el archivo .tex para ver cómo se utilizan.

Imagen a la que se le ha añadido cuadros y texto desde latex:



En muchas ocasiones es necesario realizar un diagrama de bloques, más abajo se muestra

un ejemplo de ello. En la red hay multitud de ejemplos que pueden ser fácilmente modificables para un fin concreto, como por ejemplo en esta web: `http://www.texample.net/tikz/examples/tag/block-diagrams/`.

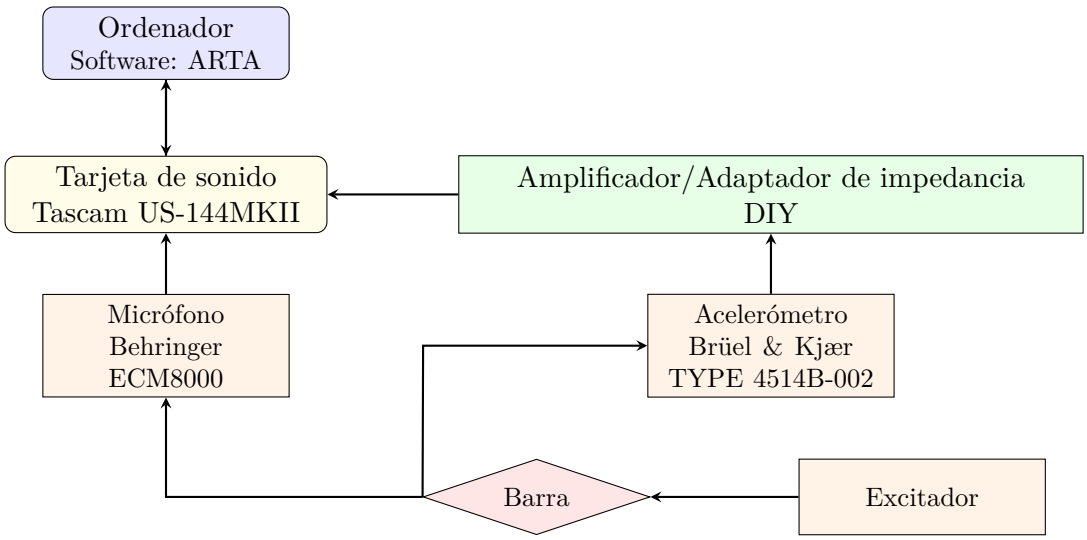


Figura 7.1: Diagrama realizado en latex con Tikz.

7.2. Gráficas

Existen múltiples formas de generar gráficas para latex. Hay disponibles herramientas como GeoGebra que dispone de la utilidad para exportar los gráficos en formato Tkiz. También funciones para Matlab que genera las gráficas que muestra habitualmente pero en código para Tkiz.

7.2.1. Línea

La forma más simple, aunque no sencilla cuando abarca muchos datos es la siguiente:

```
\begin{figure}[ht]
\centering
    \begin{tikzpicture}
        \begin{axis}
            [ymin=0,ymax=5, % Límites del eje y
            xmin=0,xmax=6, % Límites del eje x
            ylabel= eje Y,  % Nombre del eje y
            xlabel= eje X] % Nombre del eje x
            \addplot+[smooth] coordinates % Une los puntos curva suavizada
            {(0,0) (1,2) (2,3 (4,3))}; % Puntos de la gráfica
        \end{axis}
    \end{tikzpicture}
\caption{Gráfica sencilla.}
\end{figure}
```

El resultado es el siguiente:

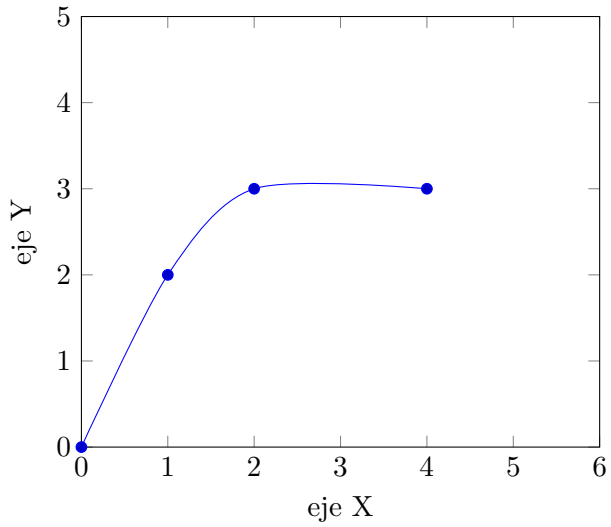


Figura 7.2: Gráfica sencilla.

