UNIVERSIDAD DE MÁLAGA

ESCUELA DE INGENIERÍAS INDUSTRIALES

**ESCUELA DE INGENIERÍAS INDUSTRIALES**

**LENGUAJE Y CIENCIAS DE LA COMPUTACIÓN**

# TRABAJO FIN DE GRADO

## VÍDEOS DOCENTES PARA APRENDIZAJE DE MATLAB EN INGENIERÍA INDUSTRIAL

Grado en

Ingeniería Electrónica, Robótica y Mecatrónica

Autor: MOHAMED AMINE ZEGGAF

Tutor: JOSÉ GALINDO GÓMEZ

MÁLAGA,12 Junio de 2023

VÍDEOS DOCENTES PARA APRENDIZAJE DE MATLAB
EN INGENIERÍA INDUSTRIAL

Mohamed Amine Zeggaf

# Resumen

Este proyecto tiene como propósito principal la elaboración de vídeos educativos para facilitar el aprendizaje de MATLAB desde cero. Los contenidos seguirán principalmente el temario de la asignatura Sistemas Informáticos, impartida en diversos grados de la Escuela de Ingenierías Industriales de la Universidad de Málaga, y coordinada por el profesor Dr. José Galindo Gómez, tutor del presente trabajo. Los vídeos se alojarán en YouTube, aprovechando el canal ya existente del tutor, el cual se llama “Aprende conmigo Informática”.

Los vídeos tratan desde temas básicos como el entorno de MATLAB, sus variables, matrices y operaciones aritméticas simples, hasta otros más avanzados como la resolución de sistemas de ecuaciones, cálculo simbólico, números complejos, o gráficas en 2D.

El compromiso fundamental del trabajo ha sido ofrecer contenido educativo de calidad. Para ello, se han seleccionado cuidadosamente fuentes adecuadas y se ha verificado la información antes de incorporarla. El objetivo es ayudar a los estudiantes a entender mejor los conceptos teóricos y su aplicación práctica en la ingeniería. La naturaleza audiovisual del contenido posibilitará un aprendizaje más visual e interactivo y a un ritmo personalizado, lo que puede incrementar la motivación y la retención de información. Esta metodología es especialmente útil para aquellos que encuentran dificultades en la comprensión de conceptos mediante lectura o explicaciones en clase.

Los vídeos también serán de utilidad para cualquier persona interesada en aprender MATLAB, sin importar su nivel de conocimientos previo o su localización física. Esto permite difundir y hacer accesible la enseñanza de MATLAB a una audiencia extraordinariamente amplia.

Por último, estos vídeos también son concebidos como recursos para la educación a distancia en casos de necesidad, como ocurrió con la pandemia de COVID-19. Con la posibilidad de nuevas pandemias debido a diversos factores ambientales, la existencia de recursos educativos *online* es de gran importancia.

**Palabras clave:** MATLAB, Sistemas Informáticos, Programación de ordenadores, Vídeoseducativos, Docencia en línea, YouTube, Programación científica

# Abstract

The main purpose of this project is to create educational videos to facilitate learning MATLAB from scratch. The contents will primarily follow the syllabus of the subject entitle “Sistemas Informáticos”, taught in different degrees at the School of Industrial Engineering of the University of Malaga, and coordinated by lecturer Dr. José Galindo Gómez, tutor of this work. The videos will be hosted on YouTube, leveraging the existing channel of the tutor, which is called “Aprende conmigo Informática”.

The videos cover basic topics such as the MATLAB environment, its variables, matrices, and simple arithmetic operations, until more advanced ones such as the resolution of systems of equations, symbolic calculus, complex numbers, or 2D graphics.

The main commitment of the work has been to offer quality educational content. To this end, suitable sources have been carefully selected and the information has been verified before incorporating it. The aim is to help students better understand theoretical concepts and their practical application in engineering. The audiovisual nature of the content will enable learning at one's own pace, more visual and interactive, which can increase motivation and retention of information. This methodology is especially useful for those who face difficulties with understanding concepts through reading or class explanations.

The videos will also be useful for anyone interested in learning MATLAB, regardless of their previous knowledge level or their locations. This allows the dissemination and accessibility of MATLAB teaching to an extraordinarily broader audience.

Finally, these videos are also conceived as resources for distance education in cases of need, as happened by the COVID-19 pandemic. With the possibility of new pandemics due to various environmental factors, the existence of online educational resources is of great importance.

**Keywords:** MATLAB, Computer Systems, Computer Programming, Educational videos, Online teaching, YouTube, Scientific programming.

# ÍNDICE DE CONTENIDO

# ÍNDICE DE FIGURAS

# ÍNDICE DE TABLAS

# Capítulo 1: Introducción, antecedentes y objetivos

## 1.1. Introducción

La programación resulta ser una herramienta fundamental en el mundo en el que vivimos, pues es la base de muchos avances tecnológicos. Podemos notar su impacto en casi todos los ambientes de la vida, desde la comunicación entre personas hasta la investigación científica, pasando por la industria y el entretenimiento. Algunos ejemplos donde la programación resulta ser esencial son la inteligencia artificial (**IA**), la robótica, el internet de las cosas (**IoT**) y una amplia gama de tecnologías emergentes.

Por ello, en [1] se dice que “el conjunto de recursos, procedimientos y técnicas usadas en el procesamiento, almacenamiento y transmisión de información se ha matizado de la mano de las tecnologías de la información y las comunicaciones”.

En cuanto al ámbito de la educación superior en ingeniería, la programación se ha ido haciendo cada vez más imprescindible en el proceso de formación de nuevos profesionales para los tiempos venideros. Las universidades han ido incluyendo cada vez más lenguajes de programación y cursos específicos para el aprendizaje de estos en sus planes de estudios. Esto es debido a que el desarrollo de las habilidades en la programación permite a los ingenieros ser capaces de desarrollar soluciones a problemas de gran complejidad. Entre los lenguajes más empleados en las universidades encontramos:

- **Python** [11]: Es un lenguaje de programación de alto nivel, fácil de aprender y muy versátil. Se utiliza ampliamente en la enseñanza de la programación, así como en

campos como análisis de datos, inteligencia artificial, desarrollo web y automatización.

- ❖ **SymPy** [12]: Es una **biblioteca de Python** de código abierto para matemáticas simbólicas que funciona como un sistema de álgebra computacional (**CAS**), permitiendo a los usuarios realizar cálculos simbólicos en lugar de numéricos. Es útil para resolver problemas matemáticos complejos, simplificar expresiones y realizar cálculos avanzados, como límites, derivadas e integrales. Además, se integra fácilmente con otras bibliotecas de Python para realizar cálculos numéricos cuando sea necesario.
- ❖ **NumPy** [13]: Es una **biblioteca de Python** de código abierto fundamental para la computación científica que ofrece soporte para trabajar con *arrays* y matrices multidimensionales, así como funciones matemáticas de alto nivel. Facilita la realización de cálculos numéricos eficientes, permitiendo realizar operaciones matriciales, álgebra lineal, transformadas de Fourier y más. NumPy es ampliamente utilizado en la ciencia de datos, el aprendizaje automático y la investigación científica.

- **Java** [14]: Es un lenguaje de programación orientado a objetos, robusto y multiplataforma. Es ampliamente utilizado en el desarrollo de aplicaciones empresariales, aplicaciones móviles (especialmente en **Android**) y sistemas integrados.
- **C** [15]: Es un lenguaje de programación de propósito general y de bajo nivel, que permite un control más directo sobre la memoria y el *hardware*. Es la base de muchos otros lenguajes, y se utiliza comúnmente en la enseñanza de fundamentos de programación y sistemas operativos.
- **C++** [16]: Es una **extensión de C** que incluye características de programación orientada a objetos. Es ampliamente utilizado en el desarrollo de *software* de sistemas, videojuegos, controladores de *hardware* y aplicaciones de alto rendimiento.
- **C#** [17]: Es un lenguaje de programación desarrollado por **Microsoft**, similar a Java en muchos aspectos. Se utiliza principalmente para el desarrollo de aplicaciones de Windows y aplicaciones web utilizando el marco **.NET**. Se incluye entre los lenguajes "**visuales**" porque su entorno de desarrollo incorpora mecanismos simples y gráficos para diseñar la interfaz del programa (formularios con botones, cajas de texto, gráficos, etiquetas…).

- **JavaScript** [18]: Es un lenguaje de programación de alto nivel, principalmente utilizado en el desarrollo web para agregar interactividad y funcionalidades en el lado del cliente a las páginas web. También se utiliza en el desarrollo de aplicaciones móviles y servidores con **Node.js**.

- **Ruby** [19]: Es un lenguaje de programación de alto nivel y fácil de leer, que se utiliza principalmente en el desarrollo web y en la creación de aplicaciones web con el marco **Ruby on Rails**.

- **PHP** [20]: Es un lenguaje de programación de propósito general, especialmente diseñado para el desarrollo web y la creación de aplicaciones web dinámicas. Se ejecuta en el servidor y es compatible con la mayoría de las bases de datos.

- **R** [21]: Es un lenguaje de programación y entorno de *software* para análisis estadístico y gráficos. Se utiliza ampliamente en la investigación y el análisis de datos, y es especialmente popular en la comunidad de la ciencia de datos.

- **MATLAB** [22]: Es un lenguaje de programación de alto nivel, junto con un entorno interactivo, diseñado específicamente para facilitar cálculos numéricos, visualización de datos y desarrollo de algoritmos en diversas áreas. Es propiedad de **Mathworks**. Este potente lenguaje y entorno se emplea ampliamente en campos como las matemáticas, la ingeniería y la ciencia de datos para llevar a cabo análisis de datos, resolver ecuaciones diferenciales y realizar optimizaciones. El presente trabajo se fundamentará en el entorno y lenguaje de programación MATLAB, aprovechando sus capacidades y funcionalidades para abordar problemas específicos. En capítulos posteriores, se profundizará en el uso de MATLAB, sus herramientas, y cómo se aplican a los problemas y desafíos que se abordarán en este trabajo.

- **Octave** [23]: Es un lenguaje de programación de alto nivel, similar a MATLAB, diseñado principalmente para cálculos numéricos. Es de **código abierto** y compatible con muchas funciones de MATLAB, lo que lo convierte en una **alternativa más asequible**.

- **Julia** [24]: Es un lenguaje de programación de alto nivel que ofrece un buen rendimiento para cálculos técnicos. Está diseñado para ser fácil de usar y tiene una sintaxis similar a MATLAB. Se utiliza en aplicaciones de matemáticas, estadísticas, análisis de datos y cómputo científico.

- **Mathematica** [25]: Es un *software* matemático y un lenguaje de programación simbólico desarrollado por **Wolfram Research**. Es utilizado en diversas disciplinas como matemáticas, física, química y biología para cálculos simbólicos, manipulación algebraica y visualización de datos.
- **Maple** [26]: Es un *software* de matemáticas y un lenguaje de programación simbólico que se utiliza para el análisis y solución de problemas matemáticos, incluyendo cálculo, álgebra, ecuaciones diferenciales y análisis de datos.
- **SageMath** [27]: Es un *software* matemático de código abierto que integra varios lenguajes de programación y herramientas matemáticas en un solo entorno. Es una **alternativa de código abierto a MATLAB**, **Mathematica y Maple**.
- **Fortran** [28]: Es un lenguaje de programación de alto nivel, especialmente diseñado para cálculos numéricos y científicos. Es ampliamente utilizado en aplicaciones de matemáticas, física e ingeniería para cálculos intensivos en computación y simulaciones.
- **Scilab** [29]: Es un *software* de código abierto para cálculos numéricos y un lenguaje de programación similar a MATLAB. Es utilizado en aplicaciones de matemáticas, ingeniería y ciencia de datos para el análisis y visualización de datos, resolución de ecuaciones diferenciales y optimización.
- **IDL** (**Interactive Data Language**) [30]: Es un lenguaje de programación y entorno de desarrollo utilizado principalmente para análisis y visualización de datos en ciencias e ingeniería. Es especialmente popular en la comunidad de ciencias atmosféricas y astronómicas.
- **LabVIEW** [31]: Es un entorno de desarrollo y un lenguaje de programación gráfico, desarrollado por **National Instruments**, utilizado para el control de instrumentos, adquisición de datos, análisis y presentación de datos en tiempo real en aplicaciones de ingeniería y científicas.
- **Maxima** [32]: Es un sistema de álgebra computacional y un lenguaje de programación simbólico de código abierto. Se basa en el sistema de álgebra computacional **MACSYMA** y se utiliza para manipulación algebraica, cálculo simbólico y gráficos.
- **GAP** (**Groups, Algorithms, and Programming**) [33]: Es un sistema de álgebra computacional y un lenguaje de programación diseñado para trabajar con grupos y

estructuras algebraicas discretas. Se utiliza en investigación matemática, especialmente en la teoría de grupos y combinatoria.

- **Haskell** [34]: Es un lenguaje de programación funcional de alto nivel y fuertemente tipado. Aunque no está específicamente diseñado para matemáticas, su enfoque funcional y su capacidad para abstraer y razonar sobre funciones matemáticas lo hacen popular en ciertas aplicaciones matemáticas y de investigación.

Ortiz Zambrano et al. dicen que "la principal ventaja de saber programar computadoras no está en el hecho de que se domine algún lenguaje informático, sino en las habilidades que se desarrollan al aprender a usar dichos lenguajes, por ejemplo: aprender la forma de plantear un problema, organizar la solución del problema como una secuencia lógica de pasos y formular la toma de decisiones, es decir, se desarrolla el pensamiento lógico" [2].

Sin embargo, existen desafíos futuros en relación con la enseñanza de la programación en la educación universitaria. Uno de ellos es el creciente ritmo de cambio en la tecnología, que implica la necesidad de mantenerse actualizado constantemente. Asimismo, existe la necesidad de incorporar nuevos enfoques pedagógicos para lograr que la enseñanza de la programación sea más accesible y efectiva para un público cada vez más diverso.

Es evidente que, en los últimos años, se ha vuelto cada vez más popular utilizar diferentes formatos para los contenidos de Internet, tales como vídeos, audios (*podcasts*), gráficos y animaciones, entre otros. Aplicado a la educación, esto se conoce como educación multimedia, y tiene como objetivo proporcionar una experiencia de aprendizaje más dinámica e interactiva. Un ejemplo de esto es el aumento de la popularidad de las plataformas de vídeo, las cuales tienen entre su oferta también vídeos educativos.

Existen diversas plataformas web dedicadas a compartir vídeos para la educación. Algunas de las más populares son:

- **YouTube** [3]: Aunque es principalmente una plataforma de entretenimiento, también cuenta con una gran cantidad de vídeos educativos en una amplia variedad de temas.
- **Khan Academy** [5]: Es una plataforma educativa que ofrece vídeos sobre temas como matemáticas, ciencias, historia y economía, entre otros.
- **TED-Ed** [6]: Se enfoca en vídeos cortos que exploran conceptos educativos y de divulgación científica o de experiencias personales en una amplia variedad de temas.

- **Coursera** [7]: Ofrece cursos en línea y en vídeo de varias universidades y organizaciones en todo el mundo, que cubren una amplia variedad de temas académicos.
- **edX** [8]: Ofrece una variedad de cursos y programas en línea de universidades de todo el mundo, incluyendo vídeos de lecciones y tutoriales en una variedad de temas.
- **Udacity** [9]: Es una plataforma educativa en línea que ofrece cursos en vídeo en tecnología, ciencias de la computación, negocios y otros temas.
- **MiriadaX** [10]: Ofrece cursos en línea gratuitos en español de universidades de todo el mundo, que cubren una amplia variedad de temas académicos, como tecnología, educación, negocios, ciencias, humanidades y más.

Estas plataformas pueden ser muy efectivas para difundir vídeos de programación. La accesibilidad y usabilidad de estas plataformas hacen que sea fácil para cualquier persona crear una cuenta y empezar a subir vídeos, o simplemente buscarlos y reproducirlos, lo que significa que hay una audiencia potencial muy amplia para este tipo de contenido.

Sin embargo, el éxito de un vídeo de programación en una de estas plataformas depende de varios factores. Uno de los más importantes es la calidad del contenido, ya que los vídeos que ofrecen información útil y bien presentada tienen más probabilidades de ser compartidos y atraer a una audiencia más amplia. Por el contrario, vídeos de baja calidad o poco interesantes pueden tener un alcance limitado.

Es importante tener en cuenta la fiabilidad del contenido educativo que se puede encontrar en estas plataformas, a diferencia de las fuentes tradicionales de información educativa, como los libros de texto y los artículos académicos, cualquier usuario puede subir contenido a ellas sin la necesidad de ser un experto en el tema, aunque no en todas es igual de fácil compartir contenido. Por lo tanto, es posible encontrar información errónea, sesgada o engañosa en estas plataformas.

## 1.2. Antecedentes de la docencia *online*

La Universidad Nacional de Educación a Distancia (**UNED**) de España es una de las instituciones pioneras en el uso de las Tecnologías de la Información y la Comunicación (**TIC**) para la enseñanza a distancia y la educación en línea (*online*). Desde su fundación en 1972, la UNED ha estado a la vanguardia de la educación a distancia, adaptándose a los cambios tecnológicos y utilizando diferentes plataformas y herramientas para mejorar la

experiencia de aprendizaje de sus estudiantes. A lo largo de los años, la UNED ha implementado diversas estrategias innovadoras para facilitar el aprendizaje a distancia, que en los últimos años ha pasado a ser en línea, incluyendo el uso de plataformas de aprendizaje virtual, sistemas de tutoría online y recursos de aprendizaje multimedia, para mejorar la personalización y eficacia de la enseñanza a distancia [36].

En un contexto más amplio, el programa **ICT PSP** (Information Communication Technology Policy Support Programme), que formaba parte del programa para la competitividad e innovación (**CIP**, por sus siglas en inglés) de la Unión Europea, estuvo activo entre 2007 y 2013. Su principal objetivo era impulsar la innovación y la competitividad a través de una utilización más amplia y eficiente de las TIC por parte de los ciudadanos, gobiernos y empresas, especialmente las PYMEs (Pequeñas y Medianas Empresas).

El ICT PSP financió iniciativas que demostraban el potencial de las TIC para abordar diferentes desafíos sociales, desde la mejora de los servicios de salud hasta el gobierno electrónico y el envejecimiento activo. Además, fomentaba la creación de redes de cooperación entre diferentes actores. Tras el año 2013, el ICT PSP fue sucedido por el programa **Horizon 2020**, y desde 2021 por **Horizon Europe**, ambos con objetivos similares, pero con una mayor orientación hacia la promoción de la investigación y la innovación en la Unión Europea.

Siguiendo esta línea, la Universidad de Málaga ha implementado varios Proyectos de Innovación Educativa (**PIE**), orientados a integrar las TIC en su enseñanza. Uno de los proyectos más destacados es el **PIE17-175**, titulado "**Autoaprendizaje de programación de ordenadores**".

El artículo "Multimedia System for Self-learning C/C++ Programming Language" [35], desarrollado como parte del PIE 17-175 expone el desarrollo de un sistema de aprendizaje autónomo para la programación en C/C++. Los autores del estudio, José Galindo, Patricia Galindo y José María Rodríguez Corral, crearon una serie de archivos de presentación y vídeos que cubren los aspectos más complejos de la programación en C, permitiendo a los estudiantes aprender a su ritmo y en su propio horario. El proyecto sigue aumentando sus contenidos, ahora bajo el denominado **PIE22-218** de la convocatoria **INNOVA22** de la Universidad de Málaga.

Este estudio anterior subraya que aprender a programar en cuatro meses es un desafío importante, especialmente para aquellos estudiantes que no pueden asistir regularmente a las clases o que se encuentran en un nivel diferente al de la clase. Los

autores realizaron encuestas a los estudiantes para evaluar la efectividad del material, y los resultados revelaron que este recurso es útil tanto para los asistentes regulares a las clases como para aquellos que no pueden hacerlo, siendo preferentemente utilizado para el estudio en casa.

Otro proyecto relacionado con la docencia *online* es el TFG del estudiante Sergio Fernández Casasola, que en su proyecto "Vídeos docentes para el aprendizaje de programación C/C++ en Ingeniería Industrial" [47] ha desarrollado una serie de vídeos relacionados con la enseñanza de los lenguajes de programación C/C++.

Por último, es importante mencionar que el tutor a cargo del proyecto, en un esfuerzo constante por facilitar el aprendizaje, continúa generando contenido audiovisual relacionado con la informática a través de su canal de YouTube, "**Aprende conmigo Informática**"[4].

Este material está disponible para los estudiantes durante todo el curso, permitiéndoles revisar y profundizar en los temas a su propio ritmo. Los contenidos del canal son vídeos para aprender informática sin conocimientos previos. Particularmente, se centra en:

- **Conceptos básicos sobre informática**: *hardware*, *software*, la memoria, la CPU, sistemas operativos, bases de datos, código binario...
- **Conceptos básicos sobre programación**: compiladores, intérpretes, pseudocódigo...
- **Programación básica en lenguaje C/C++** (llega hasta *arrays* estáticos, estructuras, estructuras con *arrays*, listas con *arrays* de estructuras, algoritmos básicos de ordenación y de búsqueda).
- **MATLAB** (objeto del presente TFG): conceptos básicos sobre este *software* de programación matemática ampliamente utilizado en la industria y la investigación.
- Se están preparando también vídeos sobre **programación en PYTHON**, para un futuro cercano.

## 1.3. Objetivos

El objetivo principal del presente trabajo será crear vídeos educativos para aprender a utilizar MATLAB desde el principio, sin necesidad de tener conocimientos previos. Estos vídeos seguirán principalmente los contenidos de uno de los temas de la asignatura Sistemas Informáticos, impartida en la Escuela de Ingenierías Industriales de la Universidad

de Málaga, por diversos profesores, entre ellos el tutor del presente TFG, el Dr. José Galindo Gómez.

En este estudio, los vídeos serán subidos a la plataforma de vídeo YouTube [3], debido a su popularidad y a que permite a los usuarios cargar, ver, compartir y comentar vídeos de forma gratuita. Además, como ya se ha mencionado, el tutor de este proyecto ya tiene operativo un canal de YouTube [4] con 900 suscriptores a la fecha de redacción de esta memoria.

Por supuesto, uno de los objetivos de este trabajo es ofrecer contenido educativo fiable y de calidad para los estudiantes. Para lograrlo, se utilizarán fuentes confiables y se verificará la información antes de incluirla en el contenido.

A través de este enfoque educativo, se busca ayudar al alumnado a comprender mejor los conceptos teóricos y su aplicación práctica en la ingeniería. Los vídeos explicativos permitirán a los estudiantes profundizar en la teoría, mientras que los problemas prácticos les permitirán aplicar lo que han aprendido en un contexto real. Además, el contenido audiovisual permitirá al alumnado aprender a su propio ritmo y de una manera más visual e interactiva, lo que puede aumentar su motivación y su capacidad para retener la información. Al ser un medio audiovisual, los vídeos pueden resultar más atractivos y amenos para los alumnos que tienen dificultades para comprender los conceptos a través de la lectura o las explicaciones en clase, pues, los vídeos pueden repetirse tantas veces como sea necesario, lo que permite al alumnado revisar y repasar los conceptos.

El público objetivo de estos vídeos no sólo serán los alumnos universitarios, sino también cualquier persona interesada en aprender MATLAB, independientemente de su nivel de conocimiento previo. De esta manera, se busca difundir y hacer accesible el aprendizaje de la programación en MATLAB a una audiencia diversa, que podría incluir a estudiantes de secundaria o personas autodidactas.

Además, es una excelente idea considerar la posibilidad de que los vídeos educativos que se creen también tengan el objetivo de estar listos para una posible situación de educación a distancia forzada, como la ocurrida durante la pandemia. La propagación del virus llevó a la implementación de medidas de distanciamiento social y cierre de escuelas, lo que resultó en la interrupción de la educación de millones de estudiantes, y se ha de tener en cuenta que diversos científicos alertan de que nuevas pandemias podrían surgir en cualquier momento debido, entre otros factores, a los problemas medioambientales. La pandemia de COVID-19 ha demostrado la importancia de contar con recursos educativos

que puedan ser accesibles desde cualquier lugar y en cualquier momento. Estos vídeos educativos serían una herramienta valiosa para lograrlo.

### 1.3.1. Alcance y objetivos específicos

Para mejorar la difusión de los vídeos educativos de programación en YouTube [3], se tendrán en cuenta las siguientes estrategias:

1. Ofrecer contenido educativo de calidad: El objetivo principal es proporcionar material educativo de alto nivel para facilitar el aprendizaje de MATLAB. Cada vídeo será cuidadosamente diseñado y organizado, enfocándose en un tema específico relacionado con MATLAB, de manera que sea fácil de comprender y atractivo para los estudiantes. Además, se desarrollará un temario de estudios integral y estructurado para guiar a los alumnos a través del proceso de aprendizaje de MATLAB, abarcando desde conceptos básicos hasta técnicas un poco avanzadas. Como ya se dijo anteriormente, esta tarea se realizará teniendo en cuenta el contenido de la asignatura Sistemas informáticos. Este temario ayudará a los estudiantes a adquirir habilidades y conocimientos esenciales de manera progresiva, asegurando una experiencia de aprendizaje efectiva y enriquecedora.
2. La duración de cada vídeo se intentará que sea entre 3 y 12 minutos, salvo que algún vídeo requiera salirse de esos límites por motivos bien justificados. Esto es importante para captar la atención del estudiante durante todo el vídeo y para no aburrir demasiado. Si hay temas más largos, se estudiará si se dividen en vídeos diferentes.
3. Se cuidarán detalles de calidad general en los vídeos:
    - Cada vídeo debe tener una caratula y una despedida, incluyendo el nombre del curso y el título del vídeo lo más preciso posible, pero sin que sea excesivamente largo.
    - En la descripción del vídeo que se incluirá en la plataforma YouTube, se hará una lista detallada de todos los conceptos que se explican o se incluyen en el vídeo.
    - No pasarán más de 5-8 segundos con el vídeo en silencio y sin movimiento.
    - El sonido debe escucharse adecuadamente, sin ruidos de fondo, y a un volumen aceptable.
    - Si se está utilizando el cursor para la explicación, que el movimiento de este sea controlado y que no se mueva bruscamente o sin sentido. El cursor también debe

tener un tamaño razonable para que cualquiera pueda seguir su movimiento con relativa facilidad, pero tampoco debe ser demasiado grande para evitar tapar mucho el contenido.

- No deben aparecer textos y posteriormente desaparecer con gran rapidez que no dé tiempo a leerlos.
- La pronunciación ha de ser correcta, a ritmo moderado y sin titubeos. También se debe evitar un tono de voz excesivamente monótono.
- Cuando se use una palabra técnica o poco usual, se intentará que la primera vez también aparezca escrita, para que el usuario sepa a que nos estamos refiriendo.
- Resaltar de alguna manera la parte concreta de un código o la instrucción que este siendo explicada por la voz.
- Se usarán flechas y figuras animadas para enlazar o resaltar ciertas partes del vídeo cuando sea preciso.
- Las letras de cada texto que aparezca en los vídeos será de mínimo 16 puntos (salvo alguna excepción que pudiera darse y que este bien justificada).
- Se pondrán las palabras importantes en negrita para resaltarlas del resto del texto.
- Se usarán imágenes libres de copyright.
- Se deben evitar las imágenes borrosas.
- Los vídeos deberán tener en cuenta los conceptos que el alumno habría adquirido en vídeos anteriores y podrá remitirse a ellos.

4. Los metadatos del vídeo se seleccionarán con cuidado.

- Se buscarán títulos atractivos e informativos que sean claros, concisos y relevantes al tema tratado en el vídeo para atraer a la audiencia interesada en el tema.
- Se utilizarán las palabras clave y la descripción del vídeo para dar información sobre su contenido. Es importante utilizar etiquetas y palabras clave relevantes en la descripción del vídeo.
- Se usarán miniaturas representativas del contenido del vídeo, de modo que los espectadores se sientan motivados a hacer clic y ver el vídeo completo.

- Siguiendo las posibilidades de YouTube, cada vídeo tendrá además:
  - ❖ Tarjetas: Son vídeos sugeridos en momentos puntuales. A lo largo de la visualización de un vídeo, en la parte superior se puede sugerir otro vídeo que explica o que está relacionado con lo que se está diciendo.
  - ❖ Recomendaciones finales: Al final de cada vídeo, se añadirán dos recomendaciones, que pueden ser tanto vídeos concretos como listas de reproducción.
  - ❖ Botón de suscripción: También se añadirá al final de cada vídeo un botón que permita a los usuarios suscribirse al canal.

## 1.4. Plan de trabajo

Para realizar este trabajo, se han debido llevar a cabo una serie de tareas y actividades que se detallan a continuación.

1. Estudio del contenido teórico y práctico de MATLAB en la asignatura Sistemas Informáticos citada anteriormente.
2. Lectura de documentación relacionada con MATLAB y su programación, para poder afrontar el presente trabajo con una base de conocimientos suficientemente sólida.
3. Aprendizaje de herramientas para la creación de vídeos y audios de calidad.
4. Redacción del guion de cada vídeo, teórico y práctico, así como redacción del enunciado de los problemas específicos y elaboración de posibles soluciones.
5. Elaboración del contenido audiovisual de carácter teórico y práctico: audios, animaciones, presentaciones...
6. Edición de los vídeos, incluyendo pruebas, corrección de errores o mejoras.
7. Subida de los vídeos al canal docente de YouTube “Aprende conmigo Informática” [4].
8. Redacción de la memoria del trabajo.

# 1.5. Estructura de la memoria

El presente trabajo se compone de los siguientes capítulos:

- Capítulo 1: Introducción, antecedentes y objetivos
- Capítulo 2: MATLAB. Se hará una introducción de que es MATLAB, y se expondrá el temario que deberá ser abordado por los vídeos de este trabajo.
- Capítulo 3: Herramientas utilizadas. Se explicarán brevemente todas las herramientas empleadas para la consecución de los objetivos planteados.
- Capítulo 4: Metodología. Se explicará el procedimiento llevado a cabo para la elaboración de los vídeos, desde la planificación, hasta la edición.
- Capítulo 5: Vídeos creados y publicados. Se resumirá cada uno de los vídeos creados. Se incluirán capturas representativas del contenido.
- Capítulo 6: Conclusiones, objetivos alcanzados y futuros desarrollos. Se enumerarán de forma resumida las principales conclusiones obtenidas tras la realización de este trabajo, así como una lista de las cuestiones que podrían hacerse para continuar con este trabajo.
- Referencias. Se incluyen al final la lista de todas las referencias bibliográficas y electrónicas empleadas a lo largo de este trabajo.

# Capítulo 2:
# MATLAB

Como se mencionó en el capítulo anterior, el propósito de este trabajo es brindar un recurso adicional que facilite el entendimiento y la asimilación de los conceptos, con el fin de respaldar el logro académico de los alumnos en el área de programación y resolución de problemas utilizando MATLAB.

En este capítulo, se abordarán algunos aspectos sobre MATLAB y cómo desarrollar un programa de estudios enfocado en que los estudiantes adquieran los fundamentos esenciales de este lenguaje de programación. Para ello, se tomará como referencia el plan de estudios de la asignatura Sistemas Informáticos, impartida en la Escuela de Ingenierías Industriales de la Universidad de Málaga, por diversos profesores, entre ellos el tutor del presente TFG, el Dr. José Galindo Gómez.

## 2.1. Historia de MATLAB

**MATLAB** (abreviatura de "**MATrix LABoratory**") [22] es un entorno y lenguaje de programación de alto nivel. Fue creado en la década de 1970 por **Cleve Moler**, entonces profesor de matemáticas y ciencias de la computación en la Universidad de Nuevo México. Moler diseñó MATLAB como una herramienta para facilitar el acceso a las librerías de cálculo matricial de FORTRAN, uno de los lenguajes de programación más populares en aquella época.

En 1984, Moler se asoció con **Jack Little**, un ingeniero de control que había trabajado con Moler en la implementación de MATLAB en computadoras personales. Junto con **Steve Bangert**, fundaron **MathWorks** para comercializar MATLAB y su paquete de funciones adicionales, **Simulink**. Desde entonces, MATLAB ha experimentado un desarrollo continuo,

con mejoras en su lenguaje de programación, nuevas bibliotecas y herramientas, y un amplio ecosistema de soporte.

Como parte del presente TFG se ha elaborado un vídeo específico para resumir la historia de MATLAB y sus fundamentos básicos. El contenido del vídeo se resume en la Sección 5.1.1. En la Figura 2.1 se aprecia un fotograma de ese vídeo.


Figura 2.1 : Fotograma del vídeo titulado Qué es MATLAB

## 2.2. Características de MATLAB

MATLAB ofrece una serie de características específicas que lo convierten en un *software* muy interesante para la resolución de problemas matemáticos y técnicos. A continuación, se describen algunas de estas características con algo de detalle:

- Su **entorno de desarrollo integrado** (**IDE**, del inglés Integrated Development Environment) proporciona herramientas para la edición de código, la depuración y el análisis de datos. El IDE incluye un editor de código con resaltado de sintaxis, un depurador para identificar y corregir errores, así como herramientas de perfilado para analizar el rendimiento del código. También permite a los usuarios administrar archivos y proyectos, lo que facilita la organización del trabajo.
- Su **lenguaje de programación de alto nivel**, también conocido como **lenguaje M**, está diseñado para facilitar el trabajo con matrices y álgebra lineal, lo que lo hace

especialmente útil en campos como la ingeniería y las ciencias aplicadas. Además, permite realizar operaciones matriciales de manera intuitiva, utilizando una sintaxis sencilla y familiar para quienes tienen conocimientos básicos de matemáticas. Incluye un amplio conjunto de funciones matemáticas integradas que abarcan áreas como el cálculo, la estadística, la optimización y las transformadas como pueden ser la transformada de **Fourier**, la transformada de **Laplace** o la transformada **Z**.

- Su capacidad para visualizar datos de manera rápida y eficiente. Incluye funciones para crear **gráficos 2D**, **3D** y **4D**, histogramas y gráficos de barras, lo que permite a los usuarios explorar y analizar datos visualmente. Además, ofrece opciones de personalización para adaptar las visualizaciones a las necesidades específicas del usuario.

- Cuenta con bibliotecas especializadas, conocidas como **Toolboxes**. Algunas de estas bibliotecas son el **Optimization Toolbox**, el **Statistics and Machine Learning Toolbox** y el **Signal Processing Toolbox**, que proporcionan funciones y algoritmos adicionales para abordar problemas específicos en diversas disciplinas. También hay un Toolbox para, por ejemplo, facilitar el acceso a bases de datos desde MATLAB.

- Puede **interactuar con otros lenguajes de programación y *software***, lo que permite a los usuarios integrar su trabajo en MATLAB con otros entornos de desarrollo. Por ejemplo, puede llamar a funciones escritas en C, C++, Python y Fortran, o interactuar con aplicaciones de *software* como Excel y diversas bases de datos.

- Admite la **programación orientada a objetos** (**OOP**), lo que permite a los usuarios crear y manipular objetos con propiedades y métodos específicos. La OOP facilita la organización y el mantenimiento del código, especialmente en proyectos grandes y complejos.

- Es compatible con la **computación paralela y distribuida**, lo que permite a los usuarios aprovechar múltiples núcleos de procesadores, **GPU** (Graphics Processing Unit) y clústeres para acelerar cálculos y simulaciones. Esto es especialmente útil en aplicaciones que requieren un gran volumen de cálculos o en las que se necesita procesar grandes conjuntos de datos.

- Incluye herramientas para crear **aplicaciones personalizadas e interfaces gráficos de usuario** (**GUI**, del inglés Graphical User Interface), lo que facilita la

creación de *software* específico para resolver problemas o realizar análisis en un dominio particular. Estas aplicaciones y GUI pueden compartirse con otros usuarios, incluso aunque no tengan conocimientos en programación de MATLAB o ni siquiera dispongan de ese *software*. Esto facilita la colaboración y la comunicación de resultados en entornos académicos y profesionales.

- Su ecosistema incluye un **amplio conjunto de recursos y una comunidad activa de usuarios** que comparten conocimientos y herramientas. Los usuarios pueden acceder a **MATLAB Central** [37], un repositorio en línea que contiene archivos, tutoriales, ejemplos de código y respuestas a preguntas comunes. Además, existen numerosos grupos de usuarios, foros en línea y eventos que facilitan el intercambio de conocimientos y la colaboración entre profesionales y académicos que utilizan MATLAB. También ofrece un **centro de ayuda *online*** [38] con toda la documentación necesaria para aprender su lenguaje de programación. Este último ha resultado de gran utilidad para el desarrollo del presente trabajo.