Otro ejemplo, en este caso las lineas están calculadas directamente en LaTex y después cada una tiene una anotación (el código se encuentra en el archivo archivos/ejemplos/perjudicialesopticacentro.tex):

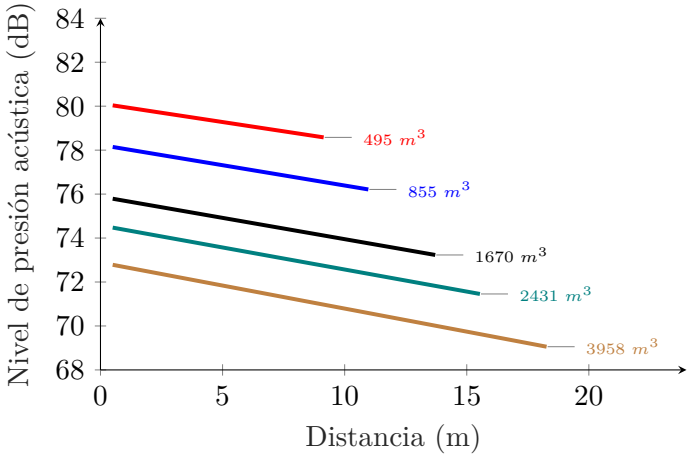


Figura 7.3: OP/S003

7.2.2. Barras

Otro ejemplo es la gráfica de barras:

```
\begin{figure}[ht]
\centering
\begin{tikzpicture}
    \begin{axis}[
        ybar=12pt,
```

```
        ymin=0,ymax=150,
        xtick=data,
        enlarge x limits={abs=2cm},
        symbolic x coords={rubio, moreno},
        bar width = 20pt,
        ylabel= número,
        xlabel= color de pelo,
            ytick align=outside,
            ytick pos=left,
            major x tick style = transparent,
            legend style={at={(0.04,0.96)},anchor=north west, font=\footnotesize, legend cell align=left,},
            ]
        \addplot[ybar,fill=blue, area legend] coordinates {
            (rubio,20)
            (moreno,100)};
        \addplot[ybar,fill=purple, area legend] coordinates {
            (rubio,110)
            (moreno,105)};
    \legend{Chicos, Chicas}
    \end{axis}
\end{tikzpicture}
\caption{Gráfica barras.}
\end{figure}
```

El resultado es el siguiente:

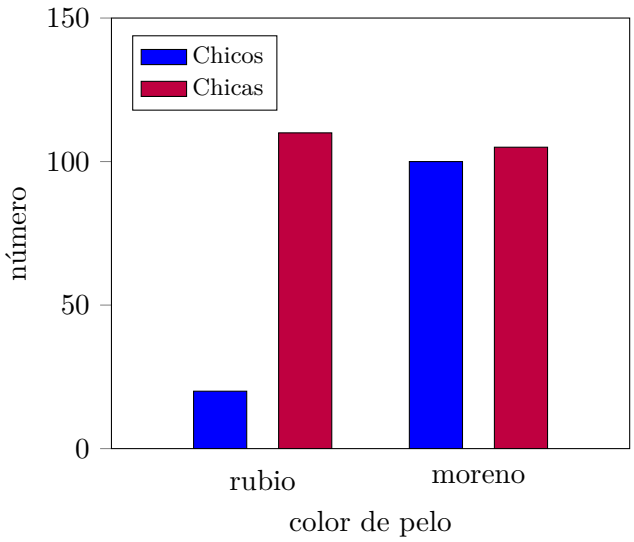


Figura 7.4: Gráfica barras.

7.2.3. Polar

Un ejemplo de gráfica polar semicircular (ver archivo archivos/ejemplos/polarnorm.tex):

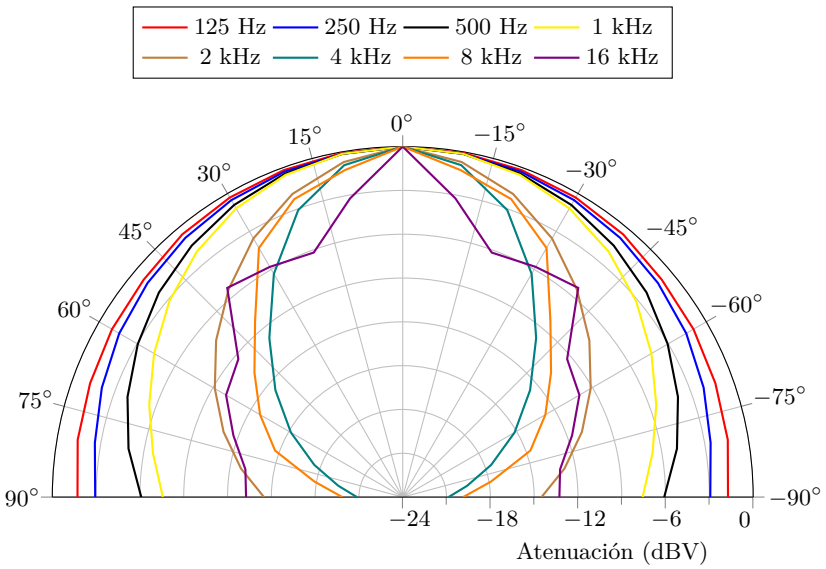


Figura 7.5: Directividad normalizada del altavoz (0 dBV en el eje).

7.3. Importados de MATLAB

Gracias a la herramienta *matlab2tikz* (`https://es.mathworks.com/matlabcentral/fileexchange/22022-matlab2tikz-matlab2tikz`) se pueden exportar las gráficas de cualquier tipo de Matlab a latex. Después de incluir los archivos de *matlab2tikz* se debe escribir una llamada después de crear la figura tal que:

Código 7.1: Ejemplo de llamada a matlab2tikz

```
fig = plot(x,y);
matlab2tikz('figurehandle',fig,'NombreArchivo.tex','height','5cm','width','13.5cm','strict',true,'↩
    ↪ showHiddenStrings',true,'showInfo',false)
```

Y para utilizar el archivo generado por la función en este documento:

```
\begin{figure}[ht]
    \centering
    {\scalefont{0.8}\input{archivos/ejemplos/ParedFina} }
    \caption{Ejemplo de gráfica obtenida con matlab2tikz.}
\end{figure}
```

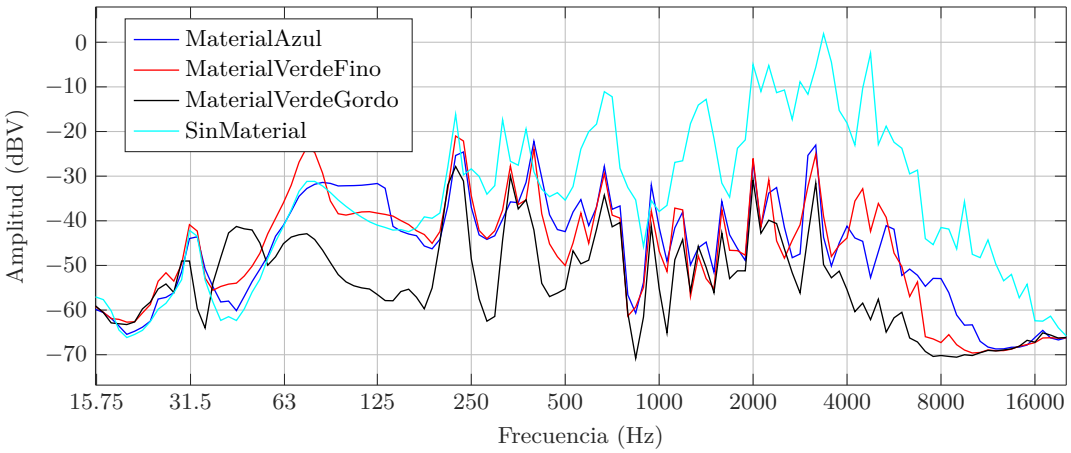


Figura 7.6: Ejemplo de gráfica obtenida con matlab2tikz.