Para resumir, se puede decir que las características de MATLAB lo convierten en una herramienta poderosa y versátil para abordar problemas en una amplia variedad de disciplinas. Su lenguaje de alto nivel orientado a matrices, funciones matemáticas y librerías especializadas, capacidades de visualización de datos, compatibilidad con otros lenguajes y *software*, programación orientada a objetos, entorno de desarrollo integrado (IDE), compatibilidad con la computación paralela y distribuida, soporte para el desarrollo de aplicaciones y GUI, y su ecosistema, comunidad de usuarios y centro de ayuda lo posicionan como una herramienta esencial en el ámbito académico y profesional.

## 2.3. Inconvenientes de MATLAB y exploración de alternativas

MATLAB, aunque es una herramienta poderosa para la computación numérica y la visualización de datos, tiene varios inconvenientes que pueden limitar su eficacia en ciertos contextos. Uno de los principales desafíos asociados con MATLAB es su alto costo. Como *software* propietario, las licencias comerciales pueden ser prohibitivas, especialmente para pequeñas empresas o individuos. Aunque MATLAB ofrece versiones para estudiantes y licencias grupales, el costo puede seguir siendo un obstáculo significativo.

Además, a pesar de su robustez, MATLAB puede ser menos eficiente en términos de velocidad y uso de memoria en comparación con los lenguajes de programación de bajo

nivel como C o C++. Esta ineficiencia puede ser un problema para aplicaciones que requieren un alto rendimiento.

Otro desafío con MATLAB es su dependencia de cajas de herramientas (Toolboxes) adicionales para ciertas funciones avanzadas. Estas cajas de herramientas deben comprarse por separado, lo que puede aumentar aún más el costo de usar MATLAB.

Además, aunque MATLAB es relativamente fácil de aprender en comparación con algunos lenguajes de programación, algunos usuarios pueden encontrar su sintaxis y estructura inicialmente difíciles de entender, especialmente si están más familiarizados con otros lenguajes. En términos de portabilidad y compatibilidad, MATLAB no es tan universal como otros lenguajes. No todos los sistemas y plataformas son compatibles con MATLAB, lo que puede limitar su uso en diferentes contextos. Finalmente, a pesar de que MATLAB tiene una comunidad de usuarios activa, no cuenta con la misma extensión y diversidad de la comunidad de código abierto que respalda lenguajes como **Python** y **R**. Esto puede limitar la disponibilidad de bibliotecas de código abierto y la resolución de problemas específicos.

Dado estos inconvenientes, es útil considerar alternativas a MATLAB. Python y R, por ejemplo, son lenguajes de programación de código abierto que pueden realizar muchas de las mismas funciones que MATLAB. Son gratuitos para usar y tienen comunidades de código abierto grandes y activas que pueden proporcionar una variedad de bibliotecas y soporte para problemas específicos. Además, Python y R son más universales y pueden ser más portables y compatibles que MATLAB. En términos de eficiencia, los lenguajes de programación de bajo nivel como **C** o **C++** pueden ser más eficientes en términos de velocidad y uso de memoria. Aunque estos lenguajes tienen curvas de aprendizaje más pronunciadas, pueden ser más adecuados para aplicaciones que requieren un alto rendimiento.

## 2.4. Aplicaciones de MATLAB en el ámbito académico

En la investigación académica, MATLAB se utiliza ampliamente en campos como la física, la ingeniería, la biología y las ciencias de la computación. Su capacidad para realizar cálculos numéricos y simbólicos de forma rápida, lo convierte en una herramienta valiosa para técnicos e investigadores que trabajan en problemas complejos.

También se utiliza en la enseñanza, tanto a nivel de pregrado como de posgrado. Sus capacidades de visualización y su lenguaje fácil de aprender permiten a los estudiantes comprender conceptos abstractos y aplicarlos en la resolución de problemas prácticos. En el ámbito académico, resulta ser una herramienta poderosa para la simulación y el modelado de sistemas dinámicos, como sistemas eléctricos, mecánicos, químicos y biológicos. De hecho, su paquete adicional Simulink, proporciona un entorno gráfico para modelar, simular y analizar sistemas dinámicos multidominio. Esto permite a los investigadores y estudiantes evaluar y comprender el comportamiento de sistemas complejos, así como diseñar soluciones eficaces.

Asimismo, es ampliamente utilizado en el procesamiento de señales e imágenes, donde se aplican técnicas matemáticas para analizar, modificar y sintetizar señales e imágenes. Estas aplicaciones incluyen la detección de patrones, la restauración de imágenes, la compresión de datos y la extracción de características.

El aprendizaje automático y la ciencia de datos son campos en rápido crecimiento que utilizan algoritmos y modelos matemáticos para extraer información valiosa de grandes conjuntos de datos. MATLAB ofrece herramientas y bibliotecas especializadas que facilitan la implementación y evaluación de algoritmos de aprendizaje automático y técnicas de análisis de datos.

Por tanto, podemos concluir que el ámbito académico se beneficia del uso de MATLAB en la enseñanza e investigación de estas áreas, así como en el desarrollo de nuevas técnicas y algoritmos destinados a desarrollar soluciones para problemas del mundo real.

## 2.5. Temario

Como el objetivo de este trabajo es hacer vídeos educativos para un curso de introducción al lenguaje de programación de MATLAB, se tendrá que establecer un temario inicial en el cual se basarán estos vídeos. Este temario, tal como se mencionó al principio de este capítulo, se basará en los apuntes de la asignatura Sistemas Informáticos de diversos Grados en Ingenierías Industriales de la Universidad de Málaga. Particularmente, esa asignatura se imparte en los grados de Electrónica Industrial, Electricidad y Mecánica, así como en los dobles grados asociados. Su temario oficial consta de tres temas y puede verse en su guía docente [39]. Este temario aparece en el siguiente listado tal y como aparece en la guía docente:

1. **Bases de datos**

   # **OBJETIVO**: Hoy día cualquier empresa suele tener su base de datos, por lo que es importante tener conceptos básicos que faciliten su uso y que permitan la comunicación con los que empleen o administren bases de datos.

   # **CONTENIDOS**:

   - Introducción a las BD relacionales: Modelo relacional.
   - Modelo Entidad/Relación (ER).
   - Lenguajes relacionales (SQL).
   - Prácticas con un Sistema Gestor de Bases de Datos (SGBD) actual: Diseño, relaciones, consultas simples...

2. **Resolución de problemas científicos con MATLAB**

   # **OBJETIVO**: Aprender los fundamentos de Matlab desde un nivel principiante. Se resolverán problemas típicos en cualquier ingeniería.

   # **CONTENIDOS**:

   - Introducción. Entorno de trabajo, ficheros .mat y .m
   - Operaciones con vectores y matrices.
   - Operaciones matemáticas básicas. - Funciones gráficas en 2D, 3D y 4D.
   - Sistemas de ecuaciones lineales.

3. **Entornos visuales**

   # **OBJETIVO**: Aprender a sacar beneficio de las aplicaciones en Matlab. Se verán programas simples y cómo desarrollar entornos atractivos e interactivos para el usuario.

   # **CONTENIDOS**:

   - Programación básica en Matlab: E/S, tipos de datos, y estructuras de control.
   - Creación de funciones simples.
   - Introducción a la creación visual de entornos de usuario.
   - Elementos para diseñar interfaces visuales: ventanas, eventos, componentes, etc.

- Integración con bases de datos.

Podemos apreciar que los Temas 2 y 3 tratan sobre MATLAB, y para este TFG nos hemos centrado en el Tema 2. Los contenidos de este segundo tema se exponen con un poco más de detalle en la Tabla 2.1.

| | |
|---|---|
| **Conceptos básicos sobre MATLAB**. | Escritorio de MATLAB. |
| | Ficheros .mat para almacenar variables. |
| | Operadores elementales. |
| | Números complejos. |
| | Ficheros .m o *scripts.* |
| **Vectores y matrices**. | Creación, composición y operaciones aritméticas elementales. |
| | Sistemas de ecuaciones lineales. |
| | Polinomios. |
| **Gráficos en MATLAB**. | Gráficos simples 2D. |
| | Personalización de gráficos (colores, tipos de línea, etc). |
| | Gráficos 3D (curvas y superficies). |
| | Gráficos 4D. |
| | Interpolación de datos 3D y representación gráfica. |
| | Reunión de gráficas en la misma figura |

Tabla 2.1 : Contenidos del tema 2 de la asignatura Sistemas Informáticos.

# Capítulo 3:
# Herramientas utilizadas

En este capítulo se describirán las herramientas *software* que se han empleado durante este trabajo. El objetivo de este trabajo no será hacer un manual completo de ellas, sino dar sus características más importantes, centrándose en aquellas que han sido relevantes para la realización del presente proyecto.

## 3.1. Microsoft PowerPoint

Microsoft PowerPoint [40] es un programa informático desarrollado por Microsoft para crear diapositivas y presentaciones multimedia. Fue lanzado en 1987 como una herramienta de presentación básica, pero desde entonces ha evolucionado para incluir una variedad de características avanzadas y efectos visuales. Hoy en día, es una herramienta ampliamente utilizada en entornos empresariales, educativos y gubernamentales para crear presentaciones de diapositivas profesionales.

Con PowerPoint, los usuarios pueden crear diapositivas con texto, imágenes, gráficos, tablas y otros elementos multimedia. La aplicación incluye una amplia variedad de plantillas predefinidas que pueden ser personalizadas para adaptarse a las necesidades del usuario. Las presentaciones pueden ser creadas en varias versiones del programa, entre las que encontramos PowerPoint 2016, PowerPoint 2019 y PowerPoint 2021. Además, es compatible con otros programas de Microsoft Office, como Word y Excel, lo que permite una fácil integración y transferencia de información.

Este programa también incluye herramientas de colaboración en línea, lo que permite a los usuarios trabajar en presentaciones en tiempo real con otros miembros del equipo. Además, los usuarios pueden compartir sus presentaciones a través de plataformas de intercambio de archivos como OneDrive y SharePoint.

La razón por la cual la herramienta ha sido incluida en este trabajo es su utilidad para crear vídeos educativos. La ventaja de usar PowerPoint para crear vídeos es su capacidad para agregar sonidos, efectos y animaciones a las diapositivas. Los usuarios pueden agregar transiciones suaves entre ellas, así como animaciones de entrada y salida para los elementos de estas, además de efectos de sonido, como pueden ser las narraciones pregrabadas.

Otra de las ventajas de PowerPoint para crear vídeos educativos es que es una herramienta que muchos educadores y estudiantes ya conocen y utilizan regularmente en sus cursos. Esto significa que no sería necesario aprender una nueva herramienta de edición de vídeo o descargar programas costosos.

### 3.1.1. Crear vídeos con Microsoft PowerPoint

La creación de un vídeo educativo a través de la herramienta PowerPoint implica seguir una serie de pasos que han de seguirse para obtener un resultado de calidad. En primer lugar, se requiere elaborar una presentación con diapositivas que contenga el material educativo que se desea incorporar en el vídeo. Es recomendable incluir imágenes, gráficos, texto y audio según corresponda al contenido.

Posteriormente, es fundamental animar el contenido de las diapositivas y ajustar los tiempos de dichas animaciones. Es crucial que la duración de las transiciones y animaciones permita a los estudiantes leer y comprender el contenido sin generar desinterés en los alumnos. Para agregar animaciones, es necesario seleccionar el elemento deseado y, posteriormente, hacer clic en la pestaña “Animaciones” ubicada en la barra superior de herramientas.

Cabe destacar que existen distintos tipos de efectos disponibles:

- Efectos de entrada (color verde): definen la manera en la que los elementos aparecen en la diapositiva para el lector.
- Efectos de énfasis (color amarillo): permiten aplicar efectos a los elementos que ya se encuentran a la vista del lector, por ejemplo, modificando su tamaño y color.
- Efectos de salida (color rojo): determinan la manera en la que los elementos desaparecen de la vista del lector en la diapositiva.
- Trayectorias de la animación: permiten mover los elementos a través de diferentes posiciones en la diapositiva.

Es esencial considerar que la selección de cada tipo de efecto dependerá del propósito educativo que se busque con el vídeo y del mensaje que se quiera transmitir. Asimismo, se debe tener en cuenta la adecuada combinación de los efectos para lograr una presentación clara y atractiva para el público estudiantil.

En caso de que se desee aplicar más de un efecto a un mismo elemento, por ejemplo, si se desea que una imagen aparezca y luego desaparezca después de cierto tiempo, se debe aplicar el primer efecto y luego hacer clic en “Agregar animación”. De esta manera, se desplegará un menú con todas las opciones de efectos que se pueden añadir a la animación del elemento seleccionado. De esta forma, se podrán crear presentaciones más dinámicas y con una mayor variedad de efectos visuales para el público estudiantil.

Una vez se hayan añadido los efectos, se tendrá la posibilidad de ajustar los intervalos, desencadenantes y otras propiedades correspondientes a los mismos. Para ello, es necesario abrir el “Panel de animación”, el cual se encuentra disponible en la barra de herramientas en el menú “Animaciones”. Al hacer clic en dicho botón, se desplegará un panel ubicado a la derecha de la ventana, que mostrará todos los efectos aplicados a los elementos de la diapositiva en cuestión. En la Figura 3.1 puede verse un ejemplo de diapositiva con el “Panel de animación” a la derecha y el menú “Animaciones” activo en la barra de herramientas en la parte superior. Cada línea con una estrella verde representa un objeto que tiene que aparecer y ahí se detalla el momento, la velocidad, si hay repeticiones, si el objeto se queda o desaparece, etc.

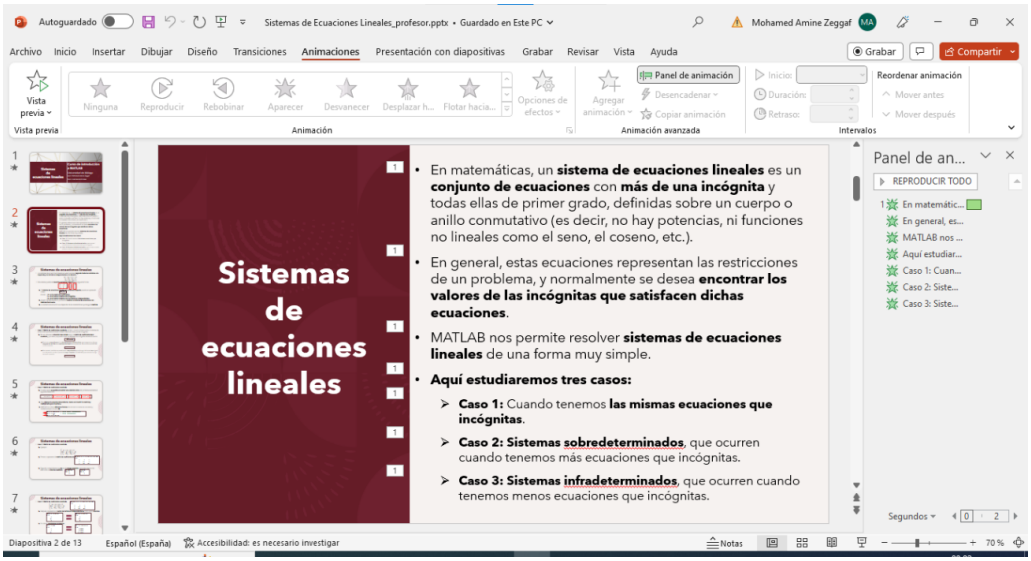


Figura 3.1 : Editando un vídeo con PowerPoint. A la derecha puede verse el “Panel de animación” y arriba en la barra de herramientas el menú “Animaciones”

En este panel, se requiere realizar doble clic sobre el efecto específico que se desee ajustar, para que se despliegue una nueva ventana. Esta última, suele contar con dos pestañas, si bien es posible que, dependiendo del tipo de efecto, se presenten más pestañas adicionales. Las dos pestañas iniciales son las de “Efecto” e “Intervalos”, respectivamente.

En la pestaña de “Intervalos”, se encuentran varias opciones de configuración para los efectos de la presentación. Es en el formulario que se despliega donde es posible establecer el modo de inicio, el retraso, la duración del efecto, las repeticiones y los desencadenadores tal y como se puede ver en la Figura 3.2.

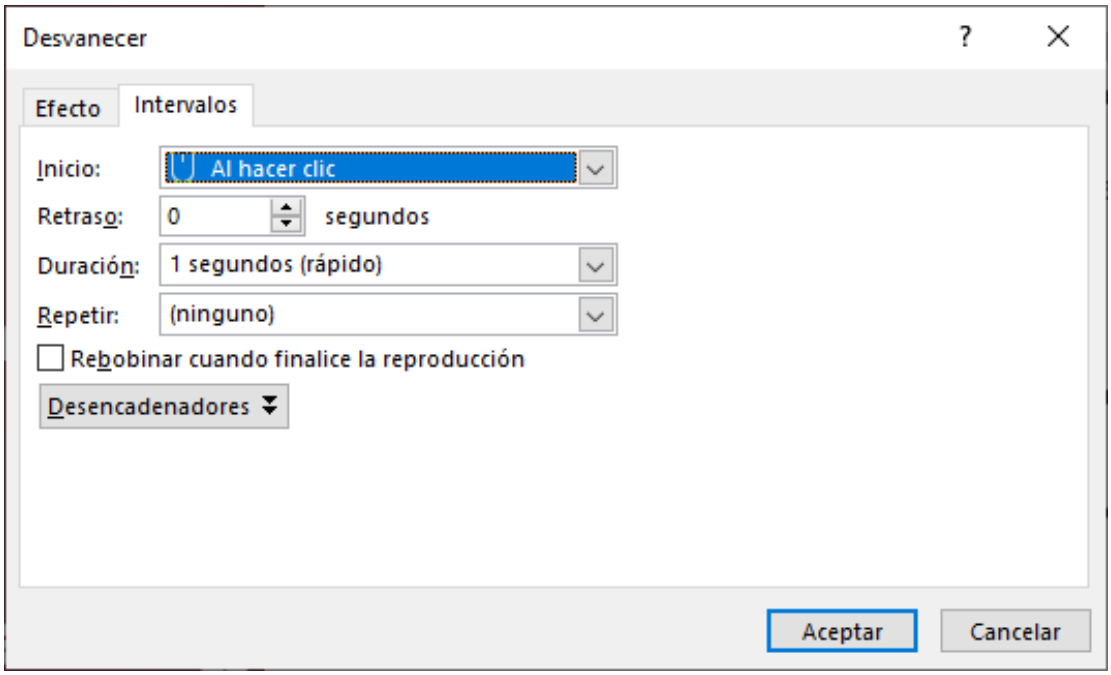


Figura 3.2 : Modificando intervalos de una animación en el formulario “Intervalos”

Para cada objeto, el programa ofrece tres modos de aparición diferentes: al hacer clic, con la anterior y después de la anterior. Esos modos los podemos elegir en el menú “Inicio” del formulario que aparece en la Figura 3.2. Para generar un vídeo correctamente, es necesario configurar la opción en modo con la anterior o después de la anterior. El efecto del primer objeto que aparezca debe ser configurado en modo después de la anterior para que empiece automáticamente al finalizar la presentación de la diapositiva anterior.

El “Retraso” se refiere al tiempo que debe transcurrir entre la activación del inicio y el comienzo del efecto. Esta opción se puede utilizar para planificar el orden de los efectos en la presentación. La duración del efecto determina el tiempo que transcurre mientras se ejecuta el efecto, siendo más rápido cuanto más corta sea la duración y más lento cuanto más larga sea.

La opción de “Repetición” se emplea para configurar la cantidad de veces que se desea que se repita un efecto, en caso de que sea necesario.

Además de las opciones de la pestaña “Intervalos”, también se dispone de la pestaña “Efecto”, que ofrece varias opciones de configuración que varían según el efecto aplicado. El formulario que se despliega al hacer clic en esta pestaña aparece en la Figura 3.3.

En el caso de esta presentación, la opción más relevante en la pestaña de efecto es la de “Sonido”, ya que permite agregar un audio al efecto, como una grabación de voz que describa la diapositiva a partir de la aparición del objeto en cuestión. En este trabajo se ha empleado la estrategia de asociar la grabación de voz de toda la diapositiva a la primera animación, y se ha introducido un retraso en la aparición de las siguientes animaciones en función de la línea de tiempo de la grabación.

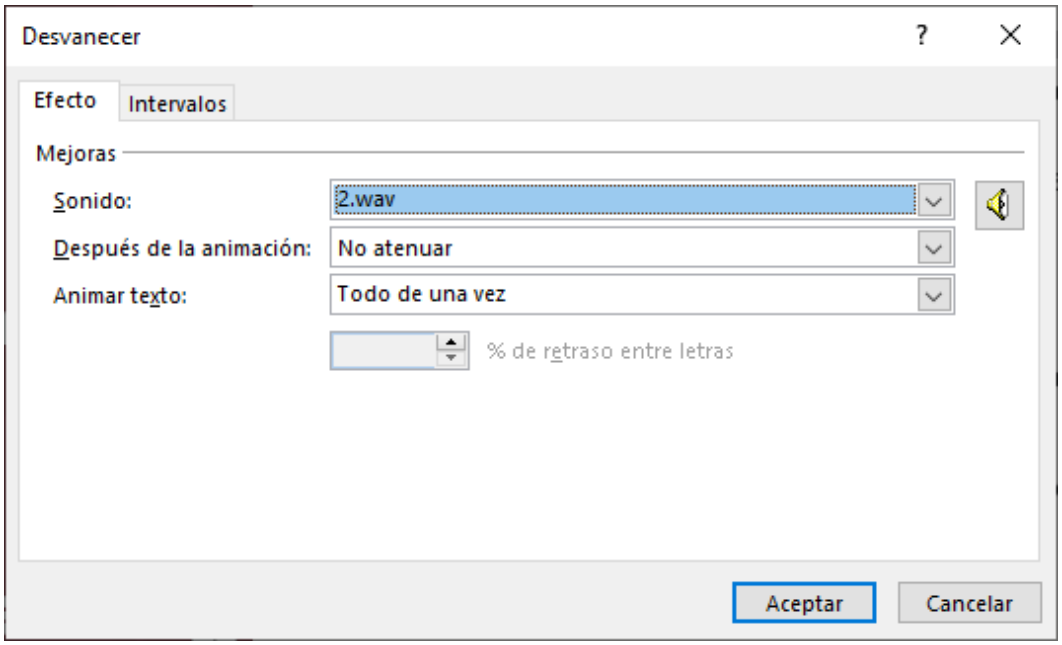


Figura 3.3 : Modificando efectos de una animación en el formulario “Intervalos”

Una vez configurados los efectos e intervalos de todas las animaciones de las diapositivas, solo resta generar el vídeo. Para hacerlo, se debe hacer clic en la pestaña “Archivo” en la barra de herramientas y, a continuación, en “Exportar”. Se abrirá un menú para poder exportar la presentación en diferentes formatos, y se deberá elegir la opción “Crear un vídeo”.

En ella se nos permite establecer la calidad del vídeo resultante. En este proyecto se ha optado por seleccionar la resolución ***Full HD* (1080p)**, la cual proporciona una calidad óptima sin generar un tamaño excesivo como podría suceder con la resolución ***Ultra HD* (4k)**.

# 3.2. Audacity

Audacity [41] es un programa de edición de audio de código abierto que proporciona una amplia gama de herramientas y funcionalidades a los usuarios para la grabación, edición y exportación de archivos de audio. Desde su lanzamiento en el año 2000, Audacity ha experimentado un constante desarrollo y se ha convertido en una herramienta indispensable para creadores de contenido, músicos, ingenieros de sonido y aficionados del audio en general.

En la presente sección, se describirán las características fundamentales de este programa, su trayectoria histórica y su capacidad para optimizar cualquier proyecto de audio. Posteriormente, se analizará el uso de Audacity para mejorar la calidad del audio en los vídeos educativos.

## 3.2.1. Historia de Audacity

Audacity fue concebido en el año 2000 por Dominic Mazzoni y Roger Dannenberg, quienes en ese entonces eran estudiantes de doctorado en la Universidad Carnegie Mellon. Lo desarrollaron como parte de un proyecto de investigación, y desde entonces el programa ha sido mantenido y perfeccionado por un equipo de desarrolladores voluntarios de distintas partes del mundo.

El proyecto comenzó alojándose en el sitio web de SourceForge [46] y, a lo largo de los años, ha sido galardonado y reconocido en varias ocasiones por su valiosa contribución a la comunidad del *software* de código abierto. La versión más reciente disponible al momento de redactar este documento es Audacity 3.2.5.

## 3.2.2. Características principales de Audacity

Audacity proporciona una plataforma eficiente para la grabación de audio proveniente de diversas fuentes, tales como micrófonos, mezcladores de audio y captura de audio desde la computadora en modo *loopback*. Además, ofrece soporte para grabaciones multipista, lo cual permite a los usuarios trabajar con varias pistas de audio de manera simultánea.

Este *software* brinda una amplia variedad de herramientas de edición de audio, incluyendo funciones básicas como cortar, copiar, pegar y eliminar. Asimismo, ofrece herramientas de edición avanzada, como la modificación de velocidad, tono y tempo, lo cual facilita la adaptación del audio a los requisitos específicos de cada proyecto. Además, proporciona una amplia selección de efectos para mejorar y procesar las grabaciones, tales

como ecualización, compresión, normalización, reverberación, eliminación de ruido entre otros. Dichos efectos pueden aplicarse a pistas de audio individuales o a una mezcla completa.

Audacity es compatible con una gran variedad de formatos de archivo, lo cual permite a los usuarios importar y exportar archivos de audio en diferentes formatos, tales como WAV, AIFF, MP3, OGG y FLAC, entre otros. Asimismo, es posible importar archivos MIDI y archivos de vídeo con el objetivo de extraer el audio.

El programa está disponible para sistemas operativos Windows, macOS y Linux, lo cual lo convierte en una herramienta accesible para usuarios de distintas plataformas.

### 3.2.3. Usos comunes de Audacity

Audacity es una herramienta sumamente popular entre los creadores de *podcasts* debido a su facilidad de uso y amplia variedad de funciones. Además, esta herramienta es ampliamente utilizada por músicos y productores para editar y mezclar pistas de audio, aplicar efectos y ajustar niveles de volumen. Aunque Audacity no es una estación de trabajo de audio digital (*DAW*) profesional completa, sigue siendo una herramienta útil para trabajos de edición y mezcla básicos.

Audacity se presenta como una herramienta de gran valor en entornos educativos, ya que tanto docentes como estudiantes pueden hacer uso de ella para grabar y editar presentaciones, proyectos de investigación y otros materiales de aprendizaje. Es relevante destacar que, al tratarse de un programa de código abierto y gratuito, se convierte en una opción altamente accesible para instituciones educativas con recursos limitados. En virtud de esta razón fundamental, se ha optado por su utilización en el presente trabajo.

### 3.2.4. Cómo usar Audacity

Para comenzar a utilizar este *software*, es necesario acceder al sitio web oficial de Audacity [41] con el fin de descargar la versión más actualizada y compatible con el sistema operativo en uso (Windows, macOS o Linux). Una vez descargado el archivo ejecutable, se deberá proceder a la instalación correspondiente. Una vez instalado, se debe ejecutar el programa y familiarizarse con su interfaz, la cual aparece en la Figura 3.4. Se caracteriza por su gran facilidad de uso y comprensión.

Posteriormente, es preciso configurar las preferencias de audio para garantizar una correcta utilización del programa. Para ello, se debe acceder al menú “Editar” en caso de emplear los sistemas operativos Windows o Linux, o al menú “Audacity” en caso de utilizar

el sistema operativo macOS. Este menú se encuentra ubicado en la esquina superior izquierda de la ventana de Audacity. Al desplegar la lista de opciones, se deberá seleccionar “Preferencias”. Al acceder a la ventana de “Preferencias”, se visualizarán diversas categorías en el lado izquierdo de la pantalla, tal y como se puede ver en la Figura 3.5. En la lista de categorías de la ventana de Preferencias, se debe seleccionar “Dispositivos” para configurar los dispositivos de grabación y reproducción.

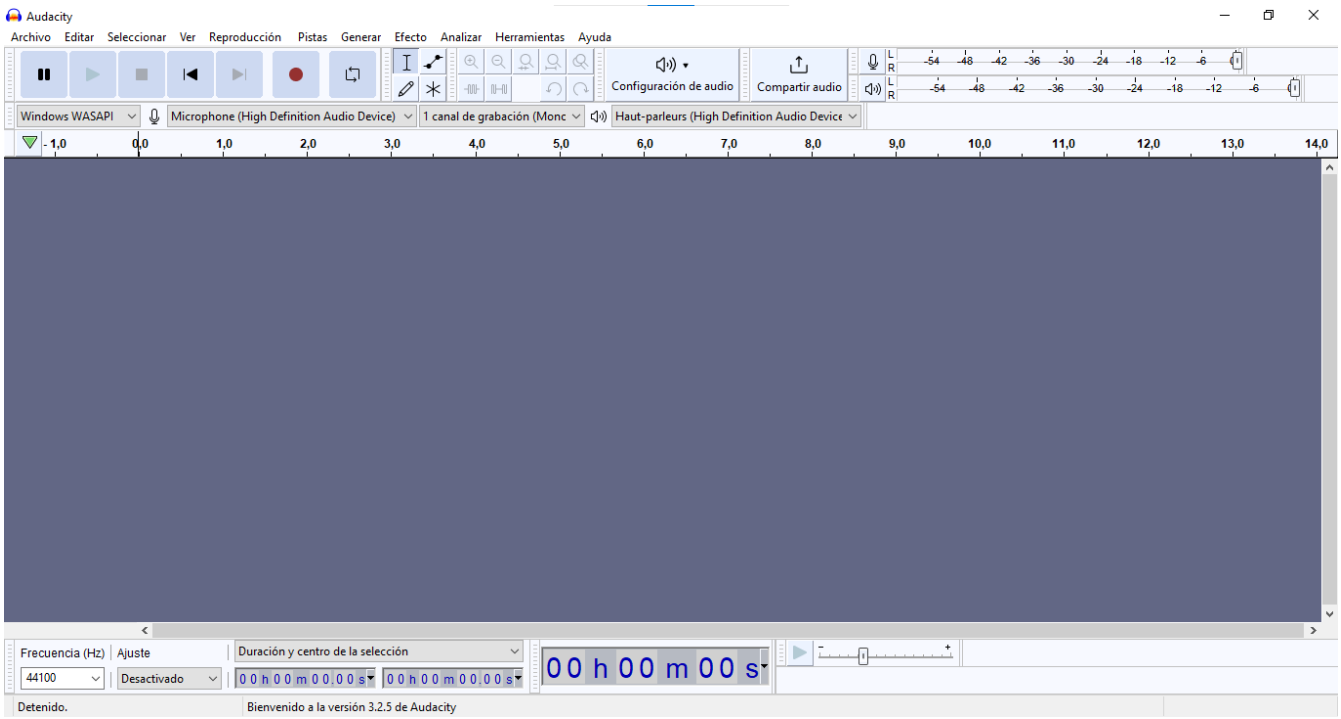


Figura 3.4 : Interfaz de usuario de Audacity

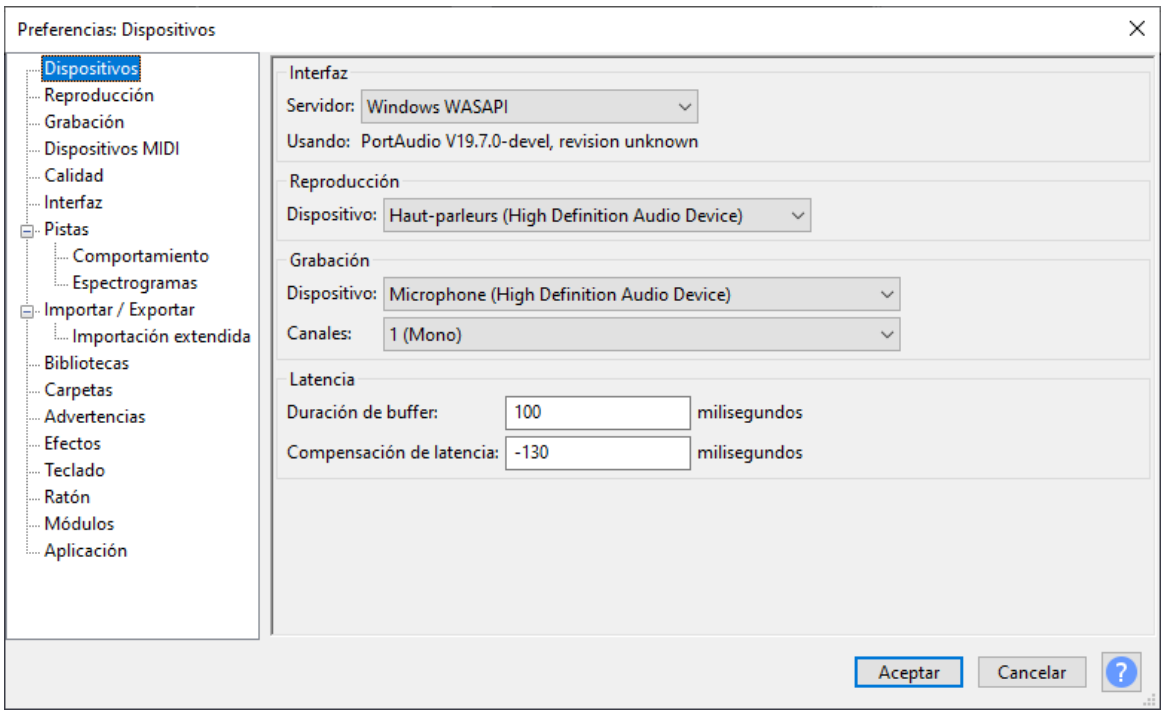


Figura 3.5 : Ventana de “Preferencias” en la categoría “Dispositivos”

En la sección de “Reproducción”, se halla un menú desplegable etiquetado también como “Dispositivo”. Al hacer clic en este menú, se podrá seleccionar la fuente de salida de audio, como altavoces o auriculares, para poder escuchar las grabaciones y ediciones realizadas en Audacity.

En la sección “Grabación”, se encuentra otro menú desplegable etiquetado también como “Dispositivo”. Al seleccionar este menú, se podrá elegir la fuente de entrada de audio deseada, como un micrófono, una interfaz de audio u otro dispositivo de entrada de audio conectado al equipo. Incluso se podrá utilizar como dispositivo de grabación las salidas de los altavoces si se requiere trabajar en modo *loopback*.

Debajo de la sección de “Dispositivos de grabación”, se encuentra una opción llamada “Canales”. Haciendo clic en ese menú desplegable, se podrá seleccionar la cantidad de canales para grabar el audio. Se podrá elegir entre *“Estéreo*” para grabar en dos canales o *“Mono”* para grabar en un único canal. Las opciones disponibles en este menú dependerán de las características del dispositivo seleccionado en el menú de dispositivos de grabación.

Una vez realizados los cambios necesarios en la configuración de los dispositivos de audio, se debe ajustar la calidad del audio. Para hacerlo, se debe hacer clic en la categoría “Calidad” en la lista de categorías de la ventana de “Preferencias”. Al seleccionar esta categoría, se mostrarán diversas opciones para ajustar, tal y como se puede ver en la Figura 3.6. En esta categoría, únicamente dos de ellas son de interés: la frecuencia de muestreo predeterminada y el formato de muestra predeterminado.

La frecuencia de muestreo es un aspecto fundamental en el proceso de grabación y reproducción de audio digital, y hace referencia a la cantidad de muestras extraídas de una señal de audio analógica por segundo con el fin de convertirla en una señal digital. La selección de la frecuencia de muestreo adecuada depende de varios factores, incluyendo el tipo de audio que se está grabando, el propósito del archivo de audio y los requisitos del dispositivo de reproducción.

En general, se recomienda elegir una frecuencia de muestreo que sea al menos el doble de la frecuencia más alta presente en la señal de audio original. Esto se debe a que el teorema del muestreo de Whittaker-Nyquist-Kotelnikov-Shannon establece que, para reconstruir una señal de audio analógica de manera precisa a partir de su versión digital, se debe muestrear a una tasa que sea al menos dos veces mayor que la frecuencia más alta presente en la señal analógica. Sin embargo, es importante tener en cuenta que seleccionar una frecuencia de muestreo más alta genera archivos de audio más grandes,

lo que puede ser un problema en situaciones donde el almacenamiento de datos es limitado, como en dispositivos móviles o en transmisiones de audio en tiempo real por Internet. En estos casos, se puede optar por una frecuencia de muestreo más baja, aunque esto puede afectar negativamente la calidad del sonido.