Ejemplo de una gráfica 3D generada en Matlab y exportada por *matlab2tikz*:

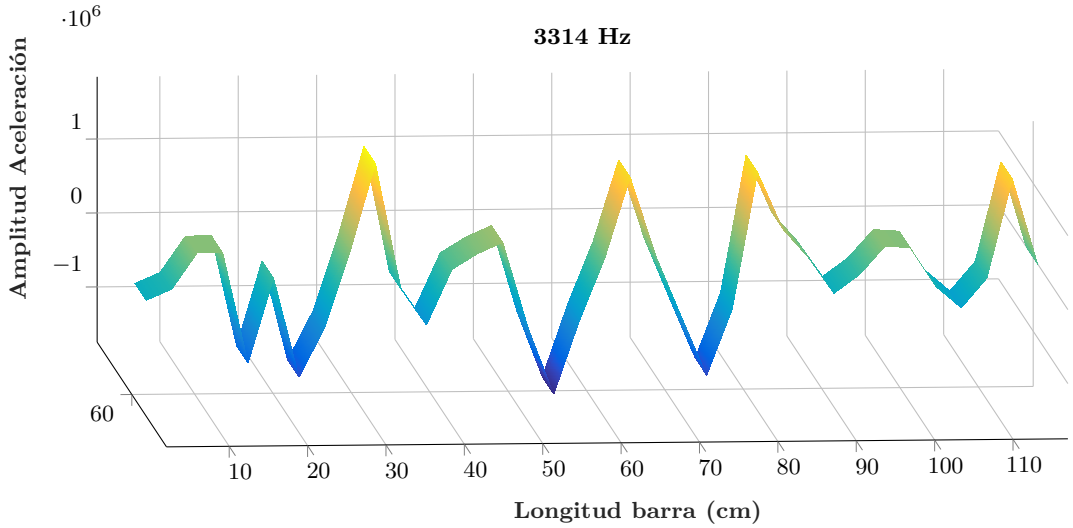


Figura 7.7: Amplitud de la aceleración en el modo número 8.

7.4. Ejemplo avanzado

El potencial del paquete *Tikz* es muy alto, se pueden realizar muchísimas cosas. En la red se facilitan muchos ejemplos para poder ver el funcionamiento y aprender. Existen hilos donde la gente publica sus mejores diseños de *Tikz* como en `https://tex.stackexchange.com/questions/158668/nice-scientific-pictures-show-off` o páginas donde facilitan muchas plantillas como `http://www.texample.net/tikz/examples/all/`.

Un ejemplo de lo que se puede llegar a conseguir es el siguiente:

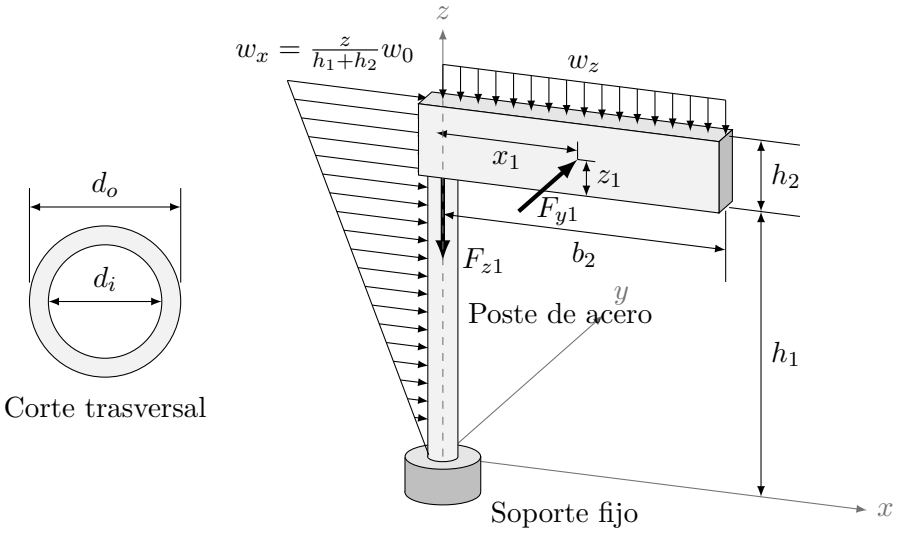


Figura 7.8: Señal realizada con Tikz, sin imágenes.

8. Conclusiones (Con ejemplos de matemáticas)

8.1. Matemáticas

En LaTeX se pueden mostrar ecuaciones de varias formas, cada una de ellas para un fin concreto.

Antes de ver algunas de estas formas hay que conocer cómo se escriben fórmulas matemáticas en LaTeX. Una fuente de información completa es la siguiente: `https://en.wikibooks.org/wiki/LaTeX/Mathematics`. También existen herramientas online que permiten realizar ecuaciones mediante interfaz gráfica como `http://www.hostmath.com/`, `https://www.mathcha.io/editor` o `https://www.latex4technics.com/`

Para mostrar una ecuación numerada se debe utilizar:

```
\begin{equation}
    \nabla\times{\mathbf H}=\left[\frac{1}{r}\frac{\partial}{\partial r}(rH_\theta)-\frac{1}{r}\frac{↩
        ↪ \partial H_r}{\partial\theta}\right]{\hat{\mathbf z}}
    \label{ecuacion}
\end{equation}
```

$$\nabla \times \mathbf{H} = \left[\frac{1}{r}\frac{\partial}{\partial r}(rH_\theta) - \frac{1}{r}\frac{\partial H_r}{\partial \theta}\right]\hat{\mathbf{z}} \tag{8.1}$$

Si es necesario agrupar varias ecuaciones en un mismo índice se puede escribir del siguiente modo:

```
\begin{subequations}
    \begin{eqnarray}
        {\mathbf E}&=&E_z(r,\theta)\hat{\mathbf z}\label{ecu1} \\ % Salto de línea
        {\mathbf H}&=&H_r(r,\theta))\hat{ \mathbf r}+H_\theta(r,\theta)\hat{\bm \theta}\label{ecu2}
    \end{eqnarray}
\end{subequations}
% Se incluye '&' entre la igualdad para centrar las ecuaciones desde el '='.
```

$$\begin{aligned} \mathbf{E} &= E_z(r,\theta)\hat{\mathbf{z}} && \text{(8.2a)} \\ \mathbf{H} &= H_r(r,\theta))\hat{\mathbf{r}} + H_\theta(r,\theta)\hat{\boldsymbol{\theta}} && \text{(8.2b)} \end{aligned}$$

Otras dos formas que son las habituales en muchos lugares para incluir ecuaciones son:

```
Ejemplo de fórmula en línea con el texto $\int_{a}^{b} f(x)dx = F(b) - F(a)$, esta ecuación quedará dentro ↩
    ↪ del texto.

Esta otra, al utilizar dos '\$', se generará en una línea nueva $$\int_{a}^{b} f(x)dx = F(b) - F(a)$$
```

Ejemplo de fórmula en línea con el texto $\int_a^b f(x)dx = F(b) - F(a)$, esta ecuación quedará dentro del texto.

Esta otra, al utilizar dos '$', se generará en una línea nueva

$$\int_a^b f(x)dx = F(b) - F(a)$$

También se puede añadir información adicional a una ecuación con la función *condiciones* creada para esta plantilla:

```
\begin{equation}
    \underset{z=z_0}{\mathrm{Res}}(f(z))=\frac{1}{(m-1)!}\lim_{z \rightarrow z_0}\left[\frac{\text{d↩
        ↪ }^{m-1}}{\text{d}z^{m-1}} \left[\left(z-z_0\right)^m f(z) \right] \right]
\end{equation}

\begin{condiciones}[donde:]
% Excepto 'Descripción y valor' el resto no es necesario el símbolo $para texto matemático.
% Item & Relación & Descripción o valor
    m  & \rightarrow  & Es la multiplicidad del polo $z_0$ \\
    z_0 & \rightarrow  & Es la parte que se iguala a 0 con el polo. \\
    f(z)& \rightarrow  & Es la función contenida en la integral.
\end{condiciones}
```