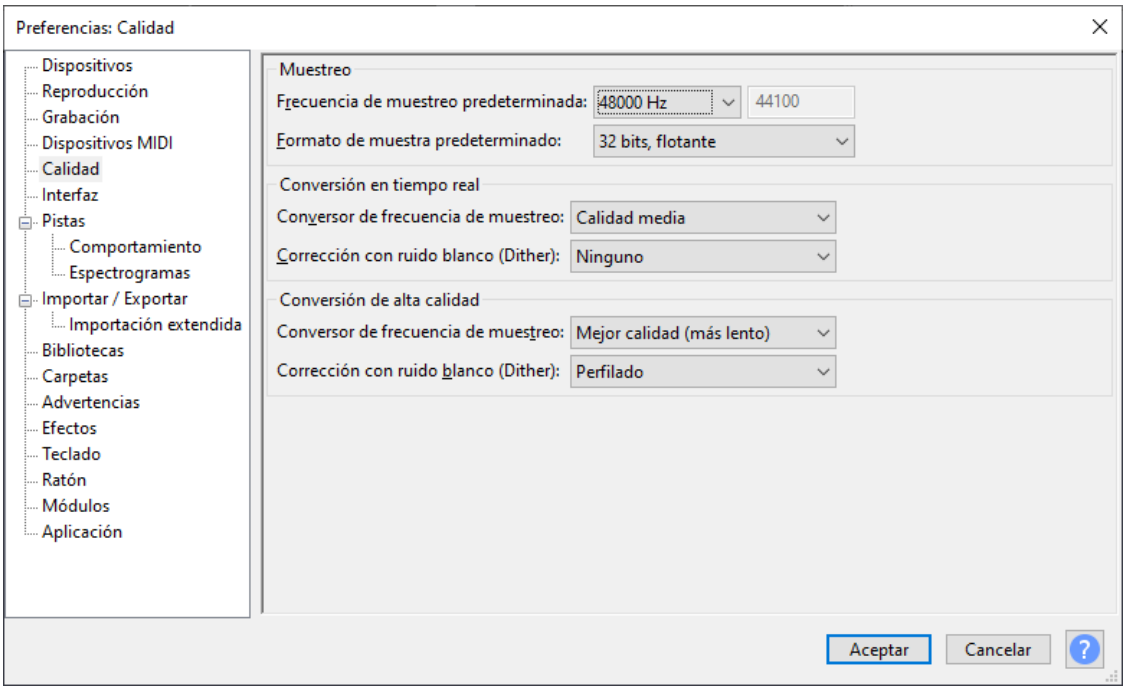


Figura 3.6 : Ventana de “Preferencias” en la categoría “Calidad”

A continuación, se presentan algunas de las frecuencias de muestreo comúnmente utilizadas en la grabación y reproducción de audio digital:

- 44.100 Hz: es la frecuencia de muestreo estándar utilizada en la producción de CD de audio. Es apropiada para la mayoría de los proyectos de grabación y edición de audio, y ofrece una calidad de sonido aceptable.
- 48.000 Hz: se utiliza en la producción de DVD y en proyectos profesionales de audio y vídeo. Ofrece una calidad de sonido ligeramente superior a la de 44.100 Hz y es una opción popular entre los profesionales de la producción de medios digitales y la edición de vídeo.
- 88.200 Hz y 96.000 Hz: se utilizan en proyectos de audio de alta resolución, como la producción musical, donde se busca capturar detalles más sutiles y precisos en las grabaciones. Estas frecuencias de muestreo generan archivos de audio de mayor tamaño y demandan mayor capacidad de almacenamiento y procesamiento, pero ofrecen una calidad de sonido excepcional.

En relación con el presente proyecto, el cual se enfoca en las grabaciones de voz humana, es fundamental tener en cuenta que el espectro de frecuencias audibles por el ser

humano se encuentra en un rango comprendido entre 20 Hz y 20.000 Hz. Por lo tanto, considerando estas premisas, el teorema de Whittaker-Nyquist-Kotelnikov-Shannon explicado anteriormente y las características del objeto de estudio, se ha decidido seleccionar una frecuencia de muestreo de 48.000 Hz para el presente trabajo. Esta elección garantiza una adecuada representación de las señales de voz humana, permitiendo su reconstrucción posterior sin pérdida de calidad e información relevante.

El formato de muestreo o profundidad de bits es un aspecto crítico que afecta la calidad del audio digital. La profundidad de bits se refiere a la cantidad de bits que se utilizan para representar cada muestra de audio en un archivo digital. Una mayor profundidad de bits permite una mayor resolución y calidad de sonido, así como un rango dinámico más amplio. El rango dinámico se define como la diferencia entre el nivel de sonido más bajo y el nivel de sonido más alto que un sistema de audio puede reproducir sin distorsión. El nivel de amplitud se relaciona directamente con la cantidad de diferentes niveles de intensidad sonora que se pueden representar en un sistema de audio digital. Por lo tanto, un mayor número de bits en la profundidad de bits resulta en una mayor cantidad de niveles de amplitud que se pueden representar, lo que a su vez conduce a un rango dinámico más amplio.

A continuación, se presentan las diferencias entre los formatos de muestreo de 16 bits, 24 bits y 32 bits en Audacity:

- 16 bits: Este formato de muestreo, que se utiliza en los CD de audio, es adecuado para la mayoría de los proyectos de grabación y edición de sonido. Con una profundidad de 16 bits, se puede alcanzar un rango dinámico de aproximadamente 96 dB, lo que equivale a $2^{16}$ = 65.536 niveles diferentes de amplitud. La calidad del audio es suficiente para la mayoría de los propósitos y es compatible con numerosos dispositivos y programas de reproducción de audio.
- 24 bits: Este formato de muestreo ofrece una resolución y calidad de audio superior al formato de 16 bits. Con una profundidad de 24 bits, se puede lograr un rango dinámico de aproximadamente 144 dB, lo que equivale a $2^{24}$ = 16.777.216 niveles distintos de amplitud. Esto permite capturar detalles más sutiles y un mayor contraste en las grabaciones de audio. Los proyectos que requieren una calidad de audio elevada o que implican una extensa edición y procesamiento de audio, como la producción musical y la postproducción de sonido, pueden beneficiarse del uso de un formato de muestreo de 24 bits.

- 32 bits: En ciertas versiones de Audacity, se puede encontrar una opción para seleccionar un formato de muestreo de 32 bits con punto flotante. Esta opción proporciona una resolución más alta y un rango dinámico prácticamente ilimitado, con una representación de $2^{32}$ = 4.294.967.296 niveles distintos de amplitud. El formato de 32 bits es útil para aplicaciones de audio profesional, como la mezcla y masterización de música, donde se requiere una calidad de audio excepcional y se realizan múltiples procesamientos de audio. Sin embargo, es importante tener en cuenta que los archivos de audio de 32 bits ocupan más espacio.

Al elegir el formato de muestreo predeterminado en Audacity, es fundamental tener en cuenta las necesidades específicas del proyecto y los estándares de calidad de audio requeridos. Para la mayoría de los proyectos, una profundidad de 16 bits resulta adecuada. Sin embargo, si se requiere una calidad de audio superior, se aconseja considerar el uso de un formato de muestreo de 24 o incluso 32 bits. En el caso particular de este trabajo, se ha optado por el formato de 32 bits, pues proporciona una calidad superior a pesar de que el archivo resultante sea de mayor tamaño. Además, dado que las grabaciones no son excesivamente largas, el tamaño del archivo no representa un inconveniente.

Una vez que se hayan configurado adecuadamente los dispositivos y la calidad de grabación en Audacity, es necesario seleccionar el botón “Aceptar” ubicado en la parte inferior de la ventana de Preferencias.

A continuación, se debe ajustar los niveles de grabación y monitorización. Para ello, se puede utilizar la barra de herramientas de Audacity, específicamente los controles deslizantes denominados “Nivel de grabación” y “Nivel de reproducción”, los cuales se encuentran junto a los iconos del micrófono y del altavoz, como se aprecia en la Figura 3.7.

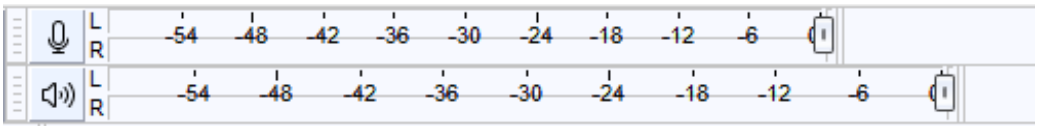


Figura 3.7 : Controles deslizantes “Nivel de grabación” y “Nivel de reproducción”, situados en la barra de herramientas

El primer control deslizante permite modificar el nivel de la señal de audio que se encuentra en proceso de grabación. Al desplazar el control hacia la izquierda se disminuirá la ganancia de entrada, mientras que al desplazarlo hacia la derecha se incrementará. Por otro lado, el segundo control deslizante facilita el ajuste del volumen de reproducción del proyecto, es decir, el nivel del sonido al escuchar las propias grabaciones y ediciones. Al

desplazar el control hacia la izquierda se disminuirá el volumen de reproducción, mientras que al desplazarlo hacia la derecha se incrementará.

Durante el proceso de grabación y reproducción, es esencial garantizar que los medidores exhiban niveles apropiados para prevenir distorsiones y asegurar una grabación de alta calidad. Los niveles óptimos se encuentran en el rango verde o amarillo del medidor, evitando así el rango rojo, el cual indica distorsión o recorte.

Una vez que se haya configurado adecuadamente todos los aspectos mencionados previamente, se podrá proceder con las grabaciones de audio. Para iniciar la grabación de audio, basta con hacer clic en el botón “Grabar” presente en la barra de herramientas, representado por un círculo de color rojo. Durante la grabación, es posible supervisar los niveles de audio mediante los medidores de nivel mencionados anteriormente. Para detener la grabación simplemente se deberá presionar el botón “Detener”, representado por un cuadrado de color negro. Estos botones se pueden ver en la Figura 3.8.

Figura 3.8 : Botón “Grabar” representado por un círculo de color rojo y botón “Detener” representado por un cuadrado de color negro

En el caso de que se deba realizar la edición de un archivo de audio previamente grabado, es necesario importar el archivo correspondiente. Para llevar a cabo esta acción, existen dos métodos disponibles:

Método 1: Arrastrar y soltar. Haciendo clic en el archivo de audio y manteniendo presionado el botón izquierdo del ratón, se debe desplazar el archivo hacia la ventana de Audacity. Posteriormente, al soltar el botón del ratón, se importará automáticamente el archivo de audio.

Método 2: Usar la opción “Importar” del menú “Archivo” de Audacity. Al hacer clic en “Archivo” en la esquina superior izquierda de la ventana de Audacity, se desplegará el menú correspondiente. Luego, se debe seleccionar “Importar” en el menú desplegado y, posteriormente, se debe elegir “Audio” en el submenú. A continuación, se mostrará un cuadro de diálogo para explorar los archivos. En él, se debe buscar y seleccionar el archivo o los archivos de audio que se deseen importar. Como se mencionó anteriormente, es posible importar una amplia variedad de archivos de audio e incluso extraer archivos de audio a partir de archivos de vídeo.

#### 3.2.4.1. Efectos

Una vez que se haya grabado el audio o haya sido importado, será posible emplear las diversas herramientas y efectos que el programa nos ofrece para editar y mejorar la grabación. Para ello, se debe seleccionar la herramienta que se desea utilizar y aplicarla en la pista de audio dentro de la ventana principal de Audacity.

En el presente trabajo se ha implementado la siguiente secuencia de efectos de manera consecutiva en casi todos los audios.

##### 3.2.4.1.1. Eliminación de ruido

El efecto de eliminación de ruido emplea un algoritmo sofisticado para detectar y suprimir el ruido de fondo, como el zumbido eléctrico, el ruido generado por ventiladores, el sonido del aire acondicionado y otros elementos. A continuación, se detallan los pasos para implementar este efecto.

En primer lugar, se debe escuchar el archivo de audio y localizar una sección que contenga el ruido de fondo, sin voz ni otros sonidos que se deseen conservar. Con la herramienta de selección habilitada, se debe utilizar el ratón para seleccionar dicha sección de ruido de fondo. La selección debe ser lo suficientemente extensa para permitir que Audacity pueda analizar y reconocer el ruido.

Posteriormente, se debe acceder a la barra de menú ubicada en la parte superior de la ventana de Audacity y seleccionar “Efecto” > “Reducción de ruido y reparación” > “Reducción de ruido...”.

A continuación, aparecerá la ventana de “Reducción de ruido”. Esta ventana aparece en la Figura 3.9. Dentro de ella, se debe hacer clic en el botón “Obtener perfil de ruido”. Al realizar esta acción, Audacity analizará las características y frecuencias específicas del ruido presente.

Luego, se debe seleccionar la totalidad de la pista de audio bien con el cursor del ratón o utilizando las teclas Ctrl+A o Cmd+A como atajo. Después, se debe regresar a la barra de menú y seleccionar nuevamente “Efecto” > “Reducción de ruido y reparación” > “Reducción de ruido…”. La ventana de Reducción de ruido se mostrará de nuevo.

Finalmente, se deben ajustar los controles deslizantes “Reducción de ruido (dB)”, “Sensibilidad” y “Suavizado de frecuencia (bandas)”. Se pueden utilizar los valores predeterminados y ajustarlos según los resultados deseados. Para facilitar la comprensión del lector, a continuación se describen las funciones de cada uno de estos controles:

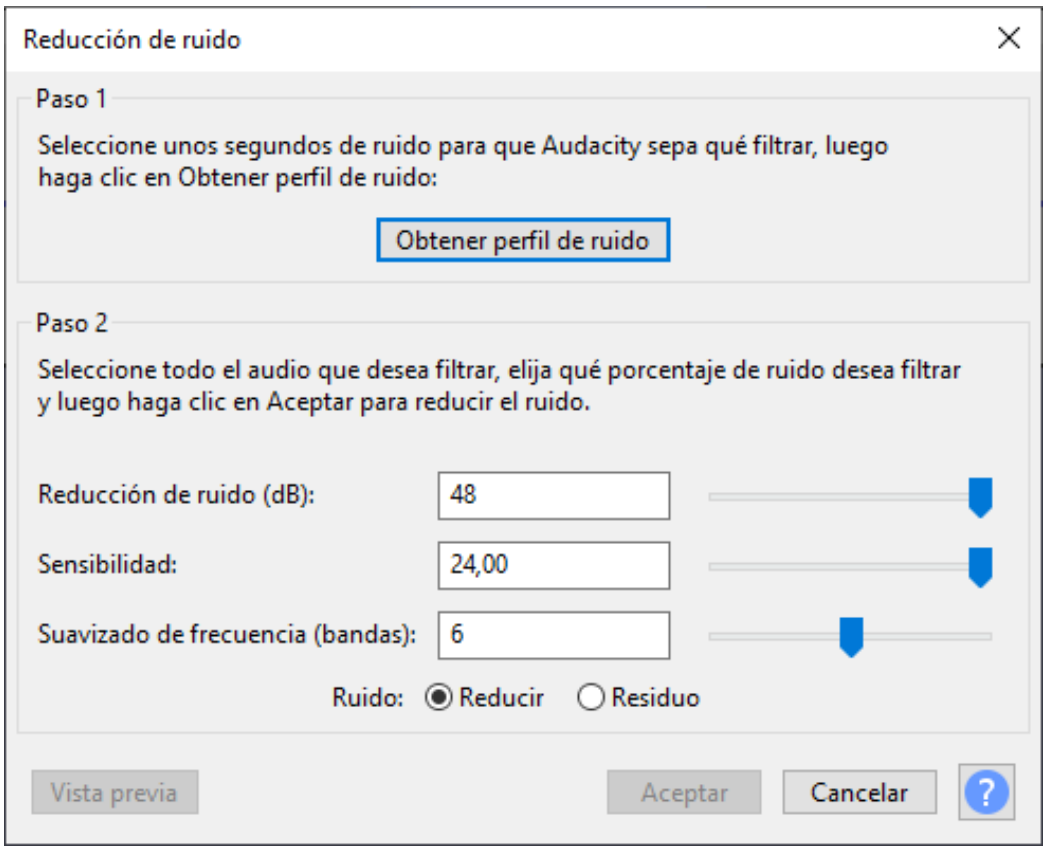


Figura 3.9 : Ventana del efecto “Reducción de ruido”

- Reducción de ruido (dB): Este control ajusta la cantidad de reducción de ruido que se aplicará al audio. Se mide en decibelios (dB), y valores más altos indican una mayor reducción de ruido. Sin embargo, aumentar demasiado este valor puede generar un sonido artificial o eliminar partes importantes del audio original. Por lo tanto, es importante encontrar un equilibrio entre eliminar el ruido de fondo y mantener la calidad del sonido.
- Sensibilidad: Este control ajusta el nivel con el que el programa buscará y eliminará el ruido. Un valor de sensibilidad más bajo indica que será más conservador al eliminar el ruido, mientras que un valor más alto hará que elimine más ruido, pero también pueda afectar al sonido original.
- Suavizado de frecuencia (bandas): Este control ajusta el número de bandas de frecuencia que se utilizan al aplicar la reducción de ruido. Un mayor número de bandas significa que el programa aplicará el efecto de reducción de ruido de manera más selectiva a diferentes frecuencias, lo que puede resultar en una reducción de ruido más precisa. Sin embargo, aumentar demasiado el número de bandas puede generar distorsiones en el sonido no deseadas. En general, un valor de suavizado de frecuencia más bajo se traduce en una reducción de ruido más suave y natural.

Si se encuentra seleccionada la pestaña “Reducir” y se hace clic en “Vista previa”, se podrá escuchar el sonido del audio procesado antes de aplicar el efecto. Por otro lado, si se tiene seleccionada la pestaña “Residuo”, se reproducirá el contenido del audio que será eliminado. En el primer caso, es esencial asegurarse de que el audio tenga una calidad adecuada, mientras que, en el segundo caso, es importante verificar que la voz no se elimine y que únicamente se escuche el ruido de fondo durante la vista previa.

Una vez que se esté satisfecho con el resultado, se debe hacer clic en “Aceptar” para aplicar la reducción de ruido a la pista de audio completa.

Es relevante tener en cuenta que, si bien la herramienta de Reducción de ruido es valiosa, no eliminará por completo el ruido de fondo en todos los casos, especialmente si el ruido es muy variable.

#### 3.2.4.1.2. Compresión

El compresor en Audacity es un efecto que contribuye al control y equilibrio de las variaciones de volumen en una grabación de audio. Su función es reducir el rango dinámico, es decir, la diferencia entre las partes más fuertes y las más suaves de una grabación, permitiendo un sonido más uniforme y una mayor inteligibilidad. A continuación, se detallan los pasos para emplear el efecto compresor.

En primer lugar, se debe seleccionar la pista o la parte de la pista de audio a la que se desee aplicar el efecto de compresión. Posteriormente, se debe acceder a la barra de menú en la parte superior de la ventana de Audacity y hacer clic en “Efecto” > “Volumen y comprensión” > “Compresor…”. Entonces se mostrará la ventana del “Compresor” con diversas opciones y controles deslizantes. Esta ventana aparece en la Figura 3.10.

A continuación, se enumeran algunos de los parámetros que pueden ajustarse, junto con los valores empleados para este trabajo:

- Umbral (*Threshold*): Corresponde al nivel de volumen (en dB) por encima del cual se aplicará la compresión. Los sonidos más fuertes que el umbral se comprimirán, mientras que los sonidos más suaves no se verán afectados. Se ha establecido en -14 dB.
- Límite inferior de ruido: Es el nivel base por debajo del cual el compresor modifica la ganancia del audio para prevenir la amplificación excesiva durante el procesamiento. Esta función resulta especialmente útil al comprimir el habla, ya que evita que la ganancia se incremente en las pausas, lo que podría provocar una amplificación exagerada del ruido de fondo. Se ha establecido en -50 dB.

- Proporción (*Ratio*): Es la cantidad de compresión que se aplicará a los sonidos que superen el umbral. Si se desea una compresión ligera y sutil, se utiliza una proporción baja, como 2:1 o 3:1. Esto significa que, por cada 2 o 3 dB que el volumen del audio exceda el umbral, se reducirá en 1 dB. En cambio, si se necesita un mayor control sobre las variaciones de volumen, se puede utilizar una proporción moderada, como 4:1 o 6:1, lo que resultará en una compresión más notable. En situaciones donde se requiere un control aún mayor sobre el rango dinámico, se puede utilizar una proporción alta, como 8:1, 10:1 o incluso más. Hay que tener en cuenta que las proporciones más altas pueden resultar en un sonido más procesado o "aplastado" y pueden afectar la calidad general del audio. Por lo general, estas últimas proporciones se suelen utilizar en contextos específicos, como en la producción de música electrónica o en la masterización. Se ha establecido en 4:1.

- Tiempo de ataque (*Attack Time*): Es el tiempo que tarda el compresor en aplicar la compresión una vez que el sonido supera el umbral. Un tiempo de ataque más corto resulta en una compresión más rápida. Se ha establecido en 0.21 s.

- Tiempo de decaimiento (*Release Time*): Es el tiempo que tarda el compresor en dejar de aplicar la compresión después de que el sonido cae por debajo del umbral. Se ha establecido en 1 s.

- Pestaña de composición de ganancia para 0 dB tras compresión: amplifica el audio resultante en todas las pistas seleccionadas después de la compresión hasta un nivel máximo de 0 dB. Todas las pistas se amplifican en la misma cantidad que en el efecto amplificar. En caso de que se vayan a realizar ajustes posteriores de ganancia en los que esta se vaya a aumentar, como puede ser el caso de un ajuste de ecualización, no es recomendable marcarla, pudiendo llevar a cabo una posterior amplificación del volumen si eso se ve necesario. Se ha dejado desmarcada.

- Pestaña compresión basada en picos: Esta opción permite cambiar la forma en que el compresor responde a los picos en el audio. La compresión basada en picos se enfoca en los picos transitorios del audio, es decir, aquellos momentos breves de alta intensidad que pueden sobresalir por encima del resto del audio. En cambio, la compresión RMS (*Root Mean Square*) se enfoca en el volumen promedio del audio. Se ha optado por utilizar la compresión RMS para este trabajo, ya que resulta más adecuada para la mayoría de las grabaciones de audio. Es decir, se ha mantenido desmarcada

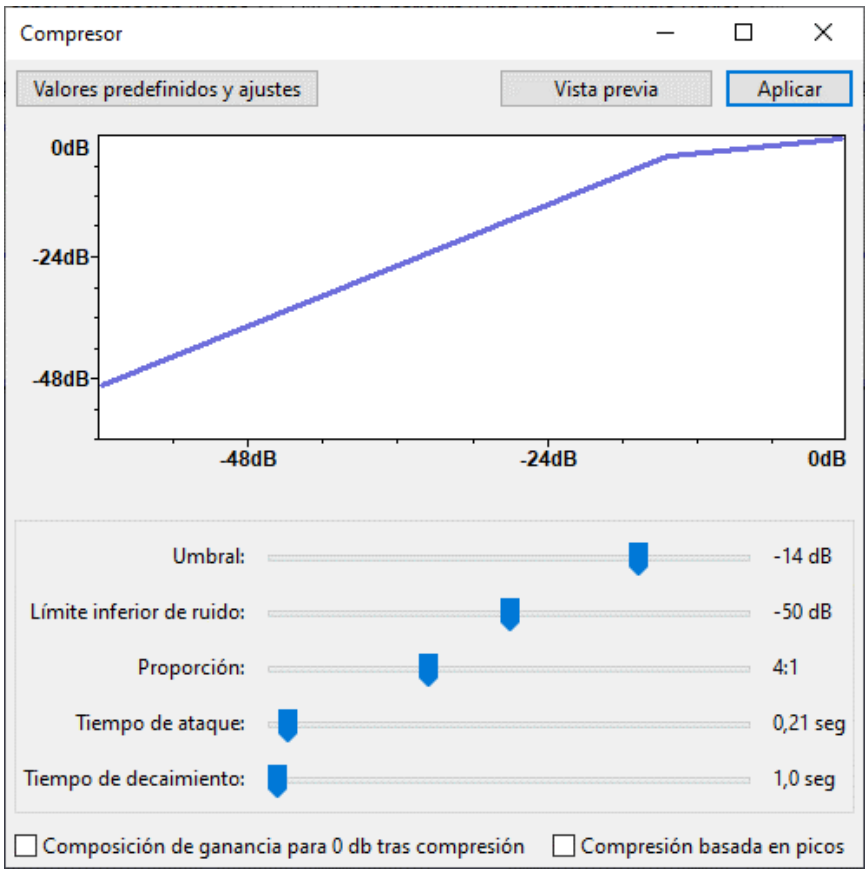


Figura 3.10 : Ventana del efecto “Compresor”

Una vez que se han ajustado los parámetros del efecto de compresión de manera adecuada, se debe hacer clic en el botón “Aceptar” con el objetivo de aplicar el efecto a la selección de audio previamente realizada.

#### 3.2.4.1.3. Ecualización

Para realizar la ecualización de grabaciones de audio, se empleará el efecto denominado ecualizador de curva de filtro. Esta herramienta de ecualización utiliza un filtro de Transformada Rápida de Fourier (FFT).

La FFT se basa en la Transformada de Fourier, que es una operación matemática que permite analizar la señal de audio en el dominio de la frecuencia. La Transformada de Fourier Discreta (DFT) es una versión discreta que se utiliza para analizar señales digitales. La DFT transforma una señal discreta en el dominio del tiempo en su representación discreta en el dominio de la frecuencia. Esta operación produce una serie de coeficientes complejos que representan la amplitud y la fase de las diferentes frecuencias presentes en la señal de audio. La FFT utiliza una versión optimizada de la DFT conocida como “algoritmo de Cooley-Tukey” para calcular los coeficientes de la DFT de manera eficiente. Este algoritmo divide la señal de audio en segmentos más pequeños y aplica la DFT a cada segmento por separado, lo que reduce el número de operaciones matemáticas necesarias

para realizar el cálculo completo. El resultado del FFT es una representación en el dominio de la frecuencia de la señal de audio, lo que permite aplicar diferentes filtros para modificar la amplitud de las diferentes frecuencias presentes en la señal.

A continuación, se presentan los pasos necesarios para utilizar este ecualizador en el programa Audacity.

En primer lugar, se debe seleccionar la sección de audio que se desea ecualizar. Posteriormente, se debe acceder al efecto Ecualizador de curva de filtro a través de “Efecto” > “EQ y filtros” > “Ecualizador de curva de filtro…”. Al realizar esta acción, se desplegará la ventana del “Ecualizador de curva de filtro”, donde se podrá visualizar una representación gráfica de las frecuencias y una línea plana que simboliza la curva de ecualización. Esta ventana se muestra en la Figura 3.11.

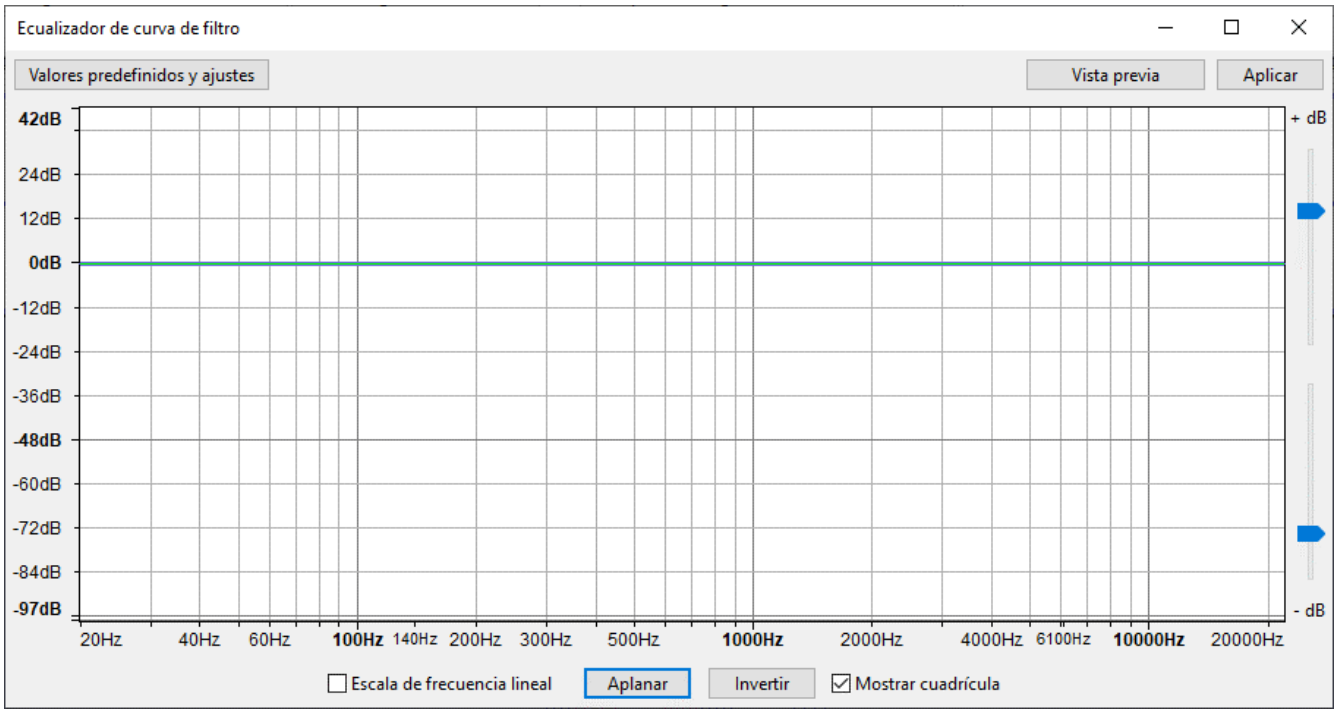


Figura 3.11 : Ventana del efecto “Ecualizador de curva de filtro”

Para comenzar el proceso de ecualización, se debe hacer clic en la línea plana para añadir un punto de control. Es posible agregar tantos puntos de control como se considere necesario. Seguidamente, se deben desplazar dichos puntos de control hacia arriba o hacia abajo para incrementar o reducir las frecuencias en el área específica. Al modificar la posición de un punto de control, se variará la forma de la curva de ecualización, lo cual repercutirá en las frecuencias del audio.

En caso de que se desee eliminar un punto de control, simplemente se debe arrastrar fuera del gráfico. También es posible emplear una de las configuraciones preestablecidas de ecualización. Para ello, se debe presionar en el menú desplegable “Valores predefinidos y ajustes” ubicado en la parte superior izquierda de la ventana y seleccionar una de las opciones disponibles.

Por último, se debe hacer clic en “Aplicar” para implementar el efecto en el audio seleccionado. Asimismo, se puede optar por utilizar la función de vista previa antes de aplicarlo.

Ecualizar la voz humana es un proceso esencial para lograr un sonido claro y equilibrado. Es fundamental confiar en nuestros oídos y experimentar con diferentes ajustes hasta encontrar el sonido deseado. Además, se debe utilizar un buen par de monitores de estudio o audífonos para asegurarnos de que estamos escuchando el sonido con precisión. A continuación, se presentan algunas recomendaciones generales para ecualizar la voz humana, sin embargo, es importante tener en cuenta que cada voz es única y puede requerir ajustes específicos según las características individuales y el contexto:

- Aplicar un filtro pasa-alto (*high-pass filter*) para eliminar las frecuencias bajas, generalmente por debajo de 80-100 Hz. Estas frecuencias pueden contener ruido de baja frecuencia, como el zumbido de los equipos electrónicos o las vibraciones del micrófono.
- Si la voz suena muy cercana o *boomy*, se deben reducir las frecuencias en el rango de 100-250 Hz. Esto puede ayudar a eliminar el efecto de proximidad causado por estar muy cerca del micrófono.
- Aumentar las frecuencias en el rango de 1-4 kHz mejora la claridad y la inteligibilidad de la voz. Hay que tener cuidado de no exagerar estos ajustes, ya que puede causar un sonido áspero o nasal.
- Aumentar las frecuencias en el rango de 4-8 kHz puede hacer que la voz suene más aireada y destacada en la mezcla.
- Si la voz tiene demasiadas “s” y “sh” pronunciadas, se recomienda reducir las frecuencias en el rango de 5-8 kHz para controlar las sibilancias.
- Aumentar las frecuencias por encima de 10 kHz puede darle un toque adicional de brillo a la voz, agregando una sensación de aire y apertura.

Al ecualizar, es útil aplicar primero cortes en las frecuencias problemáticas y luego reforzar las áreas donde la voz necesita más presencia o brillo. Esto puede ayudar a mantener un sonido más natural y equilibrado.

#### 3.2.4.1.4. Normalización

La normalización ajusta el volumen de una señal para que alcance un nivel objetivo preestablecido. El objetivo de la normalización es hacer que diferentes pistas de audio o segmentos dentro de una pista tengan un nivel de volumen similar, lo que facilita la escucha y evita cambios abruptos de volumen entre diferentes secciones de un archivo. Existen dos enfoques principales para normalizar el sonido:

- Normalización de pico: Este enfoque ajusta el volumen de la señal de audio de manera que el nivel de volumen más alto (pico) alcance un nivel objetivo específico. Por ejemplo, si el nivel objetivo es -1 dB, la normalización de pico aumentará o disminuirá el volumen de toda la pista de audio hasta que el pico más alto de la señal llegue a -1 dB. Esto ayuda a evitar la distorsión y el recorte, pero no necesariamente iguala el volumen aparente de diferentes pistas.
- Normalización de *loudness* (sonoridad): Este enfoque se basa en la percepción humana del volumen y ajusta el volumen de la señal de audio de acuerdo con un nivel objetivo de *loudness*, medido en unidades como LUFS (*Loudness Units Full Scale*) o LKFS (*Loudness, K-weighted, relative to Full Scale*). Este tipo de normalización hace que diferentes pistas de audio o segmentos dentro de una pista tengan un nivel de volumen similar en términos de cómo los perciben los oyentes.

En el presente trabajo, se ha aplicado la normalización de pico como método de normalización de sonido. A continuación, se detallan los pasos necesarios para aplicar la normalización de pico a un archivo de audio utilizando el programa de edición de audio Audacity.

En primer lugar, se debe seleccionar la pista o parte de la pista que se desea normalizar. A continuación, en la barra de menú ubicada en la parte superior de la ventana de Audacity, se selecciona la opción “Efecto” > “Volumen y compresión” > “Normalizar…”. Al realizar esta acción, se abrirá una ventana con opciones de normalización. Esta ventana aparece en la Figura 3.12. Dentro de la ventana “Normalizar”, se encuentran algunas opciones ajustables.

Es fundamental asegurarse de que la opción “Eliminar desplazamiento DC” se encuentre seleccionada, ya que se utiliza para eliminar un desplazamiento de corriente continua (DC) que pueda estar presente en la señal de audio. Este desplazamiento consiste en un cambio constante en la línea de base (nivel 0) de una señal de audio. Idealmente, la línea de base debería encontrarse en 0, y las ondas de sonido deberían oscilar de manera simétrica por encima y por debajo de este punto. Sin embargo, debido a problemas en el proceso de grabación o a imperfecciones en el *hardware*, en ocasiones la línea de base se desplaza, lo que puede afectar la calidad del sonido y causar distorsiones. Al marcar la opción “Eliminar desplazamiento DC” en Audacity y aplicar el efecto de normalización, el programa analiza la señal de audio y calcula el valor promedio de desplazamiento de DC. A continuación, Aresta este valor promedio de toda la señal de audio, resultando en una línea de base corregida en 0.

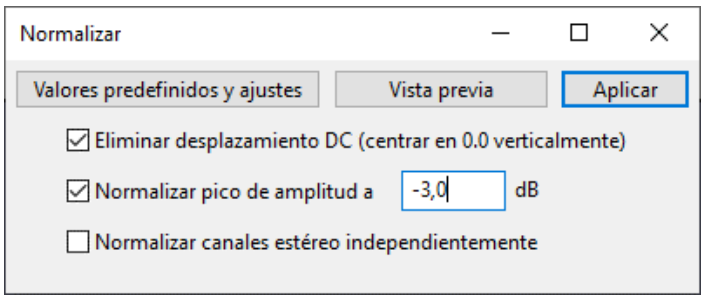


Figura 3.12 : Ventana del efecto “Normalizar”

En la opción “Normalizar pico de amplitud a”, inicialmente es necesario marcar la casilla correspondiente para poder introducir el valor de amplitud máxima deseado para la selección que se pretende procesar. La configuración predeterminada es -1 dB, si bien se puede ajustar según las preferencias del usuario. Un nivel de -1 dB se sitúa ligeramente por debajo de la máxima amplitud sin recorte (0 dB), lo cual facilita la aplicación de efectos y una reproducción exenta de distorsiones en distintos dispositivos. Es factible emplear valores más negativos, como -3 dB, con el propósito de normalizar a amplitudes inferiores en caso de que se pretenda aplicar efectos que incrementen el sonido después de la normalización, evitando así la saturación de la señal. Cabe mencionar que no se aceptan valores positivos, y los botones “Aplicar” y “Vista previa” quedarán inactivos si se introdujeran. En el supuesto de que se necesite incrementar la amplitud más allá del umbral de recorte de 0 dB, resulta conveniente utilizar otro efecto, como el de “Amplificar”. Para el trabajo en cuestión, se ha optado por valores de pico entre -1 dB y -3 dB.