$$\underset{z=z_0}{\operatorname{Res}}(f(z)) = \frac{1}{(m-1)!}\lim_{z\to z_0}\left[\frac{\mathrm{d}^{m-1}}{\mathrm{d}z^{m-1}}\left[(z-z_0)^m f(z)\right]\right] \tag{8.3}$$

donde: m → Es la multiplicidad del polo z_0
z_0 → Es la parte que se iguala a 0 con el polo.
$f(z)$ → Es la función contenida en la integral.

Si lo que deseas es una ecuación alineada a la izquierda o derecha puedes hacerlo con lo siguiente (el '&' simple es utilizado para alinear las ecuaciones desde ese punto, los iguales):

```
% Alineado a la izquierda al incluir al final el doble '&&'
\begin{flalign}
    y_{h_1}=&\begin{bmatrix}6\cos(\sqrt{6} x) \\ -\sqrt{6}\sin(\sqrt{6}x)\end{bmatrix}e^x &&\\
    y_{h_2}=&\begin{bmatrix}6\sin(\sqrt{6} x) \\ \sqrt{6}cos(\sqrt{6}x)\end{bmatrix}e^x &&
\end{flalign}

% Alineado a la derecha al incluir al inicio el doble '&&'
\begin{flalign}
&& y_{h_1}=&\begin{bmatrix}6\cos(\sqrt{6} x) \\ -\sqrt{6}\sin(\sqrt{6}x)\end{bmatrix}e^x\\
&& y_{h_2}=&\begin{bmatrix}6\sin(\sqrt{6} x) \\ \sqrt{6}cos(\sqrt{6}x)\end{bmatrix}e^x
\end{flalign}
```

$$y_{h_1} = \begin{bmatrix} 6\cos(\sqrt{6}x) \\ -\sqrt{6}\sin(\sqrt{6}x) \end{bmatrix} e^x \tag{8.4}$$

$$y_{h_2} = \begin{bmatrix} 6\sin(\sqrt{6}x) \\ \sqrt{6}cos(\sqrt{6}x) \end{bmatrix} e^x \tag{8.5}$$

$$y_{h_1} = \begin{bmatrix} 6\cos(\sqrt{6}x) \\ -\sqrt{6}\sin(\sqrt{6}x) \end{bmatrix} e^x \tag{8.6}$$

$$y_{h_2} = \begin{bmatrix} 6\sin(\sqrt{6}x) \\ \sqrt{6}cos(\sqrt{6}x) \end{bmatrix} e^x \tag{8.7}$$

Tanto con la función utilizada en (8.1,8.3), como en (8.2a,8.2b) y en las anteriores, si se les incluye un '*' después de 'equation', 'subequation' o 'flalign', se elimina la numeración de las ecuaciones pero manteniendo el resto de características.

A. Anexo I

Aquí vendría el anexo I

B. Páginas horizontales

Aquí se muestra cómo incluir páginas en horizontal.
Esta página está en vertical

Esta página está en horizontal

Esta página también está en horizontal

Esta página está de nuevo en vertical

C. Importar PDF

A continuación se muestra una página importada de un PDF externo. Observar los comentarios en el código de este anexo para más información. También puedes leer el manual con todas las opciones en `http://osl.ugr.es/CTAN/macros/latex/contrib/pdfpages/pdfpages.pdf`.

Alicante 15 DE MARZO DE 2007

Expediente número

Referencia del peticionario **AYUNTAMIENTO DE ALICANTE**

Departamento de Medio Ambiente
C/San Nicolás, nº 2, 4º
03001 ALICANTE
Contacto: Juan Luís Beresaluze

DOCUMENTO DE SÍNTESIS

ELABORACIÓN DEL MAPA ACÚSTICO MUNICIPAL DE LA CIUDAD DE ALICANTE

Fecha de realización del estudio: MAYO 2005 – MARZO 2007