Por último, una vez que se han configurado las opciones de normalización, es necesario presionar el botón “Aplicar” para llevar a cabo la normalización de pico en el

archivo de audio. En este sentido, Audacity ajustará el volumen de la pista de audio seleccionada de acuerdo con el nivel de pico establecido.

#### 3.2.4.2. Exportación

Una vez completada la edición del audio, se requiere exportarlo en uno de los formatos disponibles que ofrece Audacity. Para ello, es necesario acceder a la barra de menú ubicada en la parte superior de la ventana de Audacity y seleccionar la opción “Archivo” > “Exportar”. En ese momento, se presentarán diversas opciones de formato de archivo, tales como “Exportar como WAV”, “Exportar como MP3” u otros formatos disponibles. Audacity es un programa de edición de audio de código abierto que cuenta con una amplia gama de formatos de exportación. A continuación, se detallan algunos de los formatos que este programa nos ofrece.

- WAV (*Waveform Audio File Format*): Se trata de un formato sin pérdida de calidad que preserva la fidelidad del audio original. Es ampliamente utilizado en entornos profesionales y académicos.
- MP3 (MPEG-1 *Audio Layer 3*): A pesar de ser un formato con pérdida de calidad, el MP3 es una elección común debido a su compatibilidad y tamaño de archivo reducido en comparación con formatos sin pérdida. Para trabajos académicos, se recomienda utilizar la tasa de bits más alta posible (320 kbps) para minimizar la pérdida de calidad.
- OGG (Ogg Vorbis): Es un formato de compresión con pérdida de calidad, pero con una eficiencia de compresión generalmente mejor que la del MP3. OGG Vorbis es una opción adecuada si se busca un formato libre de patentes y de código abierto.

Al seleccionar un formato de exportación para trabajos académicos, es esencial tener en cuenta las necesidades de calidad de audio, el tamaño del archivo y la compatibilidad con otros dispositivos o programas. WAV es la mejor opción si la calidad del audio es la prioridad principal, mientras que MP3 y OGG pueden ser más adecuados si se necesita un equilibrio entre calidad y tamaño del archivo. En este trabajo se ha seleccionado el formato WAV debido a que es un formato sin pérdida y es compatible con Microsoft PowerPoint. Aunque es cierto que este formato puede resultar en archivos de gran tamaño, se ha determinado que es la elección ideal para este proyecto en particular, dado que los vídeos producidos tienen una duración media de 10 minutos, y por lo tanto el tamaño del archivo no será un problema.

## 3.3. OBS (Open Broadcaster Studio)

OBS [42] es un *software* de código abierto que brinda a los usuarios una amplia variedad de funciones y opciones para grabar y transmitir en vivo contenido multimedia. Algunas de las características principales y ventajas de este programa incluyen:

- Permite a los usuarios crear y personalizar múltiples escenas, cada una con diferentes fuentes de vídeo y audio. Las fuentes pueden incluir cámaras web, capturas de pantalla, imágenes, vídeos, texto, navegadores web y otros elementos. Los usuarios pueden cambiar fácilmente entre escenas en tiempo real durante la transmisión o grabación.
- Posibilita a los usuarios personalizar la configuración de salida de vídeo y audio, incluida la resolución, la velocidad de bits, los códecs y el formato de archivo. Esto permite a los creadores ajustar la calidad y el tamaño de su contenido según las necesidades y limitaciones de ancho de banda de su audiencia.
- Incluye una variedad de filtros y efectos para mejorar la calidad del vídeo y el audio, como la reducción de ruido, la corrección de color y la eliminación de fondo mediante la técnica de *chroma key* (pantalla verde). También hay una amplia variedad de complementos desarrollados por la comunidad que amplían aún más las capacidades de OBS.
- Se integra fácilmente con una variedad de servicios de transmisión en vivo populares, como Twitch, YouTube y Facebook, mediante la configuración de claves de transmisión y otros ajustes específicos del servicio.
- Ofrece un modo estudio que permite a los usuarios hacer ajustes y previsualizar sus cambios en tiempo real antes de aplicarlos a la transmisión en vivo. Esto puede ser útil para hacer ajustes de última hora y garantizar que todo funcione correctamente antes de comenzar a transmitir.
- Cuenta con una comunidad activa de desarrolladores y usuarios que brindan soporte, actualizaciones y recursos adicionales. Hay una gran cantidad de tutoriales, foros y guías disponibles en línea para ayudar a los nuevos usuarios a aprender a utilizar el programa y resolver problemas comunes. Todo esto es debido a su condición de código abierto.

Podemos concluir esta introducción diciendo que OBS se destaca como una solución sólida y versátil para la grabación y emisión en directo de material multimedia. Gracias a su

extenso conjunto de funcionalidades, posibilidades de personalización y respaldo comunitario, se ha vuelto una opción predilecta para creadores de contenido, jugadores que realizan transmisiones en vivo, educadores y otros expertos que requieren una herramienta fiable.

### 3.3.1. OBS para vídeos educativos

En esta sección, se abordará el uso de OBS para la producción de vídeos educativos con estándares de alta calidad. Para ello, es necesario seguir rigurosamente los siguientes pasos.

Como puede verse en la Figura 3.13, los elementos principales que conforman OBS son: la ventana de vista previa, la lista de escenas, la lista de fuentes, el mezclador de audio, y los controles de grabación y transmisión en vivo.

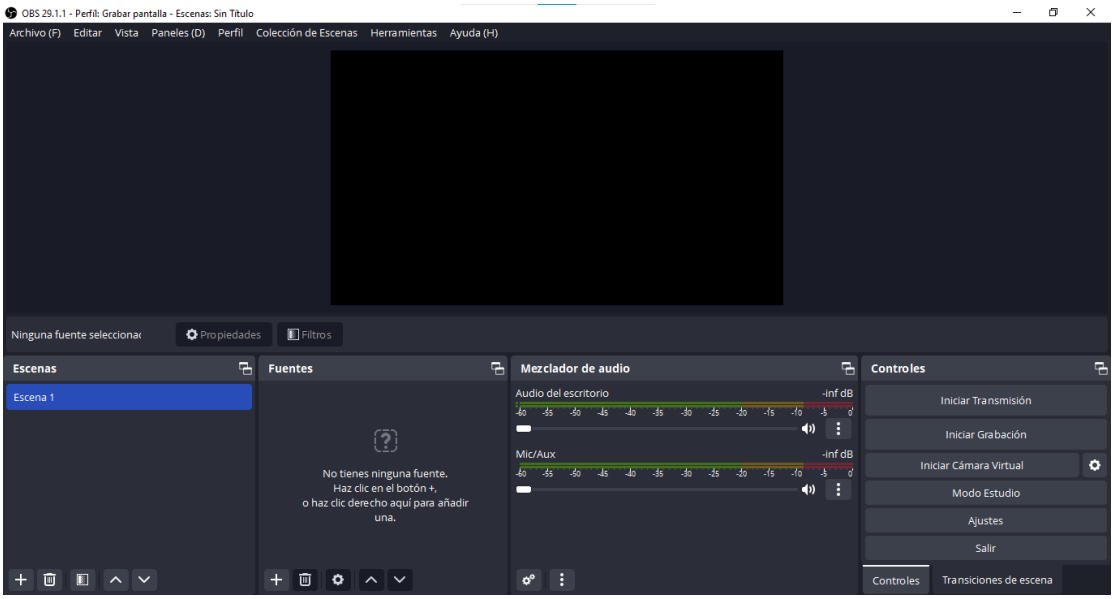


Figura 3.13 : Ventana principal de la interfaz de usuario de OBS

El siguiente paso es configurar las escenas. Cabe mencionar que una escena se trata de una composición que agrega múltiples fuentes de vídeo y audio, las cuales se combinan para generar una imagen final que se visualizará en la transmisión en vivo o en la grabación. Las escenas permiten que los usuarios diseñen y personalicen la apariencia de su contenido, pudiendo cambiar con facilidad entre distintas disposiciones y configuraciones de fuentes. Por ejemplo, en el contexto de un tutorial de programación, se podría crear varias escenas que incluyan:

- La escena de introducción, la cual puede mostrar una imagen de título o un vídeo de apertura acompañado de música de fondo.

- La escena principal, la cual puede presentar la captura de pantalla del entorno de programación (por ejemplo, MATLAB), así como una pequeña ventana de la cámara web para mostrar el rostro del presentador y un micrófono para grabar la voz.
- La escena de pausa, la cual puede mostrar un mensaje de "*Estamos en pausa*" con música de fondo, útil para momentos en los que se necesite tomar un descanso.
- La escena de despedida, la cual puede incluir una imagen de cierre o un vídeo de agradecimiento, así como información para contactar al presentador o seguir su contenido.

La configuración de escenas en OBS permite cambiar con agilidad entre ellas durante una transmisión en vivo o grabación, mediante la selección de la escena deseada en la lista correspondiente. La gestión y creación de escenas se realiza a través del panel "Escenas", ubicado en la esquina inferior izquierda de la ventana principal. Es importante destacar que el programa genera de manera predeterminada la "Escena 1".

Para agregar una nueva escena, es necesario hacer clic en el símbolo "+" que se encuentra en la parte inferior del panel, seguido de asignar un nombre único que no se encuentre en uso por otra escena. Finalmente, se deberá hacer clic en el botón "Aceptar" para guardar los cambios realizados.

Una vez creadas las escenas, es necesario configurar las fuentes correspondientes. Las fuentes pueden incluir capturas de pantalla, cámaras web, micrófonos, imágenes, vídeos, texto, entre otros elementos. Para agregar una fuente, se debe acceder al panel "Fuentes" ubicado en la parte inferior central de la ventana principal, justo al lado del panel "Escenas". Luego, se debe hacer clic en el símbolo "+" ubicado en la parte inferior del panel "Fuentes". Al hacerlo, se desplegará una lista que muestra los distintos tipos de fuentes disponibles en OBS. Algunas opciones comunes son:

- "Captura de pantalla" o "Captura de ventana": para mostrar una ventana o pantalla completa de la computadora.
- "Dispositivo de captura de vídeo": para mostrar una cámara web o dispositivo de vídeo externo.
- "Captura de entrada audio": para grabar audio desde un micrófono o dispositivo de entrada de audio.
- "Imagen": para agregar imágenes estáticas como logotipos o gráficos.
- "Texto (GDI+)": para agregar texto personalizado a la escena.

Una vez seleccionada la nueva fuente, se abrirá una ventana que permitirá crearla. En este espacio, se debe asignar un nombre descriptivo a la fuente en el cuadro “Crear nueva” y hacer clic en el botón “Aceptar”. Este procedimiento permitirá una fácil identificación de la fuente en etapas posteriores.

Luego de agregar la nueva fuente, se presentará una ventana de configuración que permitirá ajustar los parámetros específicos para el tipo de fuente que se ha añadido. En el presente trabajo se explicarán los procesos de añadir una fuente de “Captura de ventana” y de “Captura de entrada de audio”, pues son las únicas que se han usado.

En la configuración de la fuente de “Captura de ventana”, que aparece en la Figura 3.14, se selecciona la ventana que se desea capturar. Por ejemplo, se podría elegir la ventana del entorno de programación de MATLAB. Las demás opciones de configuración se dejan por defecto. En el caso particular de vídeos educativos sobre programación mediante MATLAB, es importante seleccionar la pestaña “Captura de Cursor”, a fin de permitir que el cursor se visualice durante la grabación. Esto resultará fundamental para que los alumnos puedan seguir de manera efectiva las explicaciones. Se recomienda usar un tamaño de cursor grande y de color bien visible, así como moverlo por la pantalla con precaución para no tapar información relevante ni despistar al público con movimientos al azar.

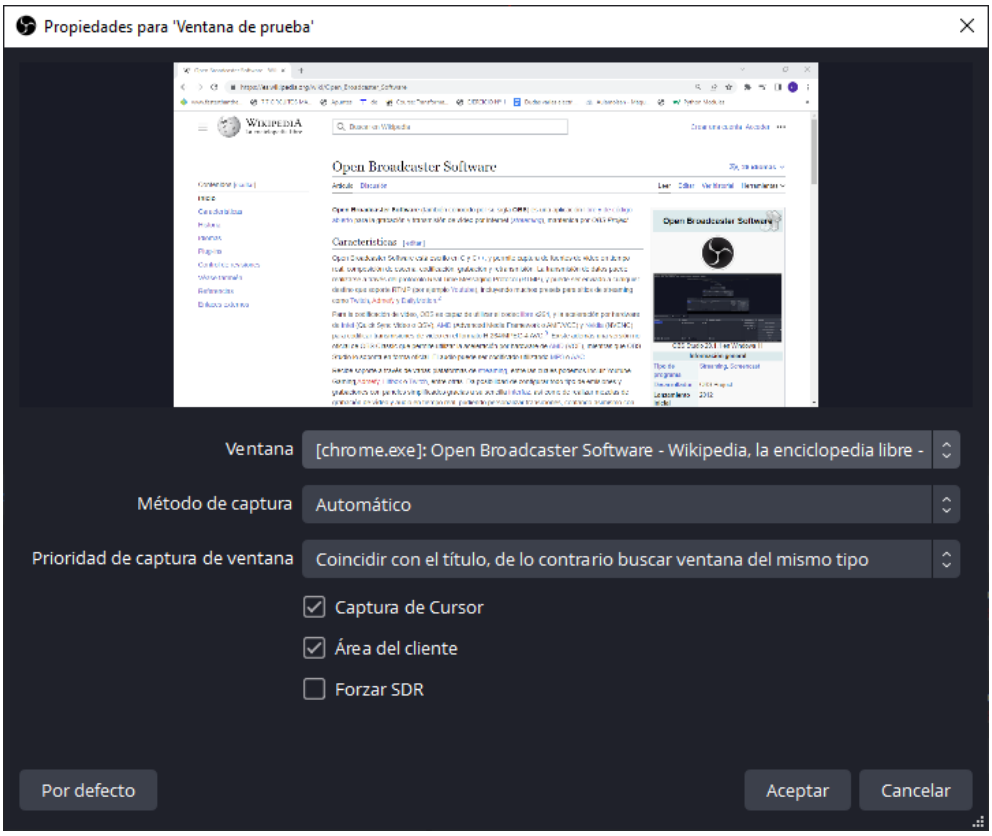


Figura 3.14 : Ventana de configuración de la fuente “Captura de ventana”

Una vez configuradas las fuentes, se debe hacer clic en “Aceptar” en la ventana de configuración de la fuente para agregarla a la escena correspondiente. La fuente aparecerá en la lista de fuentes del panel “Fuentes” y en la ventana de vista previa, si se trata de una fuente de vídeo o imagen.

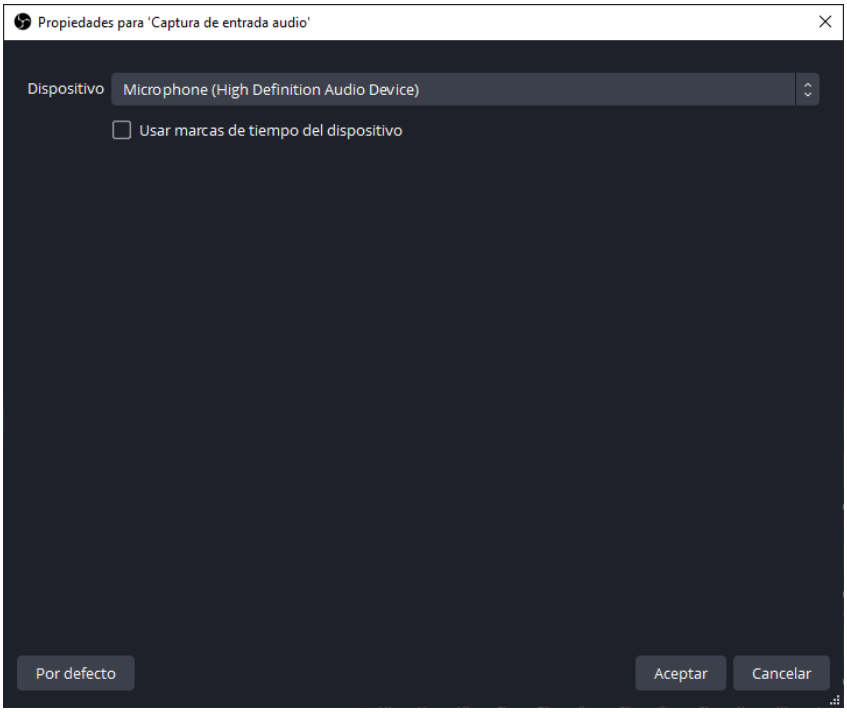


Figura 3.15 : Ventana de configuración de la fuente “Captura de entrada de audio”

Si es necesario, se pueden ajustar el tamaño y la posición de la fuente en la ventana de vista previa (en caso de tratarse de una fuente de vídeo o imagen), haciendo clic y arrastrando los bordes y esquinas de la fuente. Esta opción es útil, por ejemplo, para agregar un dispositivo de captura de vídeo, como la cámara web, y organizar su disposición en la vista previa, o para agregar un logotipo a la grabación y asegurarse de que aparezca en un lugar específico de la pantalla.

Asimismo, se pueden reorganizar el orden de prioridad de las capas de las fuentes en el panel “Fuentes”, arrastrando y soltando las fuentes en la lista. Por ejemplo, en caso de querer que un logotipo aparezca en la grabación, pero se encuentra oculto detrás de una fuente de captura de pantalla, se debe colocar el logotipo en una posición más alta en el orden de prioridad en la lista.

Una vez que se han creado las fuentes, se pueden agregar efectos a las mismas. En el presente trabajo, únicamente se han utilizado efectos de sonido. Los filtros son efectos que se pueden aplicar a las fuentes. En particular, los filtros de sonido son efectos que se aplican a las fuentes de audio, como micrófonos, grabaciones de audio y otros tipos de

entradas de audio para mejorar la calidad del audio, agregar efectos especiales o eliminar ruidos no deseados.

Para agregar un filtro a una fuente, se debe hacer clic derecho sobre ella en el panel “Fuentes” y, en el menú desplegable, seleccionar “Filtros”. Posteriormente, se abrirá una ventana que permitirá agregar los filtros que se deseen mediante el botón “+” que aparece en la parte inferior izquierda.

En el presente trabajo, se ha utilizado solamente el filtro de “Eliminación de ruido” debido a que la edición de audio se ha realizado mediante el *software* Audacity explicado en la sección anterior de este mismo capítulo. Este filtro permite utilizar dos algoritmos distintos para detectar y eliminar el ruido de fondo. Los algoritmos son Speex y RNNoise. Con el fin de mejorar la calidad de la grabación, se ha utilizado el algoritmo RNNoise, a pesar de que conlleva un mayor consumo de CPU. La ventana para agregar filtros a la fuentes de audio aparece en la Figura 3.16, con el filtro RNNoise aplicado.

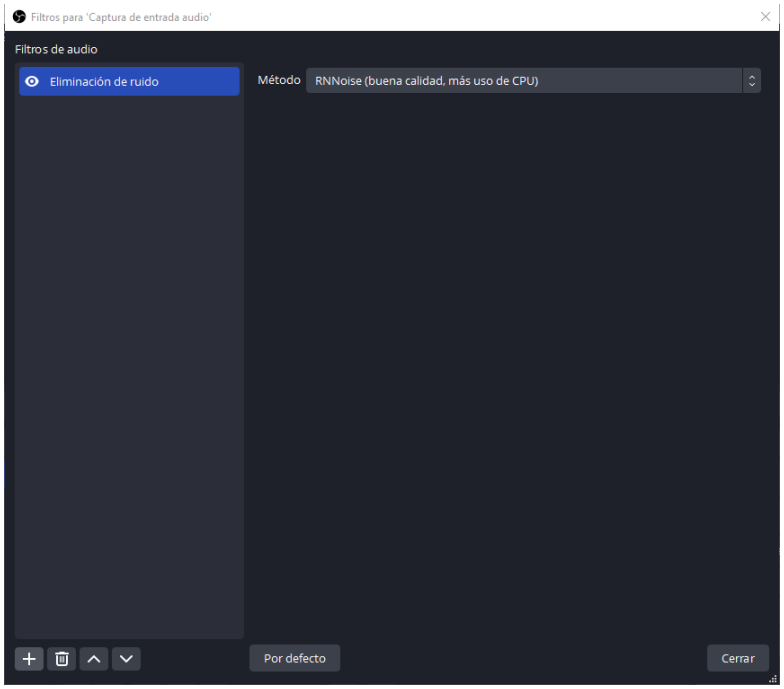


Figura 3.16 : Ventana de configuración de filtros para entradas de audio, con el filtro RNNoise aplicado

RNNoise es un algoritmo de aprendizaje profundo desarrollado para la reducción de ruido en grabaciones de audio, especialmente en señales de voz. Utiliza redes neuronales recurrentes (RNN) para analizar y procesar las señales de audio. Su arquitectura incluye capas de memoria de corto plazo (LSTM), que permiten que la red tenga en cuenta el contexto temporal de la señal y, de esta manera, identificar y eliminar mejor el ruido de fondo. El algoritmo RNNoise es altamente efectivo en la eliminación de ruido de tipo estacionario, como ruido de ventiladores o motores, así como en la eliminación de ruido no

estacionario, como conversaciones de fondo o ruido del tráfico. Al estar entrenado en una amplia variedad de ruidos y situaciones, es capaz de adaptarse a diferentes entornos y lograr una reducción de ruido significativa sin degradar la calidad de la voz. Una de las ventajas de RNNoise es su eficiencia computacional, lo que permite su implementación en tiempo real en dispositivos con recursos limitados, como teléfonos móviles o sistemas embebidos. También es de código abierto y está disponible para su uso y modificación por parte de desarrolladores e investigadores.

Una vez aplicados los filtros, se debe hacer clic en el botón “Cerrar” para cerrar la ventana de filtros y dejar aplicados todos los filtros que se hayan agregado.

El último paso será la configuración de los ajustes de grabación. Para ello, es necesario hacer clic en “Ajustes” ubicado en la esquina inferior derecha de la ventana principal. De esta forma, se abrirá una nueva ventana en la cual se debe seleccionar la pestaña “Salida”. Dentro de esta, aparecerá un menú desplegable denominado “Modo de salida”, el cual se debe seleccionar la opción “Avanzado” para obtener un mayor control y opciones de configuración de la grabación en comparación con el modo simple. A continuación, se mostrarán tres pestañas: “Transmisión”, “Grabación” y “Reproducción”. Se debe hacer clic en la pestaña “Grabación” para poder ajustar la configuración avanzada de grabación. Esta ventana con la sección “Grabación” seleccionada aparece en la Figura 3.17.

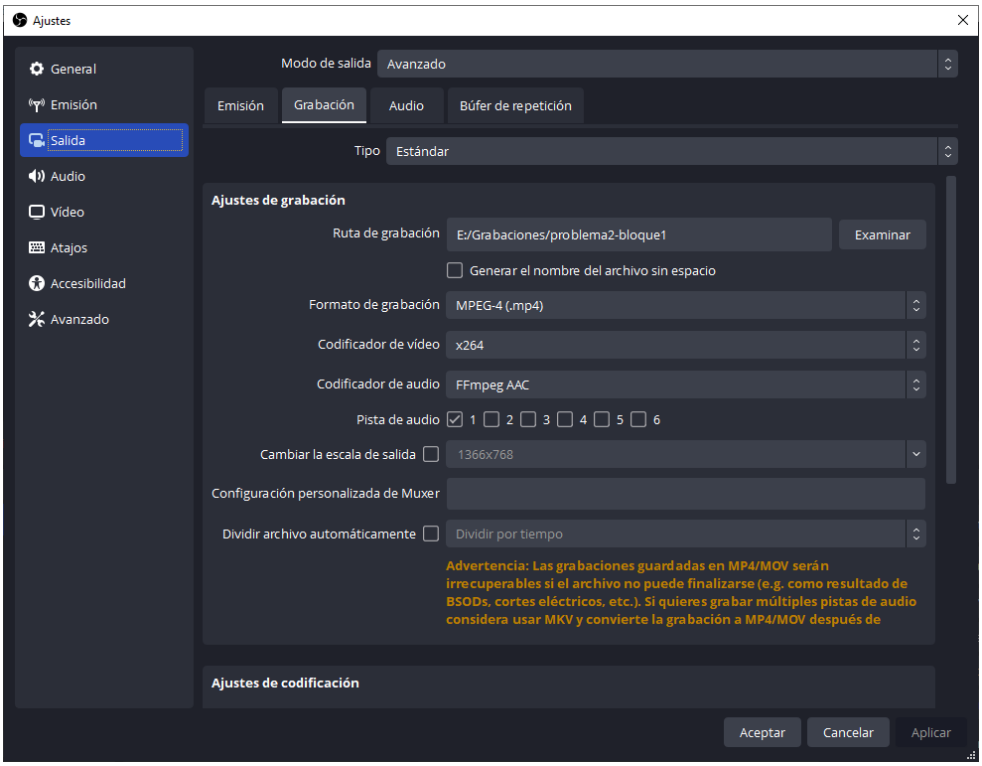


Figura 3.17 : Ventana de configuración de ajuste, con la pestaña “Salida” seleccionada en la sección “Grabación”

En primer lugar, es necesario configurar la “Ruta de grabación” seleccionando la carpeta donde se desea guardar los archivos de grabación. Posteriormente, se debe elegir el “Tipo de grabación”, el cual puede ser “Estándar”, para grabar en un solo archivo, o “Personalizado”, si se desea configurar una grabación multitrack con diferentes fuentes de audio en pistas separadas. En este trabajo se ha seleccionado la opción “Estándar”. Seguidamente, se debe elegir el formato en “Formato de grabación”. Algunos formatos populares incluyen MP4, MKV y FLV. A continuación, se presenta una breve descripción de algunos formatos populares:

- MP4 (MPEG-4 Part 14): es un formato de contenedor multimedia desarrollado por el Moving Picture Experts Group (MPEG). Es uno de los formatos más populares y ampliamente compatibles, ya que es compatible con la mayoría de los dispositivos y sistemas operativos. MP4 admite vídeo, audio, texto y metadatos, como subtítulos y descripciones. Además, utiliza una compresión eficiente para ofrecer una alta calidad de vídeo con un tamaño de archivo relativamente pequeño.
- MKV (Matroska Video): es un formato de archivo contenedor de código abierto que puede contener múltiples pistas de vídeo, audio, subtítulos y metadatos en un solo archivo. Es conocido por su flexibilidad y soporte para una amplia gama de códecs de vídeo y audio. Aunque no es tan ampliamente compatible como MP4, su capacidad para almacenar múltiples pistas de audio y subtítulos lo convierte en una opción popular para películas y programas de televisión.
- FLV (Flash Video): es un formato de archivo de vídeo desarrollado por Adobe Systems para su reproductor de Adobe Flash. Fue popular en la década de 2000 y principios de la década de 2010, especialmente para la transmisión de vídeo en línea. Sin embargo, con el declive de Adobe Flash y el auge de otras tecnologías, como HTML5 y WebM, la popularidad del formato FLV ha disminuido significativamente. A pesar de esto, todavía se encuentra en algunos sitios web y reproductores multimedia.
- MOV: es un formato de archivo multimedia desarrollado por Apple, utilizado principalmente en su reproductor QuickTime. Es un formato contenedor que puede incluir múltiples pistas de vídeo, audio, texto y metadatos en un solo archivo. Aunque está estrechamente asociado con productos y aplicaciones de Apple, como iPhone, iPad y iTunes, también se puede reproducir en otros dispositivos y sistemas

operativos mediante reproductores multimedia compatibles, como VLC Media Player y Windows Media Player (con el códec adecuado).

En este proyecto, se ha optado por utilizar el formato MP4 debido a su popularidad y amplia compatibilidad. Sin embargo, es importante tener en cuenta que si la grabación se interrumpe inesperadamente, el archivo MP4 podría dañarse y volverse irreparable.

En cuanto a la opción “Pistas de audio”, el programa permite seleccionar hasta 6 pistas distintas, lo que resulta útil si se dispone de diversas fuentes de audio y se necesita aplicar efectos o editar cada una de ellas de forma separada. En nuestro caso, se ha seleccionado una sola pista y se ha configurado el audio en modo mono.

En los menús “Codificador de vídeo” y “Codificador de audio”, se debe elegir el algoritmo que se utilizará para comprimir el vídeo y el audio antes de guardarlo o transmitirlo. La elección del codificador dependerá de factores como la calidad de vídeo deseada y el rendimiento del sistema. Cabe mencionar que los codificadores disponibles en OBS pueden variar según el *hardware* y el sistema operativo utilizado. A continuación, se mencionan algunos de los codificadores de vídeo más comunes que se pueden encontrar:

- x264: Es un codificador basado en *software* que utiliza la CPU de la computadora para comprimir el vídeo. A pesar de que es uno de los codificadores más populares y ampliamente utilizados debido a su alta calidad y compatibilidad, puede requerir una CPU potente para lograr un buen rendimiento, especialmente en resoluciones y velocidades de fotogramas más altas.
- NVENC: Es un codificador basado en *hardware* que utiliza las GPU NVIDIA para comprimir el vídeo. NVENC es más eficiente que x264 en términos de rendimiento, ya que libera recursos de la CPU y aprovecha la potencia de la GPU. Si se cuenta con una tarjeta gráfica NVIDIA compatible, este codificador puede ofrecer una calidad similar a x264 con un impacto menor en el rendimiento del sistema.
- AMD AMF: Es un codificador basado en *hardware* que utiliza las GPU AMD para comprimir el vídeo. Al igual que NVENC, AMD AMF libera recursos de la CPU y aprovecha la potencia de la GPU. Sin embargo, la calidad y el rendimiento pueden variar según la tarjeta gráfica AMD específica que se esté utilizando.
- Apple VT H.264: Este codificador está disponible para usuarios de macOS y utiliza la GPU de Apple para comprimir el vídeo. Al igual que NVENC y AMD AMF, este codificador basado en *hardware* libera recursos de la CPU y puede ofrecer una buena calidad y rendimiento en sistemas Apple compatibles.

Cuando se elige un codificador para la grabación en OBS, es importante considerar diversos factores, como la calidad de vídeo deseada, el rendimiento del sistema y la compatibilidad con los dispositivos de *hardware*. Los codificadores basados en *hardware*, como NVENC, AMD AMF y Apple VT H.264, suelen ser una buena opción si se busca liberar recursos de la CPU y aprovechar la potencia de la GPU para la codificación de vídeo. No obstante, si se prefiere la máxima calidad y compatibilidad, x264 es una opción sólida y ampliamente utilizada. En el caso específico de este trabajo, se utilizó un ordenador portátil con una tarjeta gráfica integrada en la CPU Intel Core i5-3360M. Aunque esta tarjeta gráfica no cuenta con una alta capacidad gráfica, la CPU es bastante potente. Por lo tanto, se optó por utilizar el codificador x264 para la grabación, ya que esta opción se ajusta mejor a las características de *hardware* disponibles y permite obtener una buena calidad de vídeo.

Una vez elegido el codificador adecuado para la grabación, el siguiente paso es ajustar la "Tasa de control" (*Bitrate Control*) para controlar la calidad y el tamaño del archivo de grabación. Es importante entender las diferencias entre los diferentes métodos de tasa de control para tomar una decisión informada. OBS ofrece diferentes métodos de tasa de control según el codificador utilizado. A continuación, se presentan las principales diferencias entre los cuatro métodos de tasa de control para el codificador x264, que son CBR, ABR, VBR y CRF:

- CBR (*Constant Bitrate*): Este método mantiene una tasa de bits constante durante todo el proceso de grabación, lo que genera archivos de tamaño predecible. Sin embargo, en escenas de alta complejidad, la calidad puede verse afectada, ya que el codificador no puede ajustar la tasa de bits para manejar la información adicional. Es recomendable para transmisiones en vivo, donde la estabilidad y compatibilidad con diferentes dispositivos son cruciales.

- ABR (*Average Bitrate*): ABR es similar a CBR, pero permite cierta variabilidad en la tasa de bits. Aunque la tasa de bits puede fluctuar, el promedio de bits a lo largo de la grabación se mantiene constante. Esto permite una mejor calidad de imagen en comparación con CBR, pero también puede generar archivos de tamaño menos predecible. ABR es una opción adecuada cuando se requiere un compromiso entre calidad y tamaño de archivo.

- VBR (*Variable Bitrate*): VBR ajusta la tasa de bits de acuerdo con la complejidad de las escenas. En momentos de alta complejidad, la tasa de bits aumenta, mientras que en momentos de baja complejidad disminuye. Esto permite una mayor calidad

de imagen y una mejor compresión en comparación con CBR y ABR, pero puede generar archivos de tamaño más variable y es menos adecuado para transmisiones en vivo.

- CRF (*Constant Rate Factor*): CRF es un método de tasa de control que se centra en la calidad de imagen en lugar de la tasa de bits. El usuario define un valor de CRF, y el codificador ajusta la tasa de bits automáticamente para mantener la calidad deseada. Valores más bajos de CRF generan mayor calidad y archivos más grandes, mientras que valores más altos reducen la calidad y el tamaño del archivo. CRF es ideal para grabaciones locales donde la calidad de la imagen es prioritaria y no se requiere un tamaño de archivo específico.

Al elegir un método de tasa de control para la grabación o transmisión en OBS, es importante considerar nuestras prioridades en términos de calidad de vídeo, predictibilidad del tamaño de archivo y requisitos de ancho de banda (en el caso de transmisiones en vivo). CBR y VBR son opciones más comunes para la mayoría de las situaciones, pero cada método tiene sus propias ventajas y desventajas.

En el caso de este trabajo, se ha utilizado el método de CBR (*Bitrate Constante*) para la tasa de control. Al seleccionar una tasa de bits adecuada para CBR, es importante tener en cuenta la resolución y la tasa de fotogramas del vídeo, así como el ancho de banda disponible (en el caso de transmisiones en vivo). En la Tabla 3.1 se presentan algunas recomendaciones de tasa de bits para diferentes resoluciones proporcionadas por Google [45] para su plataforma YouTube.

Es importante tener en cuenta que estas recomendaciones son solo una guía general y que la tasa de bits adecuada puede variar según la complejidad de la escena y el contenido del vídeo.

Para este trabajo, después de realizar pruebas con diferentes tasas de bits para comprobar el rendimiento del ordenador y teniendo en cuenta que se realizaran vídeos para ser compartidos en la plataforma YouTube [3] que tengan una calidad mínima de 1080p se ha decidido elegir una tasa de bits de 50000 Kbps.

A pesar de que ese valor pueda parecer exagerado, pues según la información publicada por Google [45], corresponde con una resolución 4K, se ha elegido teniendo en cuenta posibles pérdidas posteriores a la hora de la edición del vídeo, y sobre todo, porque no causaban ningún problema de rendimiento en el ordenador mientras se realizaban las grabaciones.

| Resolución | Bitrate |
|---|---|
| **2160p (4K Ultra HD) a 60 FPS** | 20.000-51.000 kbps |
| **2160p (4K Ultra HD) a 30 FPS** | 13.000-34.000 kbps |
| **1440p (Quad HD) a 60 FPS** | 9000-18.000 kbps |
| **1440p (Quad HD) a 30 FPS** | 6000-13.000 kbps |
| **1080p (Full HD) a 60 FPS** | 4500-9000 kbps |
| **1080p (Full HD) a 30 FPS** | 3000-6000 kbps |
| **720p (HD) a 60 FPS** | 2250-6000 kbps |
| **720p (HD) a 30 FPS** | 1500-4000 kbps |
| **480p (HQ)** | 500-2000 kbps |
| **360p (SD)** | 400-1000 kbps |
| **240p** | 300-700 kbps |

Tabla 3.1 : Tabla de *bitrate* según Google para la plataforma YouTube

En la sección de “Intervalo de fotogramas clave”, se configura la distancia entre fotogramas clave para la grabación de vídeo. Un valor de 2 es común, pero podría aumentarse a 4 o 6 si se experimentan problemas de rendimiento. En el caso de este trabajo, se ha elegido una configuración de 0, pues esta permite que el *software* de OBS decida el valor óptimo para la grabación.

En el caso de seleccionar el codificador x264, como en este trabajo, se puede ajustar el “Perfil de uso de la CPU” para equilibrar la calidad y la velocidad de codificación. Los ajustes preestablecidos más rápidos pueden reducir la calidad del vídeo, mientras que los ajustes preestablecidos más lentos pueden mejorar la calidad a expensas del rendimiento. En este trabajo se ha utilizado un perfil “Veryfast”.

Una vez finalizados los ajustes de grabación, es importante configurar los ajustes de vídeo para garantizar la calidad, resolución y velocidad de fotogramas adecuados en las grabaciones. Para acceder y modificar los ajustes de vídeo en OBS, es necesario ir a la pestaña “Vídeo” en el panel de la izquierda dentro de la ventana de “Ajustes”. La ventana que nos aparece al hacer clic en la pestaña “Vídeo” aparece en la Figura 3.18.

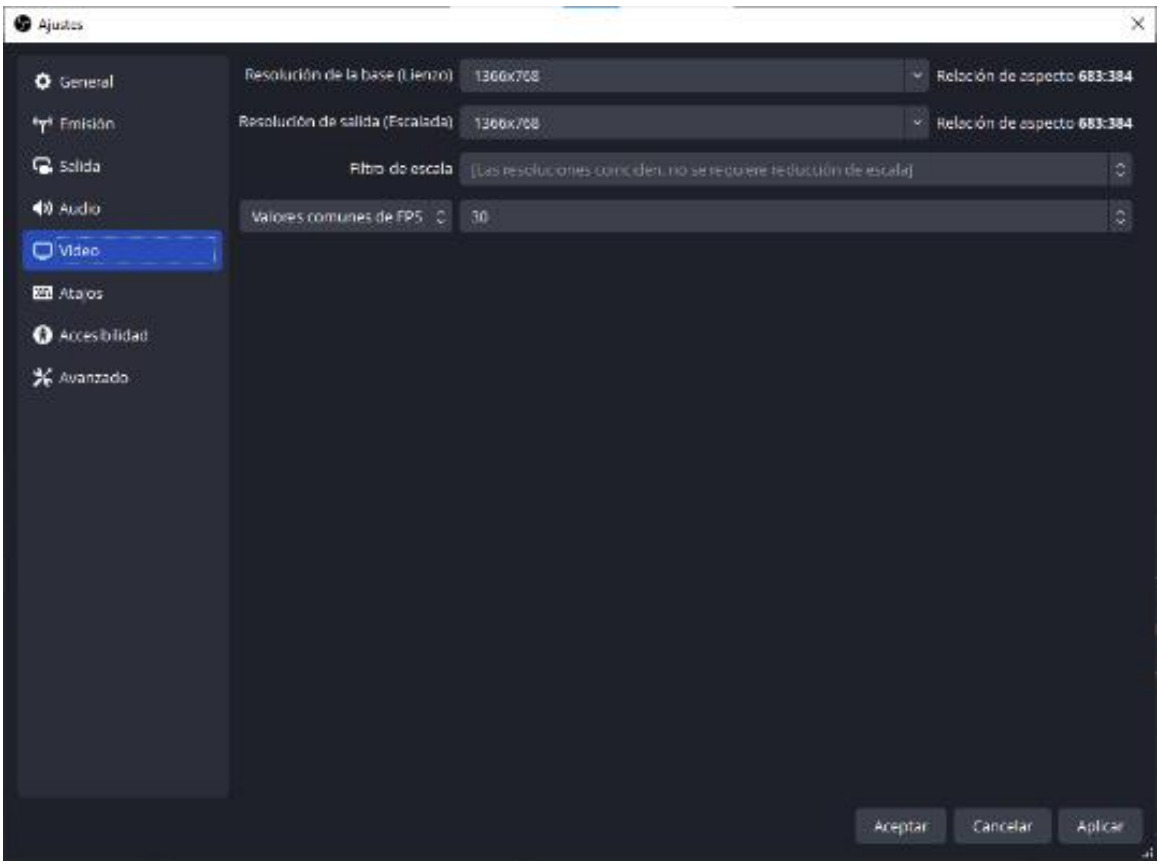


Figura 3.18 : Ventana de configuración de ajuste, con la pestaña “Vídeo”

A continuación, se presentan diversas alternativas relacionadas con la calidad del vídeo, la resolución y la velocidad de fotogramas, las cuales son:

- “Resolución de la base (Lienzo)”: Esta es la resolución del área de trabajo en la que organizamos y diseñamos nuestras escenas. Por lo general, debe coincidir con la resolución nativa de nuestro monitor, como 1920x1080 (1080p) o 1280x720 (720p). En el caso de este trabajo, el monitor utilizado tiene una resolución de 1366x768.
- “Resolución de salida (Escalada)”: Esta es la resolución final del vídeo que se grabará o transmitirá. Si deseamos ahorrar ancho de banda o mejorar el rendimiento, podemos seleccionar una resolución de salida menor que la resolución base. Por ejemplo, si la resolución base es 1920x1080, podemos seleccionar una resolución de salida de 1280x720 para obtener un vídeo de 720p. En el caso de este trabajo se ha elegido la misma resolución de salida que la de la base, es decir, 1366x768.
- “Filtro de escala”: Este filtro se utiliza cuando la resolución de salida es menor que la resolución base. Determina la calidad del escalado de la imagen. Entre los tipos de filtros encontramos:
    - Bicúbico: Basado en la interpolación bicúbica, este filtro considera los 16 píxeles más cercanos al punto de muestreo para calcular el color del píxel de

destino. Al tomar en cuenta más píxeles, logra mantener un buen nivel de detalle y nitidez en la imagen escalada. Es adecuado para la mayoría de los casos de uso, siendo una opción intermedia en términos de calidad y rendimiento.

- Lanczos: El filtro Lanczos utiliza una función sinc como función de interpolación. Este método toma en cuenta un mayor número de píxeles vecinos que el bicúbico, en concreto 32 píxeles, lo que resulta en una imagen escalada más nítida y con mayor fidelidad. Aunque consume más recursos de procesamiento, puede ser la mejor opción para usuarios que buscan la mayor calidad posible y cuentan con *hardware* más avanzado.
- Área: El filtro de área es especialmente útil cuando se reducen imágenes con detalles finos o altas frecuencias, ya que ayuda a evitar la aparición de artefactos y moiré (patrón de interferencia visual). Al promediar los píxeles en un área específica, el resultado es una imagen más suave y con menos ruido. Es especialmente útil para fuentes de vídeo que tengan detalles complejos y ruido inherente.
- Bilineal: La interpolación bilineal es un método de escalado que utiliza los 4 píxeles más cercanos al punto de muestreo para calcular el color del píxel de destino. Es un método menos preciso que el bicúbico o el Lanczos, pero consume menos recursos de procesamiento. Puede ser una opción viable para aquellos que tienen *hardware* limitado y están dispuestos a sacrificar calidad de imagen por un mejor rendimiento.

- "FPS (*Frames Per Second*)": Podemos elegir entre "Valores comunes de FPS", "Valor entero de FPS" y "Valor fraccional de FPS". Los valores comunes de FPS son opciones predefinidas, mientras que el valor entero de FPS permite ingresar manualmente un número de FPS, y el valor fraccional nos permite ingresar valores enteros del numerador y denominador de la fracción de FPS. Las opciones comunes incluyen 24, 30 y 60 FPS. Una velocidad de fotogramas más alta proporcionará una reproducción más fluida, pero consumirá más recursos y ancho de banda. En el caso de este trabajo se han utilizado 30 FPS.

Tras haber configurado los ajustes de grabación y de vídeo correspondientes, es necesario proceder a hacer clic en el botón "Aplicar", situado en la parte inferior de la

ventana de Ajustes, para luego seleccionar “Aceptar” a fin de guardar los cambios y cerrar la ventana.

Una vez llevada a cabo la configuración de las fuentes y ajustes requeridos, se procede a iniciar la grabación de los vídeos haciendo clic en el botón ubicado en la esquina inferior derecha de la ventana de OBS, el cual lleva por nombre “Iniciar grabación”. Para detener la grabación, basta con hacer clic en el mismo botón, el cual cambia de nombre y se convierte en “Detener grabación” mientras el programa se encuentra grabando.

Es pertinente resaltar que existe la posibilidad de grabar diversas secuencias de vídeo con la finalidad de editarlas posteriormente mediante algún programa de edición de vídeos. En la siguiente sección, se procederá a describir con detalle el *software* de edición de vídeo Shotcut, utilizado en el presente trabajo.

## 3.4. Shotcut

Shotcut [43] es un *software* de edición de vídeo de código abierto y multiplataforma, concebido inicialmente por Dan Dennedy en 2011 bajo el patrocinio de Meltytech, LLC. Con el transcurso del tiempo, su comunidad de desarrolladores y usuarios se ha expandido y fortalecido, contribuyendo a la mejora continua y el enriquecimiento del programa. Este esfuerzo conjunto ha permitido a Shotcut integrar funciones y herramientas avanzadas, tales como soporte para múltiples formatos, efectos visuales, transiciones, edición no lineal, entre otros. Transformándose así en una herramienta versátil y fácil de utilizar, que ofrece soluciones de edición de vídeo tanto para usuarios principiantes como profesionales.

El programa es compatible con sistemas operativos como Windows, macOS y Linux, lo que posibilita su utilización en diversos entornos y dispositivos. Además, su carácter de código abierto fomenta la colaboración y el intercambio de ideas entre usuarios alrededor del mundo, lo que ha generado una innovación y optimización constantes en el desarrollo de Shotcut.

En la actualidad, es considerado uno de los mejores editores de vídeo de código abierto disponibles en el mercado, gracias a su conjunto de características, la intuitiva interfaz de usuario y la evolución incesante impulsada por la comunidad.

En este proyecto, se empleó este *software* de edición de vídeo para llevar a cabo la segmentación y fusión de diversas grabaciones, lo cual se logró con precisión y eficiencia. Además, se utilizó el programa para separar las pistas de audio de las pistas de vídeo. Esta acción, en combinación con la capacidad del *software* Audacity, previamente mencionado,

de extraer las pistas de audio de los vídeos, permitió realizar una postproducción del audio, para posteriormente, reagruparlo con su vídeo original mediante Shotcut.

Este enfoque posibilitó el tratamiento y procesamiento de las pistas de vídeo y audio de forma individual, optimizando así las capacidades de cada programa y logrando un buen acabado en la composición. Cabe destacar que, aunque Shotcut permite la edición de audio, sus resultados en este ámbito pueden ser cuestionables, lo que hizo necesario la utilización de Audacity para garantizar un resultado final de alta calidad.

### 3.4.1. Cómo usar Shotcut

En esta sección, se enseñará cómo utilizar el sofisticado *software* de edición de vídeo Shotcut para llevar a cabo la segmentación y fusión de diversas grabaciones. Además, también se aprenderá a separar las pistas de audio de las pistas de vídeo utilizando el mismo programa.

Como puede verse en la Figura 3.19, en la esquina superior izquierda de la pantalla, se encontrará la barra de menú, que permitirá acceder a la totalidad de opciones y herramientas disponibles.

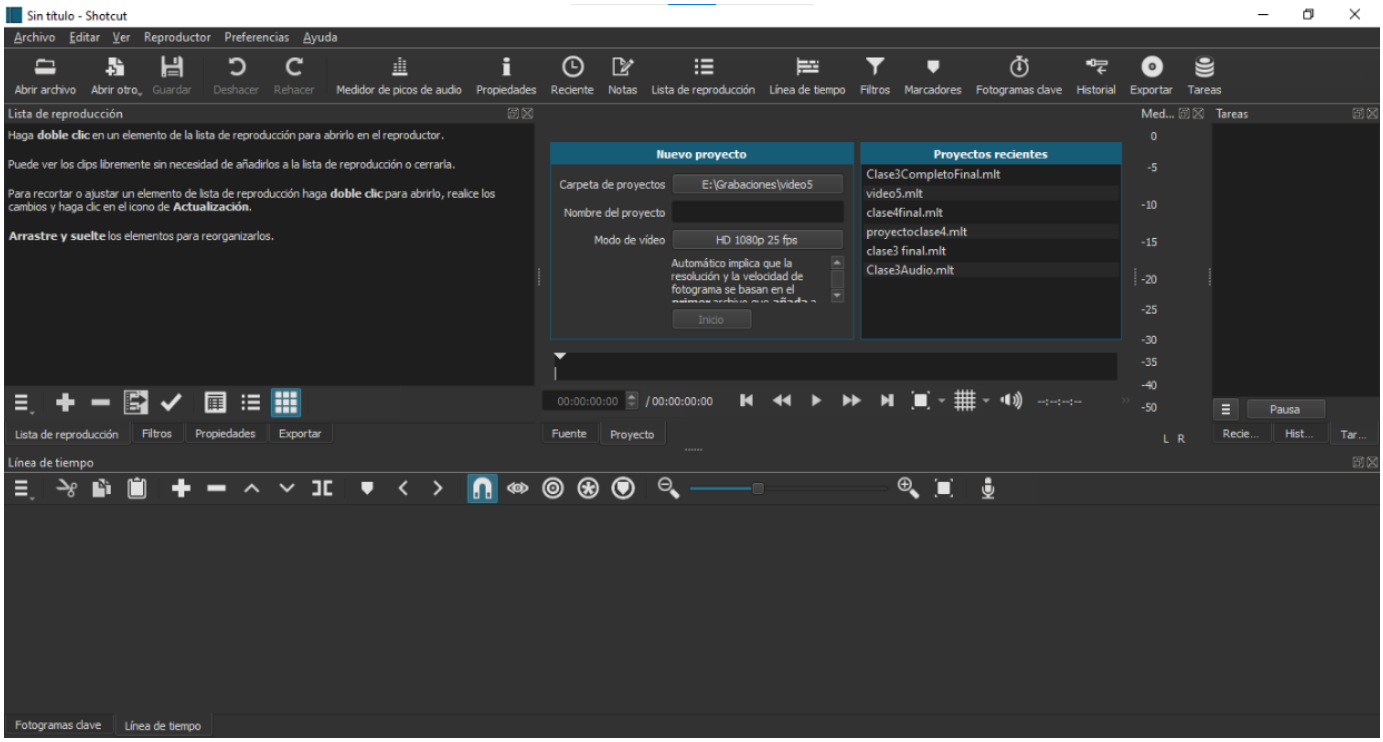


Figura 3.19 : Ventana principal de Shotcut

Cuando el programa se haya ejecutado, lo primero que se deberá hacer es importar los archivos de vídeo y audio que se necesiten para el montaje. Para ello, se deberá dirigir a la barra de menú ubicada en la parte superior de la interfaz y hacer clic en “Archivo” >

“Abrir archivo” para acceder al explorador de archivos del ordenador y poder seleccionar el o los archivos que se deseen importar.

Una vez importados, los archivos se mostrarán en la sección “Lista de reproducción”, localizada en la esquina superior izquierda de la ventana de Shotcut. Esta sección cumple la función de un panel de almacenamiento temporal y de organización de los archivos importados. En este espacio, se podrá observar una vista previa en miniatura y el nombre correspondiente a cada archivo. Adicionalmente, se tendrá la posibilidad de reorganizar el orden de los archivos simplemente arrastrándolos y soltándolos en la posición que se considere adecuada.

Para segmentar los archivos de vídeo que se deseen, se deberá arrastrar los archivos a la línea de tiempo en la parte inferior de la ventana. Luego, se utilizará la herramienta de selección para marcar los puntos en los que se desee hacer la segmentación y, a continuación, se hará clic derecho sobre la línea de tiempo y se elegirá la opción “Editar” > “Dividir en el cabezal de reproducción”. En caso de tener varias pistas, por ejemplo, una pista de vídeo y otra de audio, antes de hacer clic derecho sobre la línea de tiempo, se deberá seleccionar primero la pista que se desee dividir, para ello simplemente se hace clic izquierdo sobre la pista en cuestión.

Para fusionar los archivos de vídeo, se deberá arrastrar los segmentos de vídeo que se deseen unir en la línea de tiempo. Es necesario asegurarse de que los segmentos de vídeo estén uno al lado del otro en el orden en que se desee que se reproduzcan. Shotcut automáticamente fusionará los segmentos, creando un flujo continuo entre ellos cuando se exporte el proyecto.

Para separar las pistas de audio de las pistas de vídeo, se deberá hacer clic derecho sobre la pista y seleccionar la opción “Separar audio”. Esto creará una nueva pista de audio en la línea de tiempo, debajo de la pista de vídeo. Las pistas estarán ahora separadas y se podrán editar y manipular de manera independiente. Por ejemplo, se puede cortar segmentos no deseados del audio, ajustar el volumen, agregar efectos de sonido o incluso reemplazar la pista de audio original con una nueva. También se pueden editar las pistas de vídeo aplicando efectos, ajustando la velocidad de reproducción, recortando, rotando y más. Cabe mencionar que en este trabajo no se han aplicado efectos de sonido, ya que el programa Shotcut no ha sido ideado para esta tarea y, por tanto, los resultados serían deficientes.

Para eliminar un segmento, se deberá hacer clic derecho sobre la pista en la que se encuentre el segmento y seleccionar la opción “Cortar”. En caso de que se desee que, al

cortar el segmento, el segmento siguiente se mueva hacia la izquierda y se coloque al lado del segmento anterior del que se ha cortado, después de hacer clic derecho se deberá elegir la opción “Propagar eliminación” en lugar de la de opción de “Cortar”.

Una vez que se hayan completado todas las ediciones y se esté satisfecho con el proyecto, se tendrá que exportar a un archivo de vídeo final. Para hacer esto, se deberá hacer clic en el botón “Exportar” situado en la mitad superior de la ventana. Se habrá de seleccionar el formato de archivo de vídeo y las opciones de codificación deseadas. Luego, se deberá hacer clic en “Exportar archivo” y elegir la ubicación en la que se desee guardar el vídeo finalizado.

En este trabajo se ha elegido el formato MP4 y el *codec* libx264. En cuanto a la tasa de control, se ha optado por el método VBR basado en la calidad con una calidad del 100%. En lo que respecta a la codificación de audio, se ha seleccionado 1 canal mono con tasa de muestreo de 48 kHz, la misma que fue elegida al grabar en Audacity. Asimismo, se ha elegido el codec aac y la tasa de control nuevamente es VBR basada en la calidad, con una calidad del 100%.

## 3.5. Adobe Enhance

Adobe Enhance [44], también conocido como Enhance Speech, es una herramienta de procesamiento de audio impulsada por inteligencia artificial desarrollada por Adobe. Originalmente formaba parte de un proyecto de investigación de IA llamado Project Shasta, que recientemente fue renombrado como Adobe Podcast.

La herramienta está diseñada para mejorar la calidad de las grabaciones de voz, particularmente aquellas de mala calidad, eliminando el ruido de fondo y haciendo que la voz suene más fuerte. Cuando la herramienta funciona como se espera, el audio resultante puede sonar como una grabación realizada en una cabina de sonido profesional con un micrófono de alta calidad.

El uso de Enhance Speech es gratuito, pero requiere la creación de una cuenta de Adobe y funciona mejor con un navegador web de escritorio. Los usuarios pueden subir un archivo MP3 o WAV de hasta una hora de duración o 1GB de tamaño. Después de varios minutos de procesamiento, los usuarios pueden escuchar el resultado en su navegador o descargar el audio limpio.

Adobe no proporciona los detalles exactos de cómo funciona Enhance Speech, pero se cree que la compañía entrenó un modelo de aprendizaje profundo con muchas horas de audio limpio y ruidoso. Este modelo podría entonces "aprender" a seleccionar las frecuencias de la voz humana y sintetizar un sonido que coincida con precisión con la fuente. Sin embargo, esto es especulación hasta que Adobe proporcione más detalles técnicos.

Para usarla hay que acceder a su web [44] y posteriormente crear una cuenta. Después se pueden subir los audios que queramos mejorar. En la Figura 3.20 se puede ver una captura de pantalla de la web de Adobe Enhance.

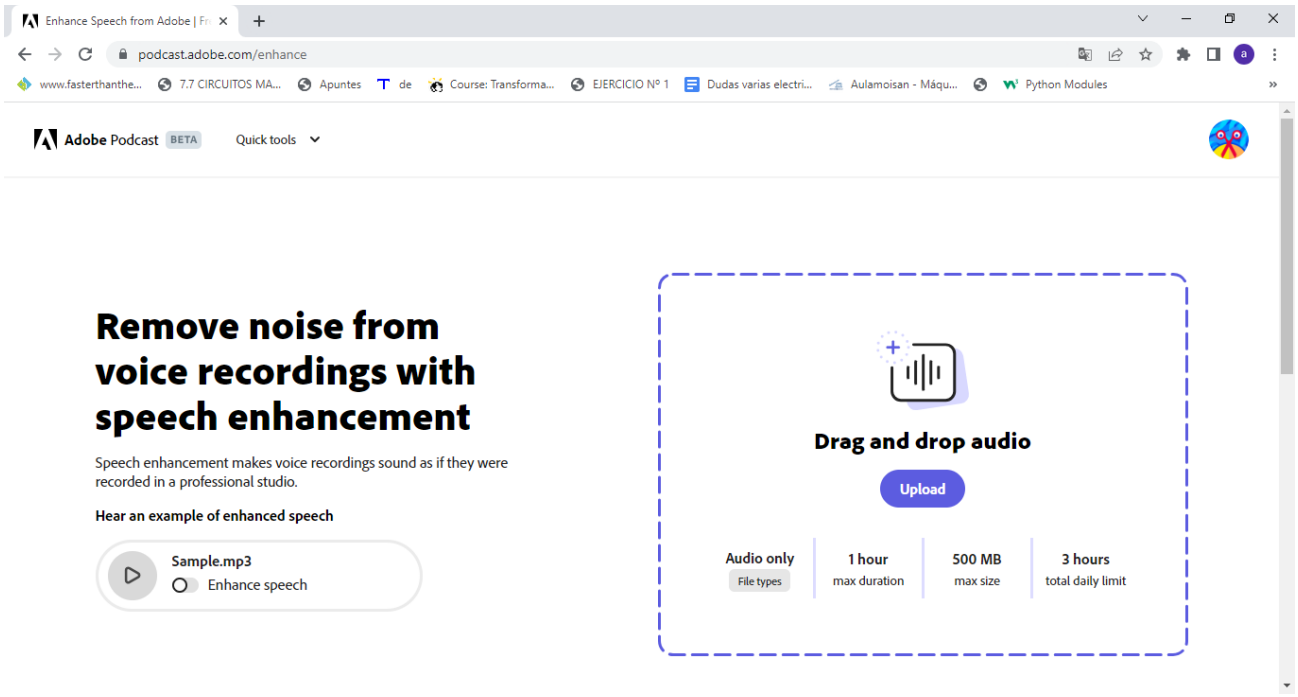


Figura 3.20 : Captura de pantalla de la web de Adobe Enhance

## 3.6. Herramientas *hardware* utilizadas

En las secciones precedentes se ha realizado una exhaustiva explicación sobre las herramientas de *software* utilizadas. En la sección actual, se procederá a la introducción de los componentes de *hardware*. Cabe destacar que, a diferencia del *software*, el *hardware* conlleva un coste económico asociado. El *software*, en su mayoría, está conformado por herramientas de código abierto, las cuales no implican un desembolso financiero, representando de esta manera una opción gratuita. Sin embargo, se debe mencionar la presencia de PowerPoint y Adobe Enhance, ambas constituyendo excepciones a la regla previamente mencionada, debido a que se tratan de *software* propietario. No obstante, estas últimas herramientas, aunque sean de propiedad, están disponibles de manera

gratuita. Adobe Enhance se encuentra a disposición del público en general, mientras que PowerPoint se encuentra al alcance sin coste para los miembros de la Universidad de Málaga.

| **Herramienta** | **Características** | **Precio** |
|---|---|---|
| **Ordenador** | Modelo: HP Elitebook 8570p<br>Procesador: Intel(R) Core(TM) i5-3360M CPU @ 2.80GHz (4CPUs)<br>Memoria RAM: 8182 MB<br>Sistema Operativo: Windows 10 Pro x64<br>Tarjeta Gráfica: Intel(R) HD Graphics 4000 | 300 € |
| **Auriculares** | Modelo: Medion (R) MD85111<br>Tarjeta de sonido: USB integrada<br>Transductores: neodimio 50nm<br>Banda de frecuencia: 20 Hz – 20 kHz<br>Impedancia: 32 ohmios | 20 € |
| **Micrófono** | Modelo: GAME MIC 320 USB<br>Filtro antiviento<br>Controles de eco, volumen y cancelación de micrófono | 30 € |
| **Total** | | 350 € |

Tabla 3.2 : Herramientas *hardware* utilizadas junto a sus características y a su precio

# Capítulo 4: Metodología

En este capítulo, se va a ofrecer una descripción exhaustiva de la metodología aplicada en el diseño, desarrollo y publicación de vídeos educativos destinados a enseñar el entorno y el lenguaje de programación de MATLAB en el canal “Aprende conmigo Informática” [4] de la plataforma YouTube. Se explorará de manera meticulosa cada una de las etapas involucradas en la creación de estos recursos audiovisuales. Este análisis incluirá la consideración de aspectos técnicos, pedagógicos y creativos, permitiendo una comprensión integral del proceso de creación de estos recursos.

El enfoque metodológico adoptado en este estudio pretende servir como una guía práctica y orientativa para aquellos interesados en desarrollar y compartir recursos educativos similares en el ámbito de la enseñanza y el aprendizaje de lenguajes de programación. Este enfoque permitirá, además, adaptar la metodología a diferentes contextos y temas, expandiendo su utilidad y aplicabilidad en la creación de contenidos educativos en línea. Es importante mencionar aquí que, para la elaboración de la metodología durante este proyecto, se ha seguido una metodología similar a la desarrollada por el estudiante Sergio Fernández Casasola en su proyecto “Vídeos docentes para el aprendizaje de programación C/C++ en Ingeniería Industrial” [47].

## 4.1. Etapas de la realización de un vídeo

Crear un vídeo educativo para enseñar MATLAB y publicarlo en YouTube [3] implica seguir una serie de etapas específicas que aseguran un resultado de calidad y coherente con los objetivos didácticos. A continuación, se detallan las principales etapas involucradas en la realización de un vídeo educativo de MATLAB para YouTube:

1. **Definición de objetivos pedagógicos y público objetivo**: Este punto ha sido desarrollado en el Capítulo 1 de este trabajo, concretamente en la Sección 1.3. Para resumir, podemos decir que el objetivo pedagógico sería crear vídeos educativos para un curso de introducción a MATLAB, para un público principiante, el cual no tiene ningún conocimiento sobre el tema o cuyos conocimientos sean básicos y, posiblemente, poco estructurados.

2. **Selección del tema en cuestión en el cual se basará cada vídeo**: Este punto también ha sido desarrollado en un capítulo anterior del presente trabajo, concretamente en el Capítulo 2. En dicho capítulo se ha desarrollado un temario basado en la asignatura Sistemas Informáticos impartida en varios grados de la Escuela de Ingenierías de la Universidad de Málaga. Por ende, cada vídeo intenta abordar algún apartado del temario. Esto es así para asegurarnos que cada vídeo tenga el suficiente contenido como para no crear una sensación de inutilidad en el estudiante, pero al mismo tiempo que los vídeos no sean demasiado largos como para generar aburrimiento, cansancio o frustración. También se han producido vídeos que podrían aparentar desviarse ligeramente del plan de estudios preestablecido. Sin embargo, su propósito es profundizar en algún apartado concreto del temario. Por ejemplo, el vídeo que explica los operadores que MATLAB ofrece para transponer vectores y matrices, tiene el objetivo de aclarar que existen dos tipos de estos operadores, a pesar de que el plan de estudios de la asignatura no hace alusión a este hecho. La intención es asegurar que el estudiante que visualice dicho vídeo no incurrirá en errores cuando se enfrente a un problema que requiera ser consciente de las diferencias entre ellos, a pesar de que en la asignatura de Sistemas Informáticos, al igual que en la mayoría de asignaturas de ingeniería, este tipo de problemas no suelen ser frecuentes.

3. **Investigación y selección de contenidos**: Tras haber tomado la decisión sobre el tema que se abordará en el vídeo, es fundamental llevar a cabo un proceso meticuloso de investigación y recopilación de información. Para lograr una enseñanza efectiva y una presentación coherente, es indispensable identificar y seleccionar las fuentes de información más relevantes y confiables. Para ello, es aconsejable consultar una variedad de fuentes, que pueden incluir apuntes académicos, libros especializados y documentación en línea, entre otros recursos. Dicho proceso de investigación permitirá profundizar en los conceptos esenciales y obtener una comprensión más sólida del tema en cuestión. En el caso específico de MATLAB, este lenguaje de programación cuenta con una plataforma en línea que

proporciona una amplia gama de información relacionada con su funcionamiento y aplicaciones [38]. Esta fuente resulta sumamente útil, ya que permite acceder a información actualizada, ejemplos prácticos y recursos didácticos adicionales que pueden enriquecer el contenido del vídeo y garantizar una experiencia de aprendizaje efectiva y significativa para el público objetivo. Por supuesto, también se ha hecho uso del material docente de la asignatura Sistemas Informáticos (diapositivas y ejercicios).

4. **Preparación del material de apoyo**: Esta fase es crucial para garantizar una transmisión efectiva y coherente de los conocimientos. La elaboración de material de apoyo de alta calidad permitirá una presentación más atractiva y comprensible para el público objetivo. Para lograrlo, se pueden emplear diversas herramientas y estrategias. Esta fase cuenta de dos subfases:

    - **Creación de un guion de contenidos**: Consiste en detallar los contenidos del vídeo con el mayor detalle posible, incluyendo los ejemplos que se usarán.
    - **Elección del tipo de estrategia (formato) del vídeo**: Hay dos opciones y ambas presentan ventajas y desventajas, por lo que es recomendable analizar las necesidades específicas del curso y del público objetivo antes de decidirse por una estrategia. En este trabajo se ha usado principalmente la primera estrategia, salvo en los vídeos de introducción al entorno y explicación de conceptos básicos, donde se ha decidido optar por la segunda estrategia, es decir, la captura de la pantalla, pues se ha considerado más efectiva.

        **A) Creación de una presentación en PowerPoint**: Esta herramienta permite diseñar una presentación visualmente atractiva y bien estructurada, con la inclusión de textos, imágenes y gráficos pertinentes. Es posible aplicar animaciones y efectos de transición que faciliten la comprensión de los conceptos y enriquezcan la experiencia educativa. Los contenidos de cada vídeo se pueden dividir en diapositivas (páginas o pizarras), de forma que el alumno puede parar el vídeo cuando la pizarra se llena de contenido y antes de borrarlo y pasar al siguiente punto. En la Sección 3.1 se describe con más detalle este *software*.

        **B) Utilización del programa OBS (Open Broadcaster Software)**: Esta herramienta permite capturar en tiempo real la pantalla del ordenador, el cursor y otros elementos multimedia que se deseen incorporar (como el

micro o la cámara). Gracias a sus funciones avanzadas, se puede mejorar la experiencia educativa al ofrecer una visión más dinámica y cercana al entorno real de trabajo de MATLAB. Al emplear esta estrategia, es fundamental planificar detalladamente las acciones y pasos a seguir durante la grabación, para asegurar una correcta transferencia de conocimientos y evitar confusiones. Si durante esta fase se ha optado por esta segunda estrategia, no se debe llevar a cabo ninguna acción, hasta que no se tenga escrito el guion, el cual se redacta en la etapa siguiente. En la Sección 3.3 se describe con más detalle este *software*.

5. **Elaboración de un guion**: Tras la preparación de los materiales de apoyo, es esencial proceder a la redacción de un guion detallado y estructurado. Este documento será la base de la narración que acompañará y explicará el contenido audiovisual desarrollado en la etapa anterior. Aunque no es necesario, se puede transcribir lo que irá diciendo la voz a lo largo de todo el vídeo (usualmente voz en *off*). Esto puede generar vídeos que pierdan la naturalidad y espontaneidad que hay al explicar algo sin un guion preciso, pero también permite corregir errores y que el guion sea revisado por terceros (particularmente por el tutor). El guion debe incluir los siguientes elementos clave:

    - Introducción: Un inicio atractivo, con un saludo de bienvenida, que sea conciso y que presente el tema, el propósito del vídeo y una breve descripción de lo que se abordará en el contenido.
    - Desarrollo: La exposición detallada de los conceptos, dividida en secciones o bloques temáticos. Cada sección debe estar organizada de manera lógica y progresiva, facilitando la comprensión del público. Es importante incluir ejemplos, analogías y explicaciones claras que permitan a los espectadores asimilar la información. Se debe tener en cuenta lo visto en vídeos anteriores, y también es posible anunciar lo que se verá en otros posteriores. El vídeo debe cumplir con las características que se han analizado en el Capítulo 1.
    - Transiciones: Establecer enlaces coherentes entre las secciones del guion, de manera que la narración fluya de forma natural y mantenga la atención del público.
    - Cierre: Una despedida que anime a la continuación con el curso.

El guion será la base para la narración del vídeo y permitirá al instructor mantener un enfoque claro y estructurado durante la grabación. Además, contribuirá a la cohesión y coherencia del material didáctico, asegurando una experiencia de aprendizaje efectiva y satisfactoria para el público objetivo.

6. **Grabación del contenido**: En esta etapa, la estrategia elegida para la creación del material audiovisual de apoyo determinará el enfoque a seguir durante la grabación. A continuación, se describen las acciones correspondientes a cada una de las dos estrategias mencionadas anteriormente:

    - **Enfoque basado en la creación de una presentación**: Si se ha optado por diseñar una presentación en PowerPoint, será necesario grabar las narraciones correspondientes a cada diapositiva siguiendo el guion previamente elaborado (ficheros de audio). Una vez realizadas las grabaciones, éstas deberán ser editadas, tal y como se explica en el Capítulo 3, y posteriormente se incorporarán como efectos sonoros, es decir narraciones en *off* a la presentación. De esta forma, la explicación se sincronizará con los elementos visuales para proporcionar una experiencia educativa cohesiva y efectiva. Después de hacer esto, simplemente se deberá generar el vídeo, tal y como ya se ha explicado. Para esta etapa se debe utilizar la herramienta Audacity para la grabación y edición del audio (*software* analizado en la Sección 3.2).

    - **Enfoque basado en la captura de pantalla**: Si se ha decidido utilizar la estrategia de grabar un vídeo mediante la captura de pantalla, se emplea el programa OBS para registrar las acciones en tiempo real. Durante la grabación, es fundamental seguir el guion de manera rigurosa y simultánea mientras se realizan las acciones planificadas en el programa MATLAB. Asimismo, se debe hacer uso del cursor del ratón para señalar y destacar aspectos relevantes, de modo que los alumnos puedan comprender y seguir las explicaciones de manera efectiva. Cuando se usa esta estrategia, se deben grabar varias secuencias, para posteriormente unirlas. Para ello se emplea el *software* Shotcut, el cual fue detallado en la Sección 3.4.

En ambos casos, es importante asegurarse de que la calidad de la grabación sea óptima, tanto en términos de audio como de vídeo. Además, la narración debe ser clara, fluida y pausada, evitando tecnicismos innecesarios y asegurándose de que los

conceptos sean accesibles para el público objetivo. De esta manera, se garantizará una experiencia de aprendizaje exitosa y satisfactoria para los estudiantes.

7. **Reedición de vídeos**: Un desafío frecuente en el ámbito de la producción audiovisual es la aparición de ciertos obstáculos técnicos tras la finalización de un vídeo. Estos inconvenientes pueden estar asociados tanto a elementos de la calidad auditiva como a aspectos de tiempo, incluyendo, por ejemplo, la presencia de pausas inesperadas o inoportunas. Ante tales circunstancias, se torna esencial emprender un proceso de revisión y reedición de las obras audiovisuales concernidas. En este contexto, se sugiere el uso de aplicaciones especializadas como Shotcut para la optimización de los vídeos. Este *software*, tal y como se describió en el Capítulo 3, proporciona un conjunto integral de herramientas diseñadas para facilitar ajustes precisos en el contenido audiovisual. De esta manera, Shotcut permite realizar recortes en el vídeo y volver a fusionar los fragmentos, solucionando algunos problemas de temporización. También nos permite separar el audio del vídeo, para editarlo (quitar ruido, por ejemplo) y posteriormente volver a fusionarlo con el vídeo. En el ámbito de la edición de audio, uno de los desafíos más comunes, en particular durante la producción de los primeros vídeos, es la falta de habilidades técnicas. En respuesta a este problema, se ha explorado la utilización de tecnologías emergentes, como la inteligencia artificial, para mejorar la calidad del audio. Adobe Enhance, es la herramienta que se ha utilizado en este proyecto para este fin (analizada en la Sección 3.5). Conforme a lo expuesto en la sección previa, Adobe Enhance ha evidenciado su competencia para generar resultados satisfactorios en la mejora de la calidad de audio. Sin embargo, es imperativo señalar que, debido a que este instrumento ha sido predominantemente entrenado con el idioma inglés, su implementación en grabaciones en español puede resultar en una alteración del acento original del locutor. Esta circunstancia podría afectar la estética integral del proyecto audiovisual. A pesar de ello, ponderando sus habilidades para realzar la calidad del audio frente a la menor repercusión estética, Adobe Enhance podría considerarse una alternativa factible en situaciones donde se disponga de habilidades limitadas en la edición de audio.

8. **Revisión y ajustes en el proceso de creación de vídeos**: Tras la producción de los vídeos, estos son remitidos al profesor encargado de tutorizar el trabajo con el objetivo de llevar a cabo una minuciosa revisión. Esta etapa es crucial para identificar posibles errores o aspectos susceptibles de mejora en el material audiovisual. En el caso de que se detecten mejoras, fallos o imperfecciones, es necesario retornar al

proceso previo de reedición, o incluso, en determinadas situaciones, a la fase de grabación o de elaboración de guion, este último caso se hace especialmente necesario en el caso de errores en el contenido pedagógico. Esta retroalimentación resulta esencial para garantizar un producto final de calidad y que cumpla con las expectativas académicas y pedagógicas establecidas.

9. **Publicación en la plataforma YouTube**: Tras la aprobación del tutor, los vídeos producidos deben ser compartidos con el público a través de una plataforma de distribución apropiada. YouTube [3], al ser una de las plataformas de vídeos más populares y accesibles, tal y como se comentó en el Capítulo 1, se presenta como el medio ideal para la divulgación de estos contenidos. En el contexto de este proyecto específico, los vídeos han sido publicados en el canal "Aprende Conmigo Informática" [4]. Este canal, propiedad del Dr. José Galindo Gómez, sirve como un espacio virtual donde se comparten contenidos educativos relacionados con el campo de la informática. Cabe destacar que este canal no solo actúa como un medio de difusión de los vídeos producidos, sino también como una plataforma de interacción con los espectadores, quienes pueden comentar, preguntar y discutir acerca del contenido presentado. Antes de la publicación, es crucial realizar una serie de pasos para garantizar una correcta distribución del contenido. Estos incluyen, pero no se limitan a los siguientes apartados:

    - Creación de un título breve y adecuado a todo el contenido.
    - Descripción de los vídeos para asegurar que sean claros y atractivos, así como que sirvan de un índice de los contenidos de cada vídeo, de forma que el estudiante puede saber con anterioridad todo lo que va a aprender en ese vídeo.
    - Selección de las etiquetas adecuadas para aumentar la visibilidad del vídeo en las búsquedas.
    - YouTube exige indicar si el vídeo ha sido creado, o no, para niños. En nuestro caso siempre hemos marcado negativamente esa opción.
    - Elección de miniaturas atractivas para los espectadores (la imagen que aparece de cada vídeo, antes de empezar a reproducirlo)

- Definición de tarjetas del vídeo. Se trata de referencias a otros vídeos que se hacen en momentos concretos de la reproducción, que ayude al estudiante a adquirir la información que necesita en cada momento.
- Vídeos finales recomendados. Al terminar un vídeo, YouTube permite recomendar vídeos concretos o listas de reproducción. Pueden ser arbitrarios (elegidos por YouTube), o bien, ser fijados de antemano.
- Configuración de las opciones de privacidad de acuerdo con las necesidades y objetivos del canal.

Al seguir estas etapas, se podrán crear vídeos educativos efectivos para enseñar MATLAB o cualquier otro lenguaje de programación y alcanzar un público amplio y diverso en la plataforma YouTube [3]. En la Figura 4.1, se resume en un diagrama el contenido explicado anteriormente.

## 4.2. Tiempo empleado

En este capítulo también es necesario comentar el tiempo que conlleva la realización de vídeos de este tipo. Este tiempo depende mucho de las habilidades que se tenga en el manejo de las herramientas comentadas durante el Capítulo 3, además de las capacidades del creador de no cometer errores para que las fases de revisión y reedición no consuman la mayor parte del tiempo empleado. Estas habilidades y capacidades van mejorando a medida que se vayan produciendo los vídeos. El tiempo empleado también dependerá del volumen del contenido creado. Este volumen se resume en la Sección 5.1 y 5.2 del presente proyecto.

Es posible estimar que los primeros 10 vídeos necesitaron entre 30 y 40 horas de trabajo, es decir 5 horas al día durante unos 6 o 8 días. Teniendo en cuenta que la duración total de estos vídeos es de 1 hora, 22 minutos y 12 segundos, es decir 4932 segundos, y suponiendo un tiempo promedio empleado de 35 horas por vídeo, es decir, 126000 segundos por vídeo, y como son 10 vídeos el tiempo estimado total es de 1260000 segundos, esto nos da un estimado de 255 (4 minutos y 15 segundos) segundos empleados para cada segundo de vídeo creado, en estos primeros desarrollos.

En cambio, los últimos 17 vídeos conllevaron entre 10 y 25 horas, es decir 5 horas al día durante unos 2 o 5 días. Es posible notar la gran diferencia de tiempo invertido, eso es debido, tal y como se dijo antes a la mejora de las habilidades en la creación del contenido audiovisual. Teniendo en cuenta que la duración total de estos vídeos es de 1 hora, 42

minutos y 29 segundos, es decir, 6149 segundos, y aproximando el tiempo invertido por vídeo a 17,5 horas, es posible afirmar que para estos vídeos, obtenemos unos datos promedio aproximados de 174 (2 minutos y 54 segundos) segundos para cada segundo de vídeo creado.

Haciendo la media de ambos datos, obtenemos unos datos globales estimados de 204 segundos invertidos por segundo de vídeo creado. Si ahora multiplicamos esa cantidad por la duración total del contenido audiovisual, que es de 3 horas 4 minutos y 41 segundos, es decir, 11081 segundos, es posible afirmar que para la elaboración de todo el contenido multimedia se ha empleado un tiempo aproximado de 627 horas.

Hay que tener en cuenta que parte de ese tiempo ha sido invertido por el autor en compañía del tutor del proyecto, lo cual se le ha de reconocer y agradecer.

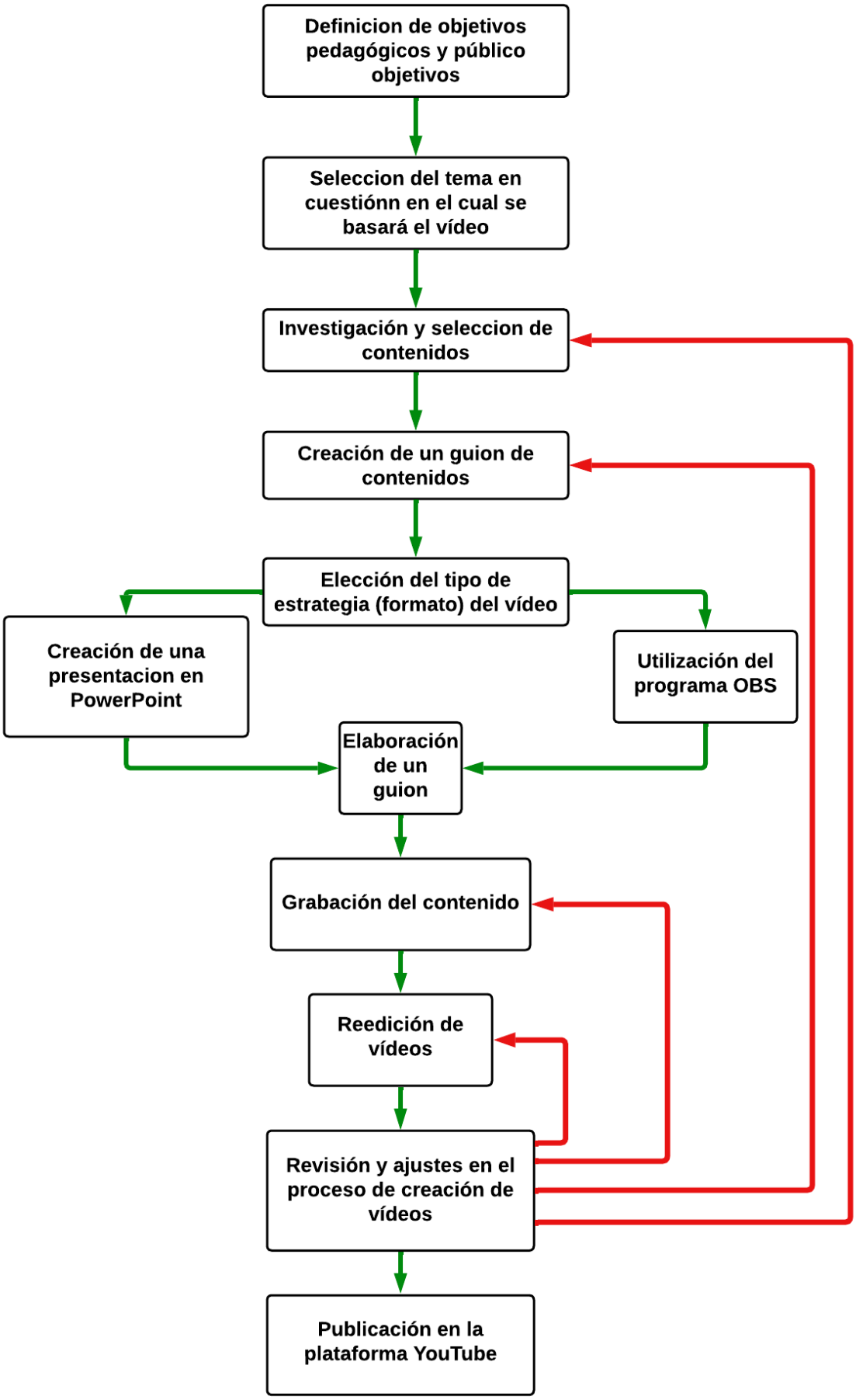


Figura 4.1 : Diagrama explicativo de las fases de creación de un vídeo

# Capítulo 5: Vídeos creados y publicados

En el presente capítulo, se realiza una presentación exhaustiva y meticulosa del contenido que comprende la serie de vídeos integrantes del curso introductorio a MATLAB, difundidos por medio del canal de YouTube "Aprende conmigo Informática" [4]. También se incluirá junto al resumen del contenido de cada vídeo la imagen usada para la miniatura que aparece en la plataforma YouTube. Además, también se incorpora en esta memoria, para cada vídeo la descripción tal y como se presenta en el mencionado canal que sirve a modo de índice del vídeo con aclaraciones particulares cuando se ha considerado interesante. Una cosa importante a resaltar aquí, es que el formato de las descripciones se ha mantenido fiel al formato que tienen en YouTube, por ejemplo, en el resto del trabajo cuando se han escrito comandos, estos se les ha dado un tipo de letra Courier, en cambio, debido a que las descripciones de YouTube no permiten usar ese tipo de letra, se ha decidió que cuando aparezcan en este capítulo, mantengan dicho formato.

Cada vídeo de este curso se enfoca en un aspecto particular de MATLAB, presentando la teoría de manera precisa y concisa, seguida de ejemplos aplicados que asisten en la consolidación del aprendizaje. El contenido sobre el cual se fundamentan estos vídeos fue presentado en el Capítulo 2 de este documento y, como se mencionó previamente, está basado en el programa de estudio de la materia Sistemas Informáticos de algunos grados de la Escuela de Ingenierías Industriales de la Universidad de Málaga, impartida por distintos profesores entre los que se encuentra el tutor de este trabajo.

Es importante destacar en este capítulo, que el orden de publicación del material audiovisual desarrollado es el mismo que el de la descripción de los vídeos. Dicho orden

debía ser acorde con el programa de la asignatura, lo cual no ha sido siempre factible debido a inconvenientes técnicos; sin embargo, la plataforma YouTube [3] propone una solución a esta problemática: la creación de listas de reproducción. Dichas listas comprenden la organización de los vídeos publicados en un canal, garantizando a los seguidores una secuencia temática que asegure una correcta asimilación de los conceptos. Por tanto, al culminar este capítulo, se presentarán todas las listas de reproducción creadas con este material, indicando el orden de visualización propuesto para los vídeos que las conforman. Asimismo, también se proporcionará la duración temporal de cada uno de los vídeos creados. Además de detallar la longitud individual de cada vídeo, se calculará y presentará la duración total del contenido generado, obtenida a través de la sumatoria de las duraciones de todos los vídeos creados.

# 5.1. Vídeos publicados

En esta sección se hará un resumen del contenido de cada uno de los vídeos junto con la descripción que incorpora cada uno de ellos en la plataforma YouTube [3], tal y como se mencionó anteriormente.

El orden de los siguientes vídeos sigue el orden natural en el que un alumno podría hacerlos para seguir el curso. Por supuesto, este orden no es estricto y puede alterarse en ciertos casos sin que suponga ningún conflicto.

## 5.1.1. Qué es MATLAB

**Descripción de YouTube**

Esta es la primera clase, y en ella vamos a ver una introducción a lo que es MATLAB, a lo que puede ofrecernos, sus características generales y algunas aplicaciones comunes. Con un poco más de detalle, veremos:

- Introducción.
- Historia.
- Características de MATLAB.
- Aplicaciones de MATLAB.
- Lenguaje de programación de MATLAB.
- Un ejemplo sencillo de uso: resolver un sistema de dos ecuaciones con dos incógnitas.

**Resumen**

Este es un vídeo introductorio al *software* de programación numérica y gráficos MATLAB. El vídeo inicia describiendo a MATLAB como una herramienta poderosa, versátil y fácil de usar utilizada para el análisis de datos y la resolución de problemas matemáticos, popular en la industria, docencia e investigación.


Figura 5.1 : Imagen de la miniatura usada en YouTube del vídeo: Qué es MATLAB

Se proporciona una breve historia de MATLAB, explicando que fue desarrollado en la década de 1970 por Cleve Moler en la Universidad de Nuevo México para resolver problemas de álgebra lineal y operaciones con matrices. Desde entonces, se ha expandido a otras áreas de las matemáticas y la ingeniería. En 1984, Moler fundó Mathworks, una compañía de *software* especializada en herramientas de computación técnica.

El vídeo destaca las características principales de MATLAB. Estas incluyen la capacidad de resolver problemas matemáticos complejos como sistemas de ecuaciones lineales, optimización, y análisis numéricos y estadísticos. Además, cuenta con una interfaz de usuario amigable con un editor de código, navegador de archivos y un depurador entre otras herramientas. También se menciona su extensa biblioteca de funciones predefinidas para cálculos numéricos y análisis de datos. MATLAB permite la creación de visualizaciones gráficas de alta calidad. Finalmente, MATLAB posee su propio lenguaje de programación,

el lenguaje M, pero también ofrece la posibilidad de integrarse con otros lenguajes como C++ y Python.

Luego, se describen algunas aplicaciones de MATLAB en campos como la ingeniería, la física, la biología, la informática, la medicina, la economía y la estadística, incluyendo el análisis y procesamiento de señales, el modelado y simulación de sistemas dinámicos, análisis estadístico y de datos, diseño y análisis de algoritmos, y optimización y control.

El vídeo concluye con un ejemplo de código. En el ejemplo, se resuelve un sistema de dos ecuaciones y se muestra la solución en la ventana de comandos de MATLAB.

### 5.1.2. Entorno de MATLAB y conceptos básicos

**Descripción de YouTube**

En este vídeo se muestran los conceptos más elementales sobre MATLAB:

- Recursos disponibles en la barra de herramientas.
- Conceptos básicos sobre variables y algunos comandos de uso frecuente.
- Ventanas del entorno: Command Window (Ventana de comandos), Workspace (Espacio de trabajo) y Current Folder (Directorio actual).
- El prompt de MATLAB.
- Asignacion básica de valores a variables.
- Los escalares se representan como matrices 1x1.
- Modo “no echo” (para no mostrar las salidas de las operaciones).
- Algunas operaciones básicas con escalares.
- Consultar valores de variables.
- Variable ans (answer). - Command History (Historial de comandos).
- Set Path (dónde MATLAB busca las funciones).
- Preferences (para modificar propiedades del entorno de MATLAB).
- Modificar colores y tipos de letra de MATLAB.

**Resumen**

Este vídeo es una introducción al entorno de trabajo de MATLAB, durante la cual se presentan varias características y herramientas que MATLAB ofrece.

Primero, se habla sobre la versión del programa que se está utilizando, indicando que la mayoría de las funciones son compatibles con las versiones más antiguas y nuevas. Se hace hincapié en la importancia de la barra de herramientas situada en la parte superior, que incluye tres pestañas principales: “HOME”, “PLOTS” y “APPS”.

En la pestaña “HOME”, se muestra cómo se pueden crear nuevos “scripts” y “live scripts”, además de presentar opciones para importar datos en diferentes formatos y abrir “Simulink”, una herramienta de programación gráfica. Se explican otras características, como la organización de los directorios en el “Set Path”, donde MATLAB buscará las funciones.

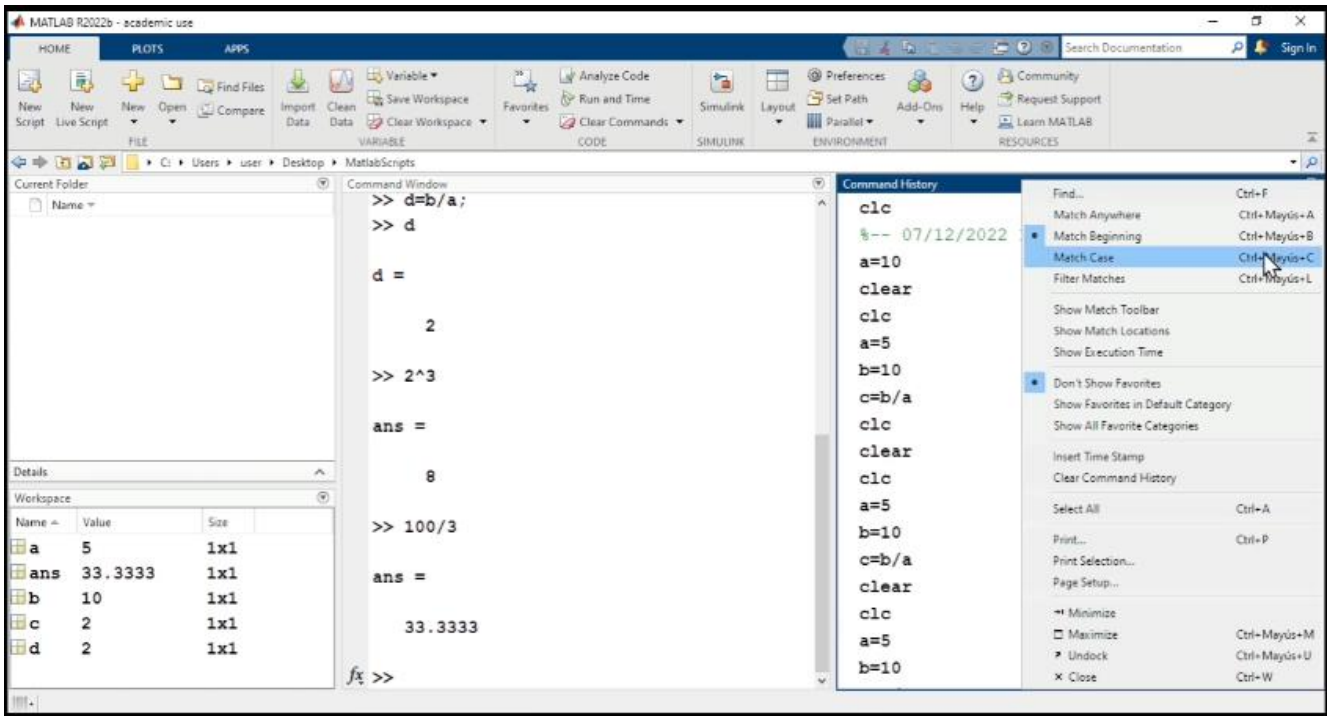


Figura 5.2 : Imagen de la miniatura usada en YouTube del vídeo: Entorno de MATLAB y conceptos básicos

En la pestaña “PLOTS”, se ven las opciones de gráficas que ofrece MATLAB, mientras que en la pestaña “APPS” se muestran las aplicaciones predefinidas, las cuales consisten en librerías de funciones y entornos gráficos listos para ser usados.

Se presenta también la personalización de la disposición de ventanas a través del menú “Layout”, y se mencionan las tres ventanas principales que aparecen en el modo default: “Current Folder” (directorio actual), “Command Window” (ventana de comandos) y “Workspace” (espacio de trabajo).

En la ventana de comandos, se realizan una serie de ejercicios básicos para ilustrar cómo MATLAB interpreta y ejecuta las instrucciones. Se hace hincapié en el uso del punto y coma para evitar que se muestre el resultado de una operación.

Se introduce el concepto de variables en MATLAB, explicando que no necesitan ser declaradas con anterioridad y que MATLAB también puede usarse como una calculadora. Se menciona la variable especial **`ans`**, que guarda el resultado de la última operación ejecutada que no se haya asignado a una variable.

Posteriormente, se exploran las ventanas de "Workspace" y "Current Folder". En la primera, se pueden ver las variables guardadas y algunos detalles sobre ellas. En la segunda, se guardan todos los archivos y funciones creadas.

Para terminar, se muestran algunas opciones de personalización de la interfaz gráfica de MATLAB a través de la ventana de "Preferences", incluyendo la modificación de los colores y tipos de letra utilizados.

### 5.1.3. MATLAB: Conceptos básicos sobre variables y ficheros .mat

**Descripción de YouTube**

En este vídeo se muestran los siguientes conceptos básicos sobre MATLAB:

- Ver y modificar variables en MATLAB.
- Editor de variables (comando openvar).
- Cambiar los formatos de visualización de números del editor de variables.
- Comando clc, para borrar el Command Window.
- El prompt del Command Window.
- Comando clear, para borrar variables y liberar su memoria.
- Ficheros .mat para guardar los datos de las variables.
- Comandos save y load para ficheros .mat.
- Opciones del Current Folder.

**Resumen**

Este vídeo es una introducción al uso de variables y archivos .mat en MATLAB.

Se explican las distintas formas de visualizar el contenido de las variables, por ejemplo, escribiendo el nombre de las variables en la ventana de comandos, haciendo doble clic sobre el nombre de estas en el espacio de trabajo o bien usando el comando **`openvar`**.

Se muestra cómo usar el editor de variables para ver y editar el contenido de las variables, cambiando los valores o transformando un escalar en una matriz, completando las casillas vacías con ceros para mantener la coherencia.

Luego se explican los comandos **`clc`** y **`clear`** para limpiar la ventana de comandos y borrar todas las variables del área de trabajo, respectivamente.

La siguiente sección se centra en la importancia de guardar las variables en archivos para su uso en otras sesiones. Para guardar todas las variables en un archivo .mat, se hace clic en “Save Workspace” situado en la sección variables de la pestaña HOME. Para guardar un subconjunto de variables, se pueden seleccionar en el área de trabajo y arrastrar directamente al directorio actual. También se puede usar el comando **`save`** para guardar todas las variables o un subconjunto específico de ellas.

Finalmente, se muestran las formas de cargar las variables guardadas desde un archivo .mat al área de trabajo, ya sea haciendo doble clic en el archivo en el explorador del directorio actual o usando el comando **`load`**. Además, se menciona cómo seleccionar y cargar solo ciertas variables de un archivo **.mat** y cómo visualizar las variables almacenadas en él antes de cargarlas.


Figura 5.3 : Imagen de la miniatura usada en YouTube del vídeo: Conceptos básicos sobre variables en MATLAB y archivos .mat

## 5.1.4. Scripts o ficheros .m de MATLAB: creación y ejecución

### Descripción de YouTube

En este vídeo se explicarán una serie de conceptos sobre los scripts de MATLAB:

- ¿Que es un script?
- Crear un script, tanto desde el entorno de MATLAB, como desde un editor de textos.
- Editar un script y poner comentarios.
- Guardar script, desde el entorno de desarrollo de MATLABy desde un editor de textos.
- Ejecutar script desde el Command Window y desde el Editor de MATLAB.
- Abrir script desde el Command Window y desde el Current Folder.
- Visualización de la salida de los comandos en el Command Window.

### Resumen

El vídeo es un tutorial de MATLAB centrado en el uso de scripts. Los scripts son archivos de texto que contienen una secuencia de comandos de que se pueden utilizar para automatizar tareas y realizar un conjunto de operaciones secuencialmente.

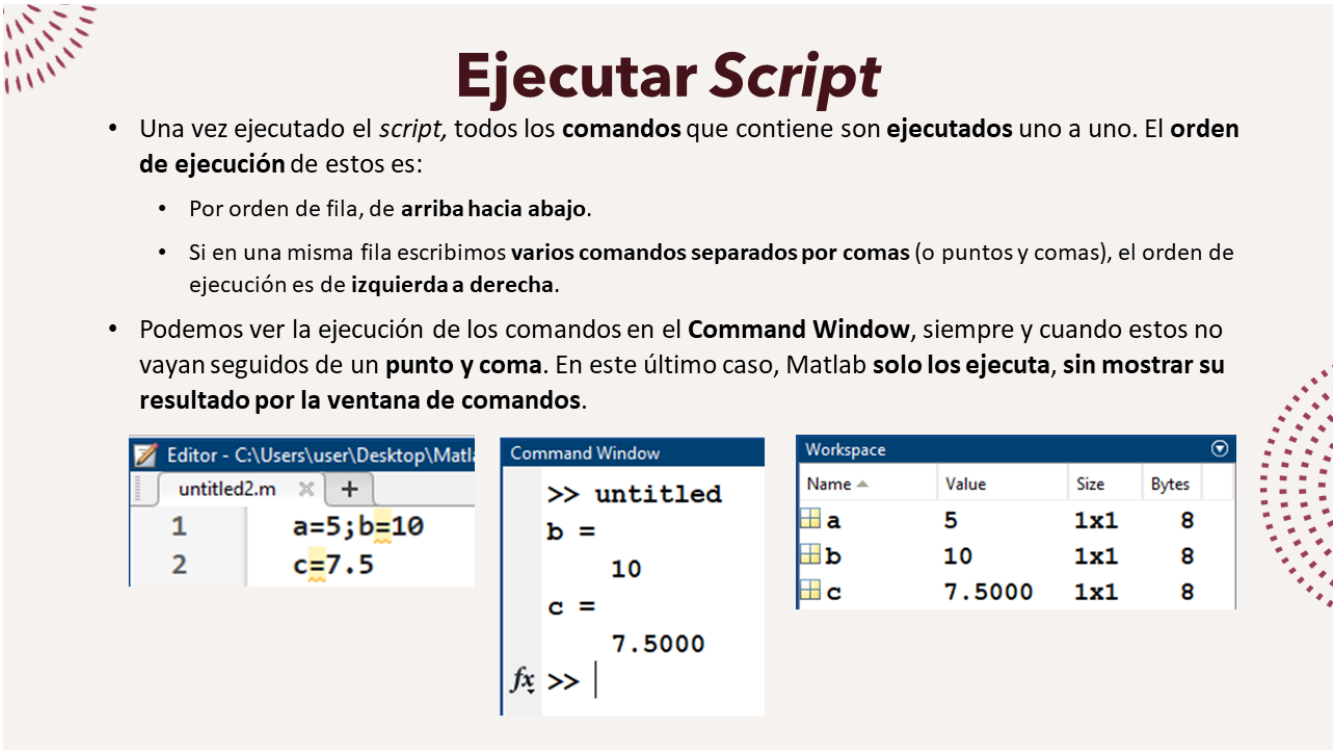


Figura 5.4 : Imagen de la miniatura usada en YouTube del vídeo: Scripts o ficheros .m de MATLAB: Creación y ejecución

Existen dos formas de crear un script: dentro del entorno de MATLAB, a través de la pestaña “HOME” haciendo clic en “New Script”, o usando un editor de texto externo, en cuyo caso el archivo debe guardarse con la extensión **.m**. En los scripts, los comandos se introducen en líneas distintas o en la misma línea separados por comas o puntos y comas.

Los scripts también pueden contener comentarios, que son precedidos por el símbolo del porcentaje `%` y son ignorados por MATLAB durante la ejecución. Los comentarios son útiles para explicar el propósito de las líneas de código y también pueden ser utilizados para desactivar algunas de estas líneas.

El tutorial también proporciona un ejemplo de script, explicando cada línea y cómo se interpreta y se ejecuta en MATLAB. Luego se explica cómo guardar y ejecutar los scripts, tanto dentro como fuera del entorno de MATLAB. Para ejecutar un script, se debe utilizar el comando **`run`** seguido del nombre del script, o simplemente el nombre del script.

Finalmente, se destaca la importancia de observar la ejecución de los comandos en la ventana de comandos, lo que permite comprender el orden en que se ejecutan las instrucciones.

### 5.1.5. MATLAB aplicado a la astrofísica: ¿En cuánto tiempo se consumirá toda la masa del Sol?

**Descripción de YouTube**

En este vídeo se resolverá un problema básico de física:

a. Sabiendo la cantidad de energía que el Sol irradia cada segundo, y usando la ecuación de Einstein E=m*c^2, debemos calcular cuánta masa se convierte en energía irradiada por el Sol cada día.

b. Sabiendo la masa aproximada del Sol y teniendo en cuenta el resultado del apartado anterior, debemos calcular el tiempo que tardará el Sol en perder toda su masa.

Los conceptos sobre MATLAB que se verán en este vídeo son:

- Creación y uso de variables (véase otro vídeo específico sobre esto).
- Operadores matemáticos básicos.
- Edición y ejecución de scripts (que son programas en MATLAB guardados en ficheros .m y que se explican detalladamente en otro vídeo).
- Comentarios en el código de un script (en MATLAB empiezan por %).

- Depuración de errores.

**Resumen**

El vídeo es un tutorial de MATLAB para resolver un problema específico que pregunta cuánto tiempo sería necesario para que el sol consuma toda su masa.

El enunciado del problema propone que el Sol irradia 3.65 x $10^{24}$ Julios por segundo debido a reacciones nucleares que convierten la materia en energía. Se requiere determinar cuánta materia solar se convierte en energía cada día utilizando la ecuación de Einstein ($E=mc^2$) y luego calcular cuánto tiempo se consumirá toda la masa del Sol, considerando que su masa es de 2 x $10^{30}$ kg.


Figura 5.5 : Imagen de la miniatura usada en YouTube del vídeo: MATLAB aplicado a la astrofísica: ¿En cuánto tiempo se consumirá toda la masa del Sol?

El proceso de resolución se divide en dos partes:

1. Primero, se introduce el dato de la potencia solar radiada y lo asigna a una variable `P`. Se convierte la potencia de Julios por segundo a Julios por día multiplicando por el número de segundos en un día. Luego, se introduce el valor de la velocidad de la luz c = 300x$10^6$ metros por segundo, y se utiliza la ecuación de Einstein para calcular la cantidad de masa convertida en energía por día. Aquí hubo un error inicial debido a que se usó una variable `E` no definida, que luego se corrigió por `P`.

2. Segundo, se calcula cuánto tiempo tomaría consumir toda la masa del Sol. Para esto, se divide la masa total del Sol por la masa consumida por día para obtener el tiempo en días, y luego se convierte este valor en años.

Finalmente, el programa se ejecuta correctamente y da los resultados de la masa consumida por día que es de 3.696 x $10^{14}$ kg, y el tiempo total que se necesitaría para consumir toda la masa del Sol que es de 1.4825 x $10^{13}$ años.

### 5.1.6. Formatos de salida de MATLAB y notación científica

**Descripción de YouTube**

En este vídeo veremos cómo modificar los formatos de visualización de números en el Command Window. Después se explicará qué es la notación científica y cómo usarla en MATLAB. Con un poco más de detalle, este vídeo explica lo siguiente:

- Formatos numéricos de salida: comando format de MATLAB.
- Formato interno de MATLAB.
- Formatos del espaciado de líneas en el Command Window.
- Notación científica en MATLAB: El carácter "e" en Matlab representa las potencias de 10 (también se puede usar "E" en mayúsculas).

**Resumen**

Este vídeo se enfoca en cómo modificar los formatos de visualización de la salida en la ventana de comandos y en explicar la notación científica.

Primero se introduce el concepto de formatos numéricos de salida, que se refieren a cómo MATLAB presenta los números al usuario en la ventana de comandos. A pesar de que se puede cambiar el formato de salida, MATLAB siempre realiza cálculos en punto flotante de doble precisión, lo que significa que los cambios en el formato de salida no afectan la precisión de los cálculos ni la forma en que se almacenan los números.

Luego se describen varios estilos de formato de salida, incluyendo **`short`**, **`long`**, **`shortE`**, **`longE`**, **`bank`**, **`hex`**, **`rat`**, y **`default`**. Cada uno de estos formatos tiene características únicas en cuanto a la cantidad de dígitos después del separador decimal, la representación en notación científica, y otros detalles. También se explican los formatos que afectan el espaciado en blanco en las salidas de MATLAB, los cuales son el formato **`compact`** y el estilo **`loose`**. Todos estos estilos anteriores se deben introducir junto al

comando **`format`**. Es decir, primero se escribe **`format`** y a continuación, separados por un espacio en blanco, cualquiera de los estilos anteriores.

Posteriormente, se introduce la notación científica, que es una forma de representar números extremadamente grandes o pequeños. Esta notación es frecuente en entornos científicos y en *software* tales como los lenguajes de programación. En MATLAB, la notación científica se escribe como un número entre 1 y 10, seguido por la letra **`e`** y un exponente entero. Se presentan ejemplos con el número de Avogadro y la constante de Boltzmann.

Finalmente, se muestra cómo cambiar el formato de salida para que MATLAB muestre todos los decimales de una constante en notación científica. A pesar de que la salida se limita a cuatro decimales con el formato **`short`**, MATLAB en realidad almacena todos los decimales ingresados, como se demuestra cambiando el formato a **`long`**.

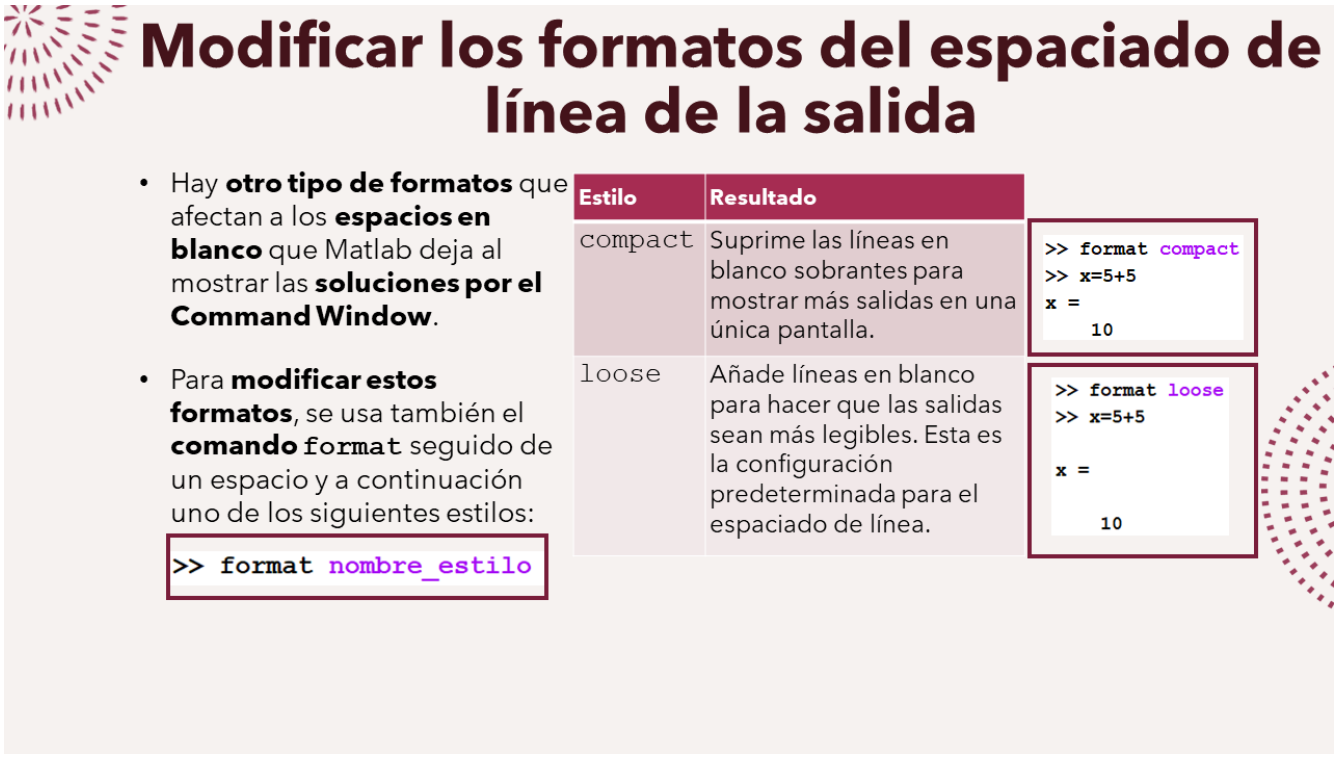


Figura 5.6 : Imagen de la miniatura usada en YouTube del vídeo: Formatos de salida de MATLAB y notación científica

## 5.1.7. Nombres permitidos y comando `help` en MATLAB

### Descripción de YouTube

En este vídeo se explica:

- Normas para dar nombre en MATLAB a variables, scripts o funciones.

- Diferencias entre un ERROR y un WARNING.
- Palabras reservadas en MATLAB que no se pueden usar como nombres de variables.
- Comando iskeyword para ver las palabras reservadas de MATLAB.
- Comando exists, para comprobar si existe una variable, script, función, carpeta o clase con un nombre determinado.
- Comando which, para localizar funciones y ficheros.
- Comando help, para que MATLAB nos muestre una pequeña ayuda de cualquier comando.

**Resumen**

En este vídeo del curso, se enseñan las reglas para nombrar variables, scripts o funciones, y se introduce el uso del comando **`help`**.

Estas reglas son:

1. Los nombres deben comenzar con una letra.
2. Los nombres pueden tener cualquier longitud, pero MATLAB solo guarda los primeros 63 caracteres.
3. Los únicos caracteres permitidos son letras, números y el guión bajo (_).
4. MATLAB distingue entre letras mayúsculas y minúsculas (case-sensitive).
5. Existen palabras reservadas que no pueden ser utilizadas como nombres.
6. MATLAB permite reasignar nombres de funciones internas como nombres de variables, pero es una práctica arriesgada y puede conducir a errores.

El vídeo enseña a reconocer y entender los mensajes de error y advertencia (*warnings*) en MATLAB que se generan cuando estas reglas no se cumplen.

Se destaca que MATLAB tiene 20 palabras reservadas que no deben ser usadas para nombrar elementos. Para conocerlas, se puede usar el comando **`iskeyword`**.

Para verificar si un nombre ya está en uso en MATLAB, se pueden utilizar los comandos **`exist`** y **`which`**; el primero devolverá un valor distinto dependiendo de si es una función, una variable o de si no existe en la memoria del programa entre otras posibilidades; mientras que el segundo mostrará la ubicación donde está guardada la función.

Finalmente, se introduce el comando **`help`**, que se utiliza para acceder a la documentación de las funciones de MATLAB. Al usar este comando seguido del nombre de una función u otro comando, MATLAB devolverá información relacionada con estos.

El vídeo concluye destacando que, en MATLAB, a pesar de lo que puede parecer, no existen constantes, incluso **`Pi`** es una función que devuelve el valor más cercano al número irracional **π**.

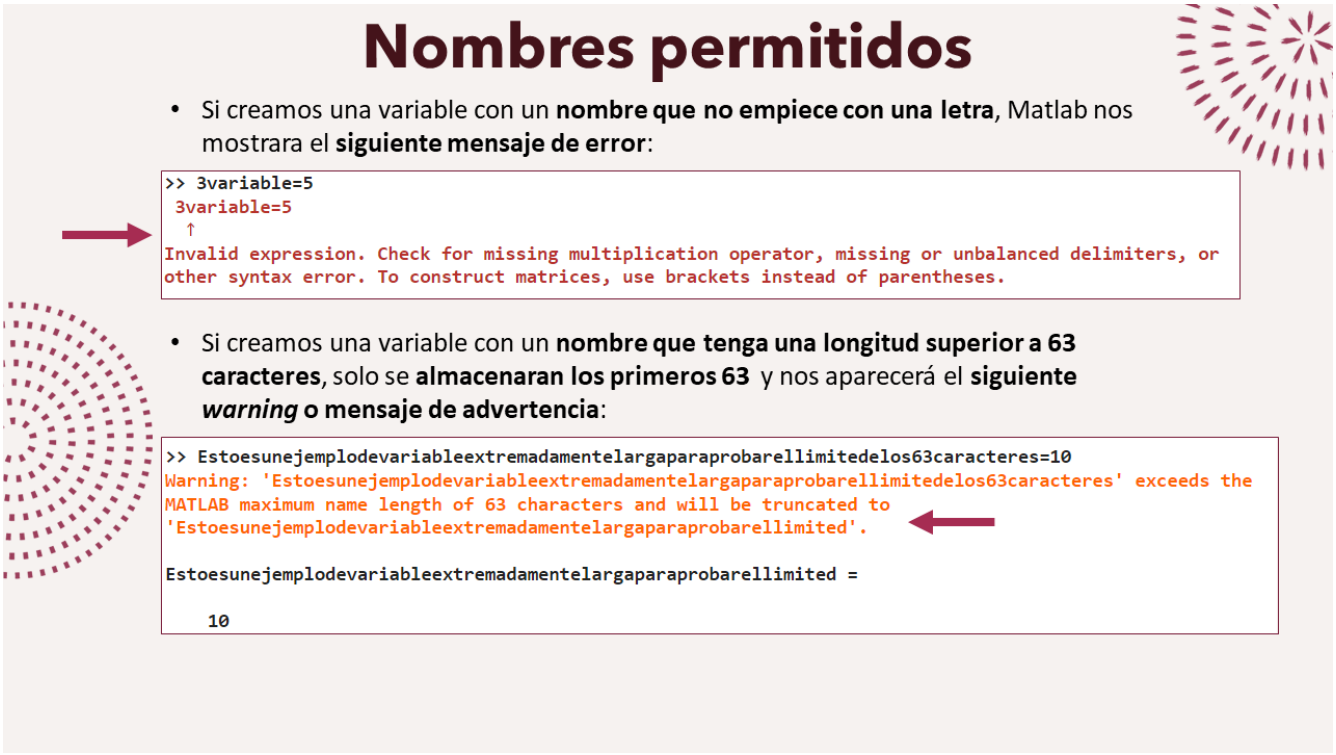


Figura 5.7 : Imagen de la miniatura usada en YouTube del vídeo: Nombres permitidos y comando `help` en MATLAB

## 5.1.8. Números complejos en MATLAB

### Descripción de YouTube

En este vídeo se explican los conceptos básicos para trabajar con números complejos en MATLAB:

- Creación de números complejos y función complex.
- Vectores y matrices de números complejos.
- Funciones básicas: real, imag, isreal y conj.
- Coordenadas polares: funciones abs y angle.
- Representación de números complejos: función plot para pintar el punto y también el vector con origen en el (0,0).

- Comando "hold on" para que al pintar lo siguiente no borre lo pintado anteriormente. Por defecto, al pintar algo, borra lo anterior.

**Resumen**

El vídeo es una lección introductoria sobre cómo utilizar números complejos en MATLAB. Comienza describiendo cómo los números complejos pueden ser usados en MATLAB para escalares, vectores y matrices, con ejemplos demostrativos para cada caso.

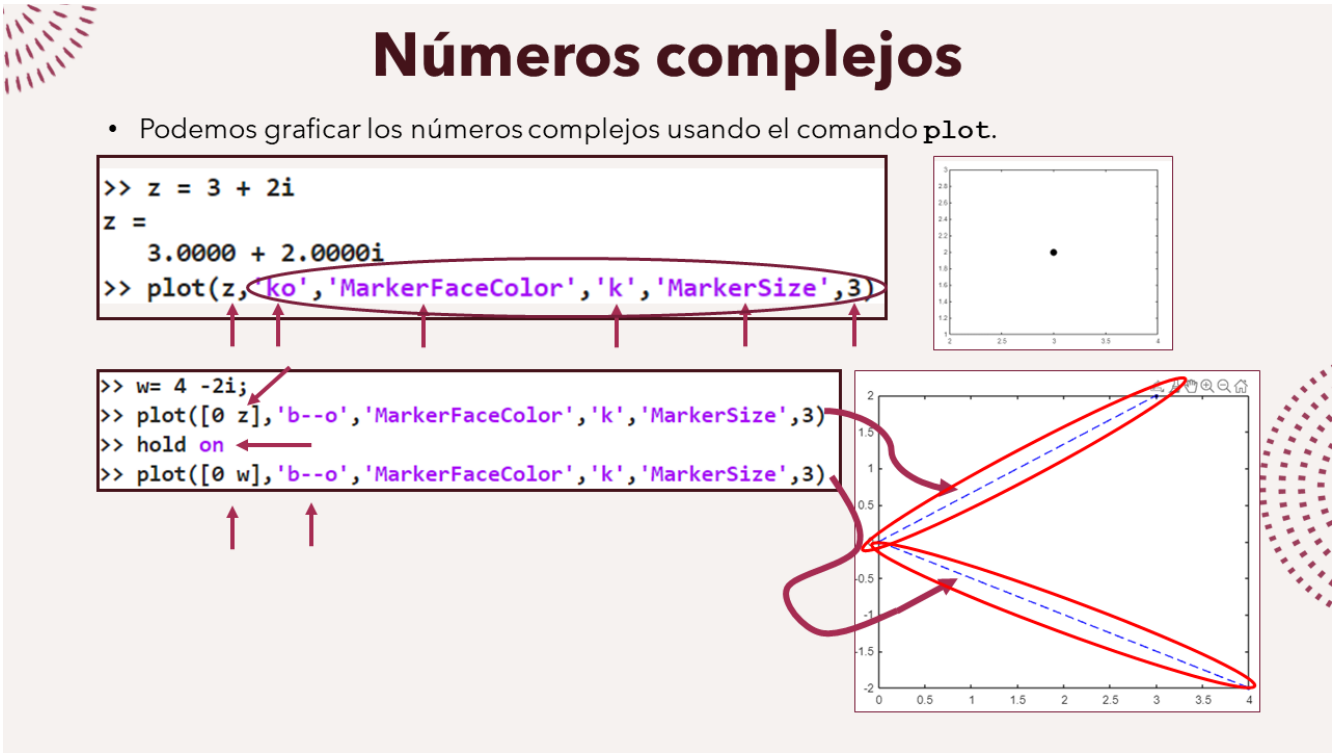


Figura 5.8 : Imagen de la miniatura usada en YouTube del vídeo: Números complejos en MATLAB

En el vídeo, se explica que los números complejos consisten en una parte real y una parte imaginaria, y se pueden representar en MATLAB utilizando los caracteres `i` o `j`. Se muestran tres métodos diferentes para definir el número complejo `5 + 4i`: usando el comando `complex` y de manera directa utilizando tanto el carácter `i` como `j`.

A continuación, se describen varios comandos que permiten manipular números complejos. Entre ellos se encuentran `real`, que extrae la parte real de un número complejo, `imag`, que extrae la parte imaginaria, `isreal`, que comprueba si un número es real, y `conj`, que devuelve el conjugado complejo de un número.

El vídeo también explica que los números complejos pueden representarse como puntos en un plano **XY**, y se pueden describir mediante coordenadas polares. Los

comandos **`abs`** y **`angle`** se usan para calcular el radio y el ángulo de un número complejo, respectivamente.

Finalmente, se enseña cómo graficar números complejos en MATLAB usando el comando **`plot`**. Se muestran ejemplos de cómo graficar un número complejo individual y cómo graficar dos vectores en una misma gráfica.

### 5.1.9. Funciones matemáticas elementales en MATLAB

**Descripción de YouTube**

Aquí se explican una serie de funciones matemáticas elementales de MATLAB. Estas funciones pueden ser usadas tanto con escalares reales o complejos como con matrices. Hay muchas más funciones y se recomienda que el alumno utilice el comando help para ampliar su conocimiento de las funciones explicadas y de otras. Estas funciones se pueden clasificar en la siguiente lista:

- Generales: abs, sqrt, nthroot, sign, rem, exp, log y log10.
- De redondeo: round, fix, floor y ceil
- Discretas: factor, gcd, lcm, factorial, primes y isprime.
- Trigonométricas (con el valor de entrada en radianes o en grados): sin, cos, tan, csc, sec, cot, sind, cosd, tand, cscd, secd y cotd (las que acaban en "d" funcionan en grados, degrees en inglés).
- Trigonométricas inversas (que devuelven el resultado en radianes o en grados): asin, acos, atan, acsc, asec, acot, asind, acosd, atand, acscd, asecd y acotd.
- Trigonométricas hiperbólicas: sinh, cosh, tanh, csch, sech y coth.
- Conversión de grados a radianes y de radianes a grados: rad2deg y deg2rad.

**Resumen**

El vídeo es una introducción al uso de funciones matemáticas en MATLAB.

Se destaca que MATLAB ofrece una extensa librería de funciones matemáticas, incluyendo funciones trigonométricas, logarítmicas, y de análisis estadísticos, que pueden trabajar tanto con números escalares (reales o complejos) como con matrices.

Se detalla que cada función en MATLAB está compuesta por tres partes: los argumentos de entrada, el nombre de la función y los argumentos de salida. Se menciona que MATLAB puede asignar automáticamente los argumentos de salida si estos no se especifican, y también se pueden utilizar funciones anidadas, donde la salida de una función

se usa como entrada de otra. Se explican varios ejemplos de uso de funciones, como el uso de la función **sin** para calcular el seno de un número. Además, se hace hincapié en el uso del comando **help** para obtener más información sobre las funciones.

Se presentan varias funciones matemáticas elementales que se pueden utilizar, como **abs** para el calcular el valor absoluto, **sqrt** para la raíz cuadrada, **nthroot** para la raíz enésima, **sign** para determinar el signo de un número. Además, se explican otras funciones como **rem** para el resto de una división, **exp** para el valor de una función exponencial, **log** para el logaritmo natural, **log10** para el logaritmo en base 10, y varias funciones para redondear números (**round**, **fix**, **floor**, **ceil**). También se describen algunas funciones para las matemáticas discretas, incluyendo **factor** para factores primos, **gcd** para el máximo común divisor, **lcm** para el mínimo común múltiplo, **factorial** para el factorial de un número, **primes** para obtener los números primos menores o iguales a un valor, y **isprime** para verificar si un número es primo.

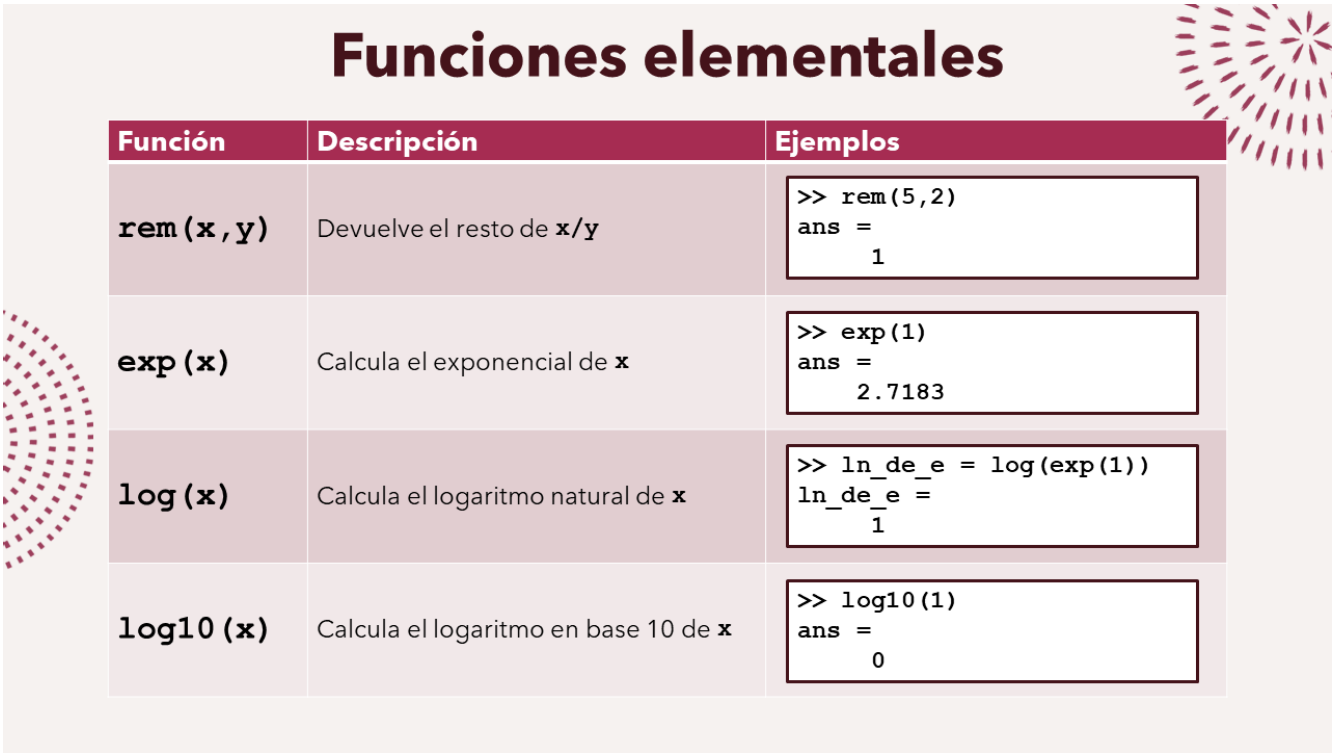


Figura 5.9 : Imagen de la miniatura usada en YouTube del vídeo: Funciones matemáticas elementales

Finalmente, se muestran diversas funciones trigonométricas e hiperbólicas, así como sus inversas, también se incluyen detalles sobre cómo convertir ángulos de radianes a

grados y viceversa, tanto mediante operaciones matemáticas como mediante las funciones **`rad2deg`** y **`deg2rad`**.

## 5.1.10. Resolviendo ecuaciones lineales con MATLAB

### Descripción de YouTube

Vamos a explicar cómo resolver sistemas de ecuaciones lineales en MATLAB, tanto normales (con tantas ecuaciones como incógnitas) como sobredeterminados e infradeterminados.

- CASO 1: Matriz cuadrada (nº ecuaciones = nº incógnitas).
  - ❖ det(A) para saber si el determinante es distinto de cero.
  - ❖ rank(A) para saber si la matriz es de rango completo.
  - ❖ Resolución del sistema con x = inv(A)*b; y con x = A\b
- CASO 2: Sistemas sobredeterminados (con más ecuaciones que incógnitas).
  - ❖ No se puede usar comando inv(A) para invertir la matriz A de coeficientes, porque A no es cuadrada.
  - ❖ Se debe usar x = A\b. Si no existe solución, nos da una que minimice el error por el método de mínimos cuadrados.
  - ❖ Gráfica de un sistema sobredeterminado de 3 ecuaciones y 2 incógnitas (3 rectas).
- CASO 3 : Sistemas infradeterminados (con menos ecuaciones que incógnitas).
  - ❖ Un sistema infradeterminado puede tener infinitas soluciones.
  - ❖ Con A\x obtenemos una solución, pero sin avisar que existen más soluciones.
- Al final del vídeo se explica el comando solve, que permite resolver sistemas de los casos 1 y 3, pero devuelve soluciones vacías en el caso 2, pues no encuentra ninguna solución exacta.
- ACLARACIÓN: Cuando el vídeo representa gráficamente 3 rectas, se calculan las coordenadas de una serie de puntos intermedios, lo cual no es necesario. Dado que son RECTAS, sería más eficiente calcular solo el punto inicial y final y unirlos con una recta (como hace plot). Si fueran curvas, SÍ sería necesario calcular los puntos intermedios con la precisión que se desee.

**Resumen**

Este vídeo enseña cómo resolver sistemas de ecuaciones algebraicas lineales con MATLAB. Comienza por establecer las bases teóricas detrás de los sistemas de ecuaciones y su representación matricial, la cual se realiza en la forma **Ax=b**.

A partir de aquí, el vídeo procede a mostrar cómo se puede solucionar un sistema lineal cuando la matriz de coeficientes es cuadrada, que es el caso más común. Los espectadores aprenden a verificar que ninguna fila de la matriz sea una combinación lineal de las otras. Esto se hace a través del cálculo del determinante de la matriz utilizando el comando **`det`** o verificando que el rango de la matriz sea igual a su dimensión con el comando **`rank`**.

Una vez que se establece que el sistema es cuadrado, el tutorial demuestra cómo encontrar una solución exacta a dicho sistema. Esto se logra invirtiendo la matriz y luego multiplicándola por el vector de términos independientes. Se presentan y discuten tres métodos para lograr esto en MATLAB: utilizando la función **`inv`**, el operador barra invertida `\`, o elevando la matriz a menos 1. El vídeo aconseja el uso de la barra invertida `\` por su mayor eficiencia y menor probabilidad de error.

Luego, se explora el caso de una matriz de coeficientes no cuadrada. En este caso, se abordan dos situaciones posibles: los sistemas sobredeterminados (con más ecuaciones que incógnitas) y los sistemas infradeterminados (con más incógnitas que ecuaciones). En el caso de los sistemas sobredeterminados, MATLAB emplea el método de mínimos cuadrados para encontrar una solución aproximada. En el caso de los sistemas subdeterminados, MATLAB ofrece una solución, aunque no advierte que existen múltiples soluciones posibles. El vídeo advierte que el uso de la función **`inv`** no es apropiado en estos casos ya que la matriz de coeficientes no es cuadrada. Sin embargo, se puede usar el operador barra invertida `\` para buscar una solución. En el vídeo titulado “Operaciones entre vectores y matrices” se explican también estos dos operadores de división, pero centrándose en operaciones particulares más que para resolver sistemas de ecuaciones.

Para cerrar, el tutorial introduce el uso del comando **`solve`** para resolver sistemas de ecuaciones. Para usar este comando, se deben definir primero las variables de la ecuación como simbólicas. Luego, se puede usar el comando con las ecuaciones como argumentos de entrada. Este comando solo se puede usar con los sistemas cuadrados o infradeterminados, pues con los sistemas sobredeterminados, como no existe una solución exacta, este comando devuelve soluciones vacías.

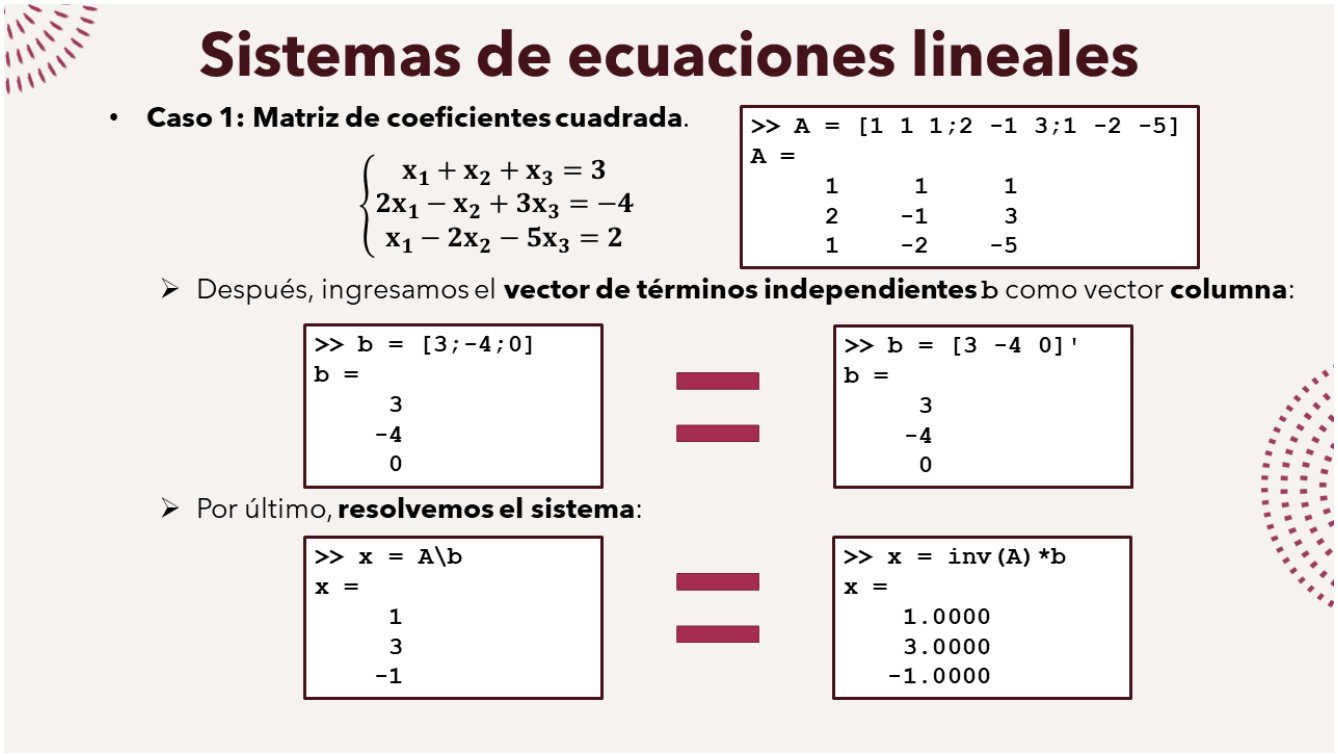


Figura 5.10 : Imagen de la miniatura usada en YouTube del vídeo: Resolviendo ecuaciones lineales con MATLAB

## 5.1.11. MATLAB: Cálculo simbólico para solucionar ecuaciones

### Descripción de YouTube

Vamos a explicar qué es el cálculo simbólico y cómo usarlo para resolver ecuaciones en MATLAB. Para poder utilizar esto se debe instalar el paquete Symbolic Math toolbox. Los conceptos que se abordan en esta clase se pueden enumerar en la siguiente lista:

- Creación de variables simbólicas con el comando syms.
- Mostrar con syms las variables simbólicas existentes.
- Creación de expresiones simbólicas.
- Uso de operadores relacionales para creación de ecuaciones e inecuaciones simbólicas.
- Función simplify para simplificar expresiones simbólicas.
- Función int para encontrar la integral indefinida de una expresión simbólica.
- Función diff para derivar una expresión simbólica.
- Función solve para encontrar los valores que anulan una expresión simbólica respecto de alguna variable simbólica.

- Función solve para encontrar los valores que cumplen una ecuación simbólica, respecto de alguna variable simbólica.
- Función solve junto con la opción returncondition para resolver inecuaciones simbólicas.
- Función eval para convertir resultados en formato simbólico a formato numérico.

**Resumen**

Este vídeo es una introducción a la realización de cálculos simbólicos en MATLAB, una funcionalidad que permite realizar operaciones propias del cálculo infinitesimal y del álgebra lineal, y que requiere del paquete opcional denominado **Symbolic Math Toolbox**.

Comienza explicando que los cálculos simbólicos son aquellos en los que intervienen expresiones matemáticas dependientes de variables y/o parámetros identificados por sus nombres. MATLAB puede calcular transformadas como Fourier, Laplace y Z utilizando estas operaciones.

Para realizar cálculos simbólicos, se utilizan objetos simbólicos. Hay dos comandos para crear variables simbólicas: **`sym`** y **`syms`**. En este curso, se introduce solo el comando **`syms`**, que es más apropiado para principiantes. Este comando se usa para crear variables simbólicas con una sintaxis específica: primero se escribe **`syms`**, seguido de un espacio y luego las variables simbólicas que se quieren crear.

Las variables simbólicas deben ser declaradas antes de su uso, a diferencia de las variables numéricas. Cada argumento de entrada o variable simbólica debe comenzar con una letra y solo puede contener caracteres alfanuméricos.

El vídeo también muestra cómo se pueden crear expresiones y ecuaciones simbólicas con estas variables. Una vez creadas las variables simbólicas, se pueden crear expresiones simbólicas que se pueden usar directamente como argumento de entrada o almacenar en una variable. Si se asigna la simbólica a una variable, esa variable pasa a ser considerada también variable simbólica.

Se muestra cómo las ecuaciones simbólicas pueden crearse utilizando operadores relacionales. Se enfatiza que el operador de igualdad se escribe con dos iguales, y no con uno solo, ya que el operador de asignación se escribe con un solo igual.

Se introduce el comando **`simplify`**, que simplifica la expresión simbólica ingresada como argumento. Además, se presentan los comandos **`int`** y **`diff`** que calculan,

respectivamente, las integrales definidas e indefinidas y las derivadas de expresiones simbólicas.

En la segunda mitad del vídeo, se discute la función **`solve`**. Esta función puede usarse para encontrar los valores de las variables que hacen que una expresión o ecuación simbólica sea igual a cero. También puede usarse para resolver ecuaciones simbólicas para variables específicas.

Se discute la necesidad de activar la opción **`returnConditions`** cuando se usa **`solve`** con inecuaciones, y cómo acceder a las condiciones de una solución.

Finalmente, el vídeo destaca que todas las funciones mencionadas devuelven expresiones simbólicas. Para convertir estas expresiones a formato numérico, se debe utilizar el comando **`eval`**. Este comando convierte las expresiones simbólicas a formato numérico para realizar cálculos numéricos.

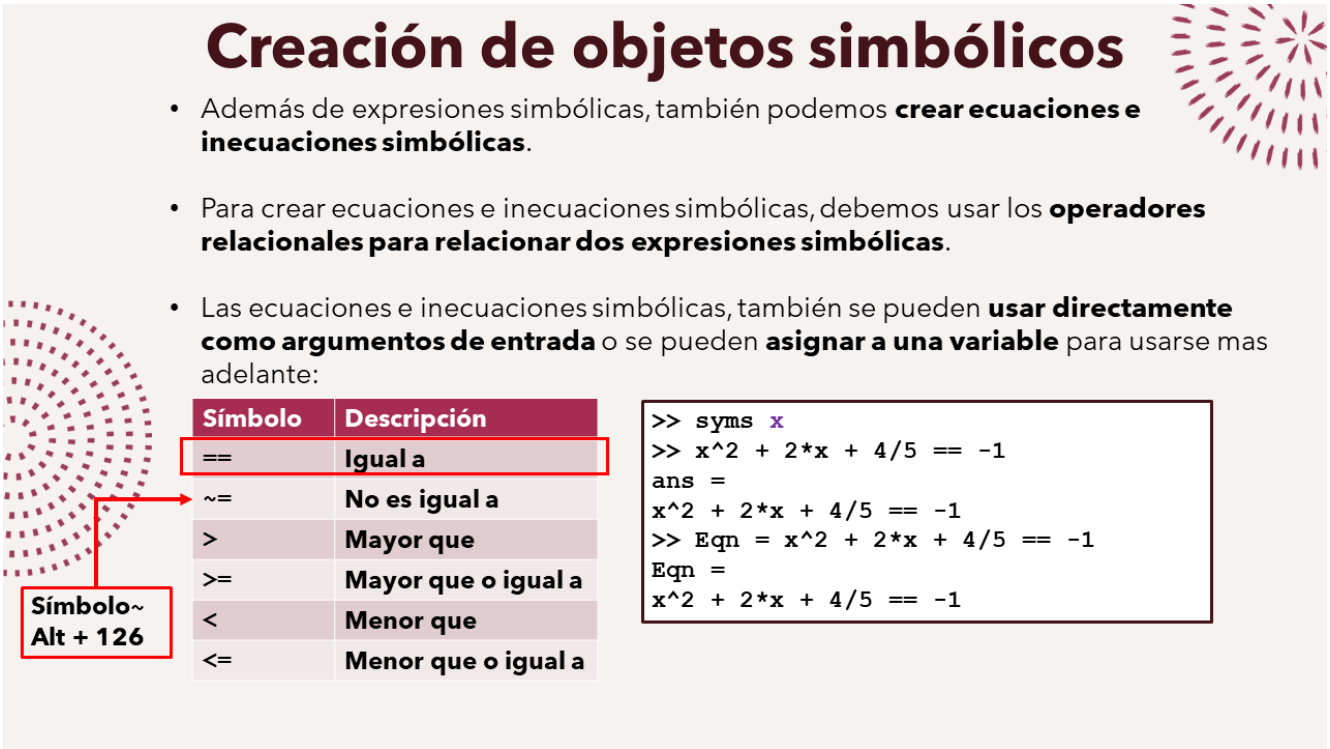


Figura 5.11 : Imagen de la miniatura usada en YouTube del vídeo: MATLAB: Cálculo simbólico para solucionar ecuaciones

## 5.1.12. Polinomios en MATLAB

### Descripción de YouTube

El vídeo explica cómo representar y trabajar con polinomios en MATLAB. Los conceptos que se abordan se enumeran en la siguiente lista:

- ¿Qué es un polinomio?

- ¿Cómo crear polinomios en MATLAB?
- Sumar y restar polinomios.
- Una lista de funciones que MATLAB ofrece para trabajar con polinomios, de la cual explicaremos solo algunas funciones.
- Función polyval(p,x) para evaluar polinomios.
- Función roots(p) para obtener las raíces de un polinomio.
- Función poly(r) para obtener los coeficientes de un polinomio a partir de sus raíces.
- Función poly(A) para obtener los coeficientes del polinomio característico de una matriz cuadrada.
- Función conv(u,v) para multiplicar dos polinomios.
- Función [q,r] = deconv(u,v) para dividir dos polinomios y obtener el cociente y el resto.
- Función polyint(p) para integrar un polinomio.
- Función polyder(p) para derivar un polinomio.

**Resumen**

El vídeo es un curso introductorio sobre cómo representar y manipular polinomios en MATLAB.

A continuación, se resume la información más relevante:

1. **Representación de Polinomios:** MATLAB representa polinomios como vectores fila que contienen los coeficientes ordenados en orden descendente de potencia. Por ejemplo, el polinomio **$3x^2 - 8x + 5$** se representa como `[3 -8 5]`. Los ceros que pueden aparecer en el vector corresponden a las potencias cuyos coeficientes son cero en la ecuación del polinomio.
2. **Adición y Sustracción de Polinomios:** Para sumar o restar polinomios en MATLAB, todos los polinomios deben tener la misma cantidad de coeficientes. Para ello se añaden ceros a la izquierda en los vectores de los polinomios que tengan menos coeficientes para igualarlos.

3. **Evaluación de Polinomios:** MATLAB ofrece la función **`polyval`** para evaluar polinomios. Esta función toma como argumentos el vector de coeficientes del polinomio y el valor en el que se desea evaluar.
4. **Raíces de Polinomios:** La función **`roots`** encuentra las raíces de un polinomio y devuelve un vector columna con las mismas. Las raíces son los valores que anulan el polinomio.
5. **Polinomio desde Raíces:** La función **`poly`** permite obtener el polinomio a partir de sus raíces.
6. **Convolución y División de Polinomios:** MATLAB proporciona las funciones **`conv`** y **`deconv`** para la convolución (multiplicación) y la división de polinomios, respectivamente.
7. **Integración y Diferenciación de Polinomios:** Las funciones **`polyint`** y **`polyder`** se usan para integrar y derivar polinomios, respectivamente; **`polyint`** toma como argumentos, primero el vector fila del polinomio y luego una constante opcional de integración (que se considera cero si no se especifica); **`polyder`** toma el vector fila del polinomio y devuelve el vector de coeficientes de la derivada.

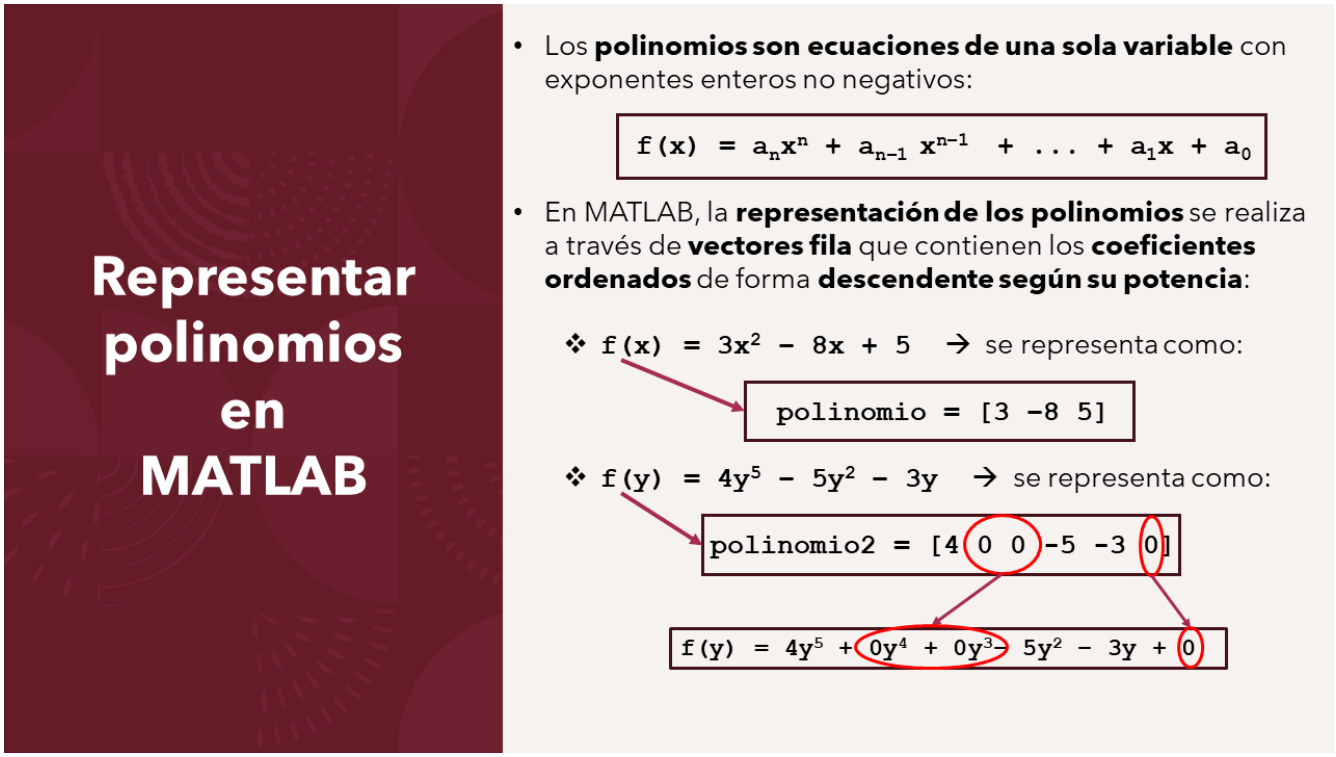


Figura 5.12 : Imagen de la miniatura usada en YouTube del vídeo: Polinomios en MATLAB

### 5.1.13. MATLAB: Problema de cálculo simbólico sobre el crecimiento bacteriano

**Descripción de YouTube**

En este vídeo vamos a resolver un problema sobre el crecimiento bacteriano en un sistema de aire acondicionado siguiendo la ley de Malthus.

Hemos medido que la población de bacterias se ha duplicado en 4 horas y queremos descubrir cuánto tiempo será necesario para que la población se multiplique por 6.

- Se explica la fórmula de la ley de Malthus.
- Se declaran variables simbólicas con syms.
- Se usa la función solve para encontrar la constante de proporcionalidad y para resolver la ecuación final.
- Se usa la función eval para que el resultado sea un número real (no simbólico).
- Se usa la función fprintf para escribir el mensaje final en el script.
- Se explica el especificador de formato %f y el carácter \n usados en la función de escritura fprintf.

**Resumen**

Este vídeo enseña cómo resolver un problema de crecimiento bacteriano que sigue la ley de Malthus en un sistema de aire acondicionado.

El problema planteado consiste en que, en cierta maquinaria de aire acondicionado, se ha detectado un cultivo de bacterias cuyo crecimiento es proporcional y sigue las leyes de Malthus. Se ha medido que la población se duplica en 4 horas, y el objetivo es calcular cuánto tiempo tardará la población en multiplicarse por 6.

Para resolverlo, se siguen dos pasos:

1. Encontrar la constante de proporcionalidad, r. Se establece la ecuación de Malthus usando la información proporcionada de que la población se duplica en 4 horas y se resuelve para encontrar el valor de r.
2. Una vez se tiene el valor de r, se usa la ecuación de Malthus nuevamente para encontrar el tiempo en el que la población se multiplica por 6.

Una vez establecido el problema, se utiliza MATLAB para resolverlo. Primero, se declara `r` como una variable simbólica y se crea una ecuación simbólica usando la ecuación

de Malthus y la información proporcionada sobre el tiempo en que la población se duplica. Luego, se resuelve la ecuación para `r` con la función `solve` y se convierte el resultado en un número decimal mediante la función `eval`.

El proceso es similar para encontrar el tiempo en que la población se multiplica por 6, pero usando esta vez el valor de `r` previamente calculado. El resultado se muestra en la pantalla utilizando la función `fprintf`.

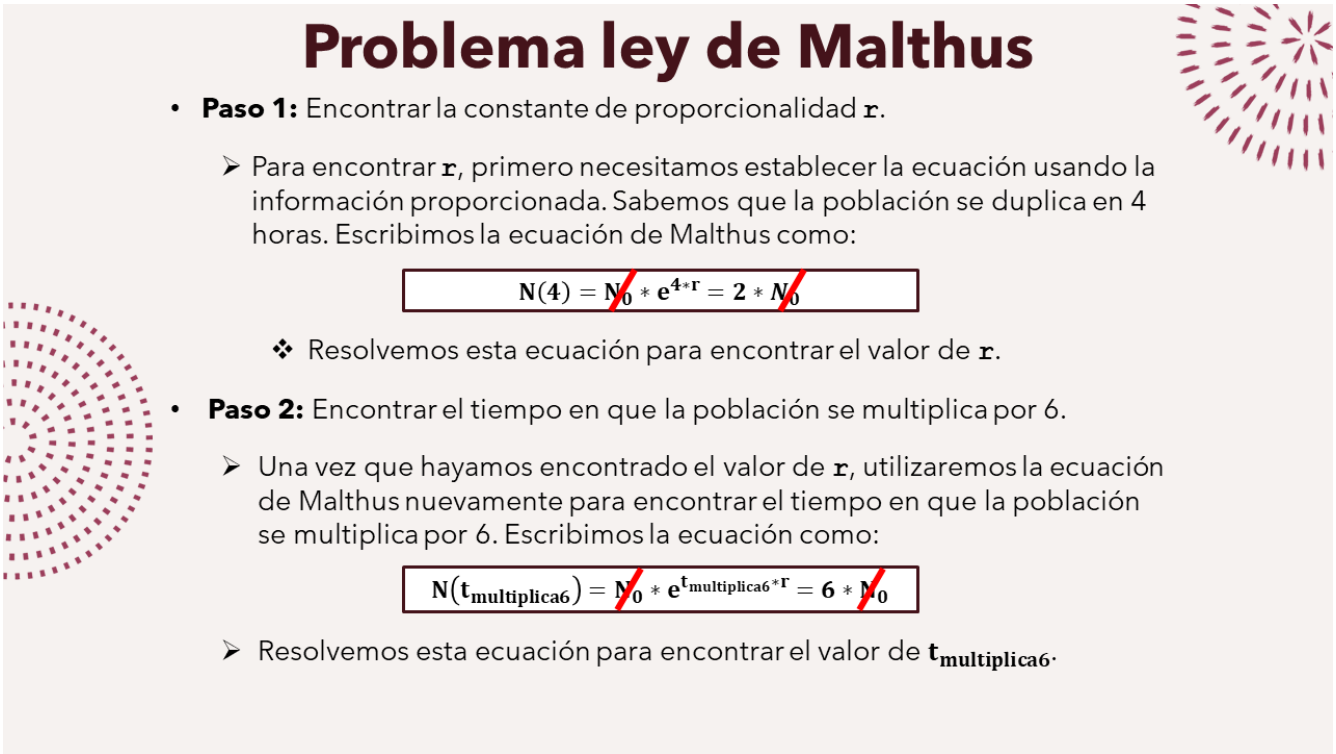


Figura 5.13 : Imagen de la miniatura usada en YouTube del vídeo: MATLAB: Problema de cálculo simbólico sobre el crecimiento bacteriano

## 5.1.14. MATLAB: Crear vectores y matrices

### Descripción de YouTube

En este vídeo vamos a estudiar cómo crear vectores y matrices. Un vector es, de hecho, una matriz en la que una de sus dimensiones es 1. Hablamos de vector fila cuando es 1 la primera dimensión (número de filas). Por su parte, un vector columna tiene a 1 su segunda dimensión (número de columnas).

- Los conceptos que se abordarán se pueden enumerar en la siguiente lista:
- Creación de vectores fila y columna desde el teclado.
- Vector vacío [ ].
- Creación de matrices desde teclado.
- Operador apóstrofo ' para transponer tanto vectores como matrices.

- Creación de vectores vacíos desde teclado.
- Funciones para crear ciertos tipos de vectores y matrices: funciones eye, zeros, ones, rand, randn, magic, linspace y logspace.

**Resumen**

Este vídeo se enfoca en cómo crear vectores y matrices. Comienza resaltando algunas convenciones recomendadas para nombrar variables, indicando que las matrices deben usar nombres en mayúsculas y los vectores y escalares en minúsculas. Menciona que al definir una matriz o un vector, no es necesario especificar su tamaño, ya que se puede ajustar en cualquier momento.

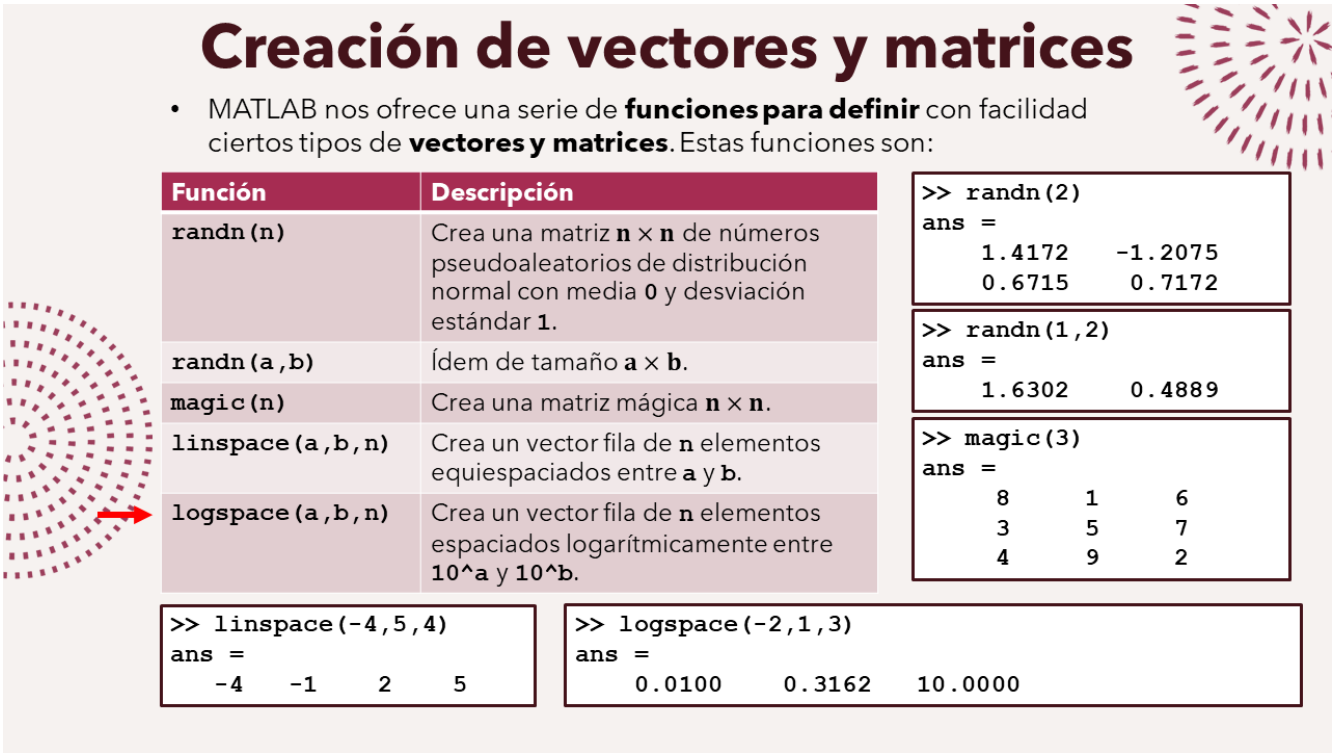


Figura 5.14 : Imagen de la miniatura usada en YouTube del vídeo: MATLAB: Crear vectores y matrices

Se explica cómo definir vectores fila y columna en MATLAB, así como la posibilidad de transponerlos. Se describen los vectores vacíos y se destaca su utilidad para eliminar elementos o incluso filas y columnas completas en futuras manipulaciones.

En cuanto a las matrices, se describe su definición fila por fila y la transposición usando el mismo operador apóstrofe utilizado con vectores.

El vídeo también explora varias funciones útiles en MATLAB para crear vectores y matrices de manera eficiente cuando su tamaño es más grande. Se introducen funciones

como **`eye`** para crear matrices de identidad, **`zeros`** y **`ones`** para matrices con todos sus elementos cero o uno respectivamente, **`rand`** y **`randn`** para matrices con números aleatorios de distribución uniforme o normal, **`magic`** para matrices mágicas, y **`linspace`** y **`logspace`** para vectores con elementos equiespaciados o espaciados logarítmicamente.

## 5.1.15. MATLAB: Composición de vectores y matrices

### Descripción de YouTube

En este vídeo se explica cómo podemos "componer" nuevos vectores y matrices a partir de otros vectores y matrices. Los conceptos que se abordan en esta clase se pueden enumerar en la siguiente lista:

- Composición de vectores fila o columna a partir de otros vectores.
- Trasponer vectores y matrices con el operador apóstrofo (').
- Composición de matrices a partir de vectores o matrices.
- Posibles errores que pueden darse al no encajar bien los elementos que usamos para componer un vector o matriz.
- Funciones para crear vectores y matrices con ciertos valores: eye, zeros, ones, rand, diag y blkdiag.
- Funciones sobre el tamaño o dimensión: size y length.
- Funciones para extraer la matriz triangular superior e inferior: triu y tril.
- Funciones para reflejar matrices usando un eje central horizontal o vertical: flipud y fliplr.

### Resumen

Este vídeo aborda cómo componer vectores y matrices en MATLAB a partir de otros vectores y matrices previamente creados. Explica que las dimensiones de los vectores y matrices deben ser coherentes para evitar errores.

Se muestra cómo crear vectores a partir de otros vectores, separando los que pertenecen a la misma fila con comas o espacios en blanco y a diferentes filas con puntos y comas. Se ilustra con ejemplos, incluyendo casos donde se requiere transponer un vector para lograr una concatenación coherente y casos en los que MATLAB devuelve un error debido a la incoherencia en las dimensiones.

La composición de matrices también se aborda, enfatizando que las dimensiones deben ser coherentes para evitar errores. Se muestra cómo crear una matriz concatenando un vector fila y un vector columna, y cómo crear matrices a partir de otros vectores y matrices.

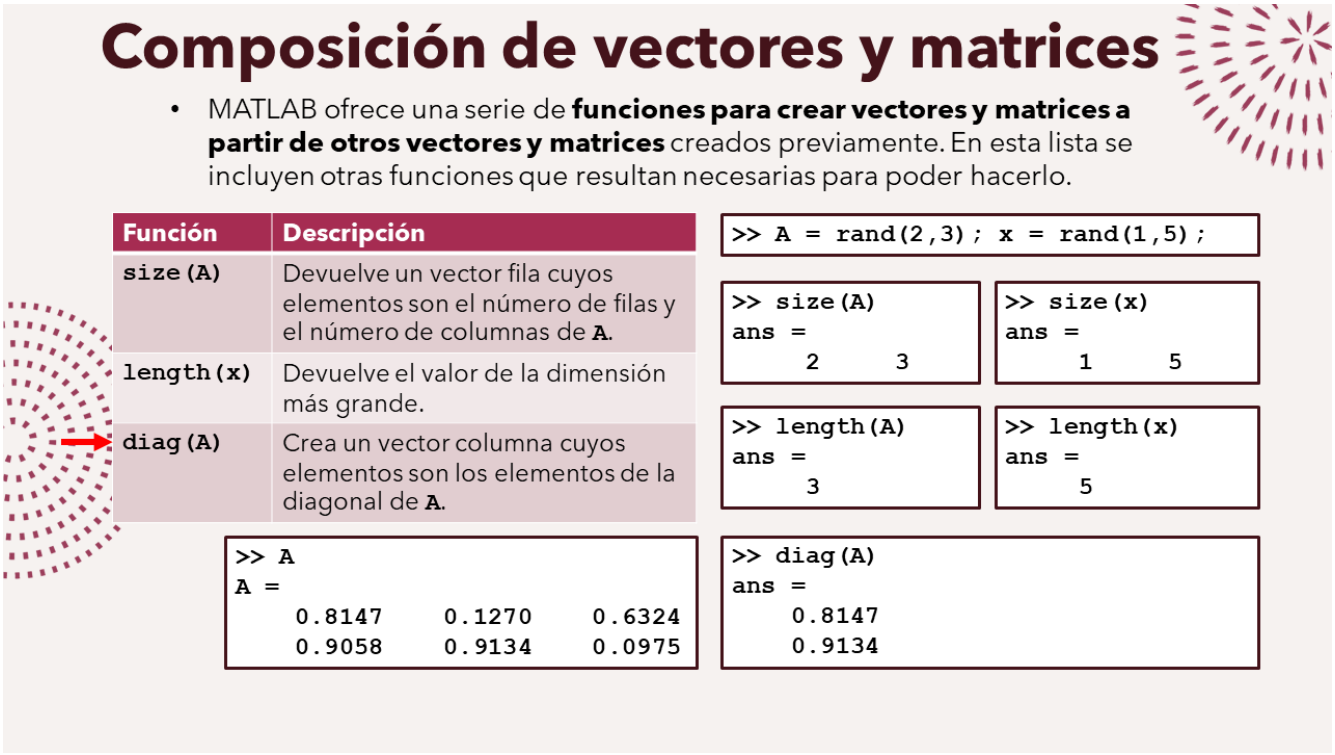


Figura 5.15 : Imagen de la miniatura usada en YouTube del vídeo: MATLAB: Composición de vectores y matrices

Se incluyen ejemplos de errores de MATLAB debido a la incoherencia en las dimensiones de las matrices, y también se ofrece un ejemplo más complejo de cómo crear una matriz a partir de matrices y vectores.

El vídeo también presenta varias funciones útiles. La función **`size`** que devuelve las dimensiones de una matriz o vector, **`length`** devuelve la longitud del vector o la mayor dimensión de la matriz, y **`diag`** devuelve los elementos de la diagonal principal de la matriz o crea una matriz cuadrada a partir de un vector.

Se muestra cómo usar estas funciones en conjunto con **`zeros`** y **`ones`** para crear matrices de ceros o unos del mismo tamaño que otra matriz. También se introduce la función **`blkdiag`** que crea una matriz diagonal a partir de varias matrices.

Además, se explican las funciones **`triu`**, **`tril`**, **`flipud`**, y **`fliplr`** que respectivamente crean una matriz triangular superior, una matriz triangular inferior, y matrices simétricas respecto del eje horizontal y vertical de la matriz de entrada.

## 5.1.16. MATLAB: Direccionar y modificar vectores y matrices

**Descripción de YouTube**

En este vídeo vamos a ver cómo se pueden direccionar los elementos de un vector o una matriz. Es decir, veremos cómo acceder a los valores de un vector o matriz, tanto de uno en uno, como en grupo. También veremos cómo modificar dichos elementos.

Las posiciones de los elementos empiezan en el valor 1 (y no en el valor cero como en algunos lenguajes de programación, como en C/C++). Con un poco más de detalle, los conceptos que explicamos aquí son:

- Acceso a un único elemento de un vector fila o columna.
- Acceso a varios elementos de un vector mediante otro vector.
- Acceso a un único elemento de una matriz, tanto con dos subíndices (fila,columna) como con un único subíndice, aprovechando que las matrices se guardan en memoria por columnas.
- Acceso a las filas y las columnas de una matriz mediante el uso de vectores.
- Modificación de los elementos de un vector o una matriz.
- Eliminación de elementos de un vector.
- Eliminación de filas y columnas de una matriz.

**Resumen**

El vídeo se centra en cómo acceder y modificar los elementos de un vector o una matriz.

En MATLAB, la numeración de las posiciones comienza desde 1, en contraposición a otros lenguajes de programación que comienza desde 0 (como C o C++, por ejemplo).

Para acceder a un elemento de un vector, se usa el subíndice de su posición. Los elementos de un vector también se pueden acceder utilizando los elementos de otro vector como índices.

Para acceder a los elementos de una matriz, se deben especificar los subíndices de la fila y la columna. También es posible acceder a los elementos de una matriz utilizando un solo índice, teniendo en cuenta que MATLAB almacena las matrices en memoria por

columnas. Además, es posible acceder a elementos de una matriz utilizando dos vectores de índices, seleccionando todos los elementos que cumplan ambas condiciones.

Para modificar los elementos de un vector o una matriz, se deben direccionar los elementos y asignarles un nuevo valor usando el operador de asignación. Se muestra un ejemplo en el que se modifica una matriz identidad 3x3, cambiando algunos de sus elementos.

El vídeo también cubre cómo eliminar filas, columnas o elementos específicos de una matriz o un vector. Primero se deben direccionar los elementos o filas/columnas que se desean eliminar y luego asignarles un vector vacío. Sin embargo, en el caso de las matrices, no se pueden eliminar elementos específicos ya que esto rompería la coherencia de su tamaño.

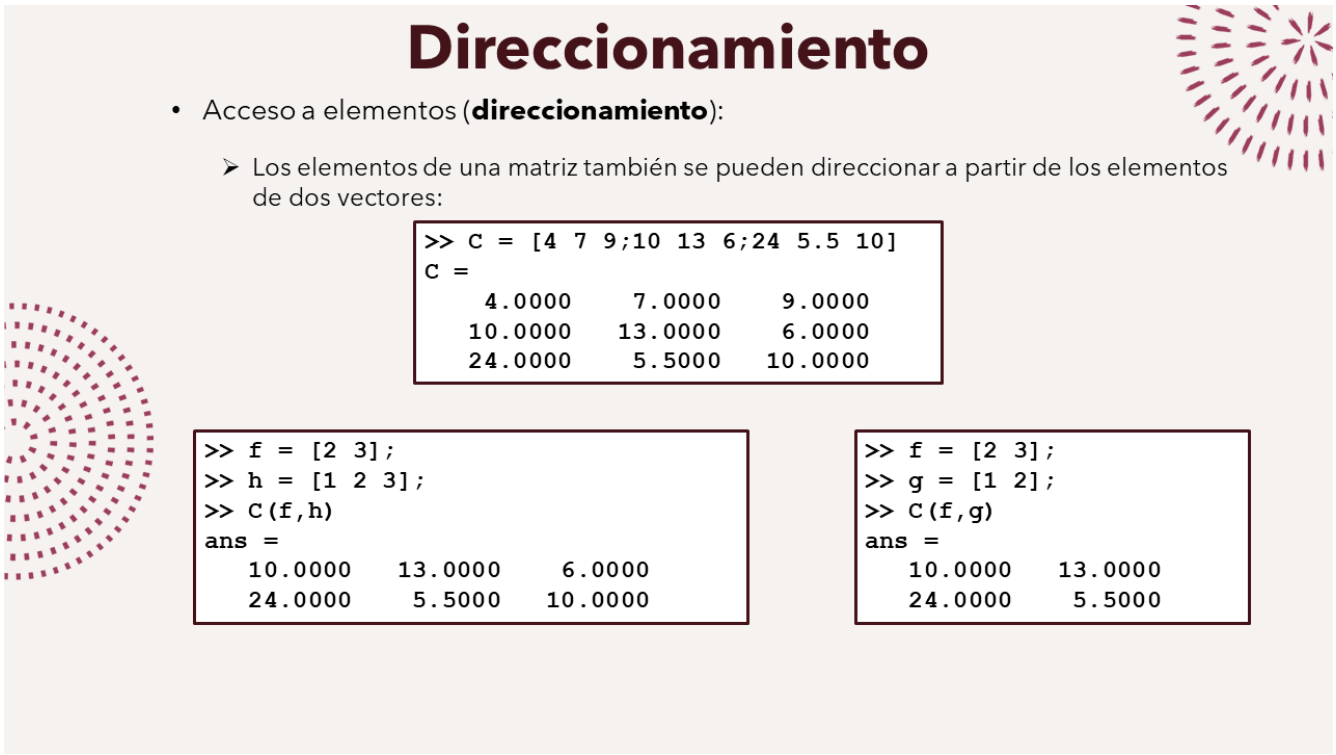


Figura 5.16 : Imagen de la miniatura usada en YouTube del vídeo: MATLAB: Direccionar y modificar vectores y matrices

## 5.1.17. MATLAB: Operador dos puntos (:)

### Descripción de YouTube

Aquí se explica el operador dos puntos de MATLAB. Es un operador muy importante que puede usarse de múltiples formas.

En resumen, se puede decir que este operador sirve para dos cosas:

1. Crear un vector fila entre un punto inicial (a) y un punto final (b) con saltos predeterminados. Si no se indica el salto se considera que es de 1 en 1. También se pueden usar con saltos negativos (cuando el primer valor es mayor que el segundo.
    - Ejemplo: El vector a:n:b, representa los valores entre a y b, de n en n ( n es el salto o incremento).
2. Cuando se usa es operador para direccionar sin indicar el valor inicial ni el final, eso quiere decir que se seleccionan todos los elementos del vector o de la matriz, o bien todas las filas o todas las columnas de la matriz (dependiendo de la posición que ocupe el operador).
    - Ejemplo: La expresión M( : , 3) se refiere a todas las filas de la columna 3, es decir, es un vector columna con los elementos de la tercera columna de M.

En ambos casos, veremos que este operador resulta útil usarlo para direccionar los elementos de un vector o de una matriz.

Por otra parte, para indicar el último elemento de un vector o de una fila/columna de una matriz, se puede usar la palabra reservada “end”. Esta palabra combinada con el operador dos puntos nos ofrece una poderosa herramienta, como muestran los ejemplos que se incluyen en el vídeo.

También veremos cómo darle la vuelta a un vector y a las columnas de una matriz.

**Resumen**

Este vídeo ofrece una visión detallada del operador (carácter **`:`**) en MATLAB, destacando su versatilidad para generar vectores y seleccionar elementos específicos en vectores y matrices. Se discute cómo este operador se emplea para generar vectores. Una ilustración de esto es **`x = a:b`**, que genera un vector que va desde el valor **`a`** hasta el valor **`b`**, incrementando en uno cada vez. Por otro lado, **`x = a:n:b`** produce un vector que se extiende desde **`a`** hasta **`b`** con incrementos de **`n`**.

Además, el vídeo menciona que, si el incremento **`n`** es negativo, el vector se llenará en un orden descendente, y se proporcionan ejemplos de este uso.

El operador dos puntos también se puede usar en la selección de elementos en matrices y vectores. Como tal, **`D(a:n:b)`** elige elementos de la matriz **`D`** que van desde la posición **`a`** hasta la posición **`b`**, con un incremento de **`n`** en cada paso. Sin embargo, cuando el operador dos puntos se usa solo, como en **`v(:)`**, selecciona todos los elementos presentes en **`v`**, devolviendo un vector columna.

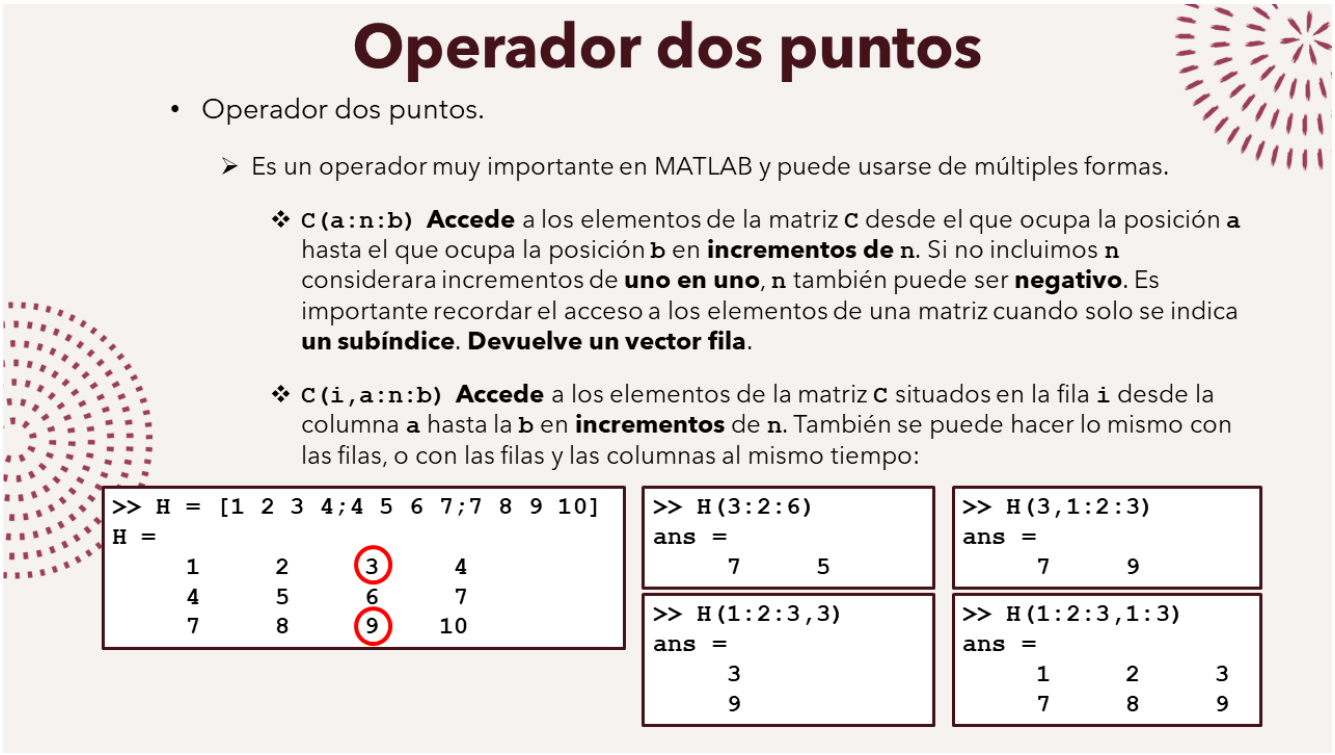


Figura 5.17 : Imagen de la miniatura usada en YouTube del vídeo: MATLAB: Operador dos puntos (`:`)

Al seleccionar todas las filas o columnas de una matriz, se utilizan las sintaxis **`a(:,j)`** y **`A(i,:)`** respectivamente. Y si se utiliza dos veces el operador, como en **`A(:,:)`**, se seleccionará toda la matriz **`A`**.

En situaciones en las que se necesita hacer referencia al último elemento de un vector o matriz, se puede utilizar la palabra reservada **`end`**. Esto se convierte en una herramienta útil cuando se desconoce la longitud del vector o el tamaño de la matriz.

Por último, el vídeo demuestra ejemplos de cómo voltear un vector y reorganizar las columnas de una matriz utilizando estas técnicas.

## 5.1.18. MATLAB: Ajuste polinomial a partir de un conjunto de puntos

### Descripción de YouTube

En este vídeo vamos a estudiar cómo aproximar un conjunto de puntos 2D (del plano XY) con un polinomio. Después, como ejercicio, veremos cómo calcular el error cometido durante esta aproximación. Con un poco más de detalle, lo que se explica es:

- Función polyfit para hallar un polinomio que aproxime un conjunto de puntos.

- Calcular el error cometido en la aproximación mediante dos fórmulas: error cuadrático medio (RMSE) y error absoluto medio (MAE).
- Representación gráfica (con plot) de los puntos y del polinomio.
- En el ejemplo, calcularemos los puntos usando una función y veremos como esa función coincide con el polinomio en el intervalo de los puntos, pero ambas funciones se distancian fuera del mismo.

**Resumen**

El objetivo de este vídeo es enseñar a los espectadores cómo ajustar un polinomio a un conjunto de puntos del plano **XY** y cómo calcular el error resultante en MATLAB.

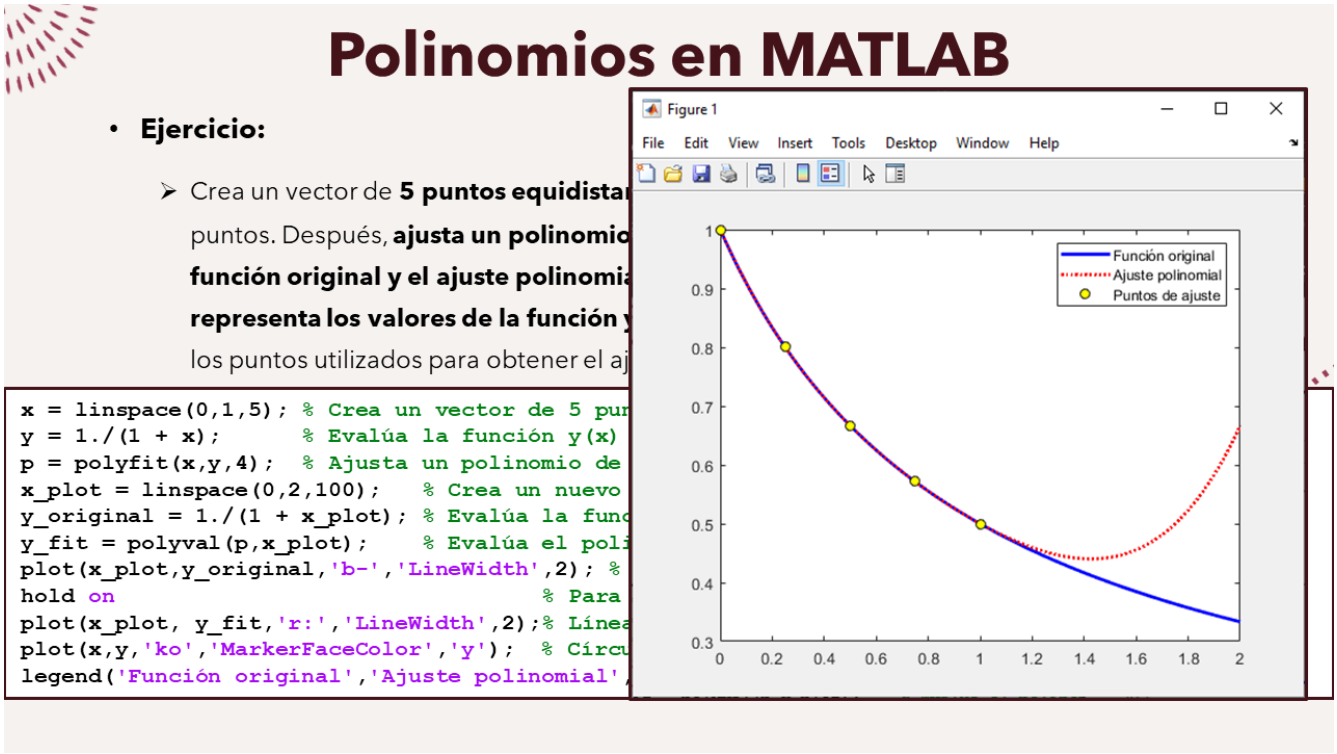


Figura 5.18 : Imagen de la miniatura usada en YouTube del vídeo: MATLAB: Ajuste polinomial a partir de un conjunto de puntos

Se utiliza la función **`polyfit`** para ajustar un conjunto de datos a un polinomio utilizando el método de mínimos cuadrados. El vídeo describe la sintaxis de la función, que incluye vectores de entrada para los ejes **X** e **Y**, el grado del polinomio y los coeficientes del polinomio ajustado.

Luego se propone un ejercicio para ilustrar el uso de la función **`polyfit`**. El ejercicio implica la creación de un vector de puntos, la evaluación de una función en esos puntos, la

adaptación de un polinomio a esos puntos y la representación de la función y el polinomio ajustado en un intervalo más amplio.

Después de completar el ejercicio, se muestra cómo calcular el error en la aproximación utilizando el error cuadrático medio (**RMSE**, por sus siglas en inglés) y el error absoluto medio (**MAE**, por sus siglas en inglés).

### 5.1.19. MATLAB: Operaciones de escalares con vectores y matrices

**Descripción de YouTube**

En este vídeo vamos a ver las operaciones básicas que se pueden realizar entre un escalar y un vector, o bien, entre un escalar y una matriz. Denotamos k como el escalar y v como un vector o matriz.

Los conceptos que se van a tratar en este vídeo son los siguientes:

- Suma, resta, multiplicación y división de un escalar con cada uno de los elementos de un vector o matriz.
- Para la resta y la división (operaciones no conmutativas) se verán ambas posibilidades. Por ejemplo, para la resta: (k - v) y también (v - k).
- Algunas operaciones requieren del operador punto para indicar a MATLAB explícitamente que es una operación elemento a elemento.
- Por ejemplo, la división de un escalar entre cada uno de los elementos de un vector o una matriz necesita del operador punto. En caso contrario, se genera un error, porque las dimensiones del escalar y del vector o matriz no son compatibles. Es decir, para operar elemento a elemento, v/k es una operación aceptada por MATLAB, pero no lo es k/v, operación que debe usar el operador punto: k./v
- Potencia, elemento a elemento: Elevar un escalar a cada uno de los elementos de un vector o una matriz, o viceversa, son dos operaciones que también necesitan del operador punto. Se producirá un error si no se pone el punto, salvo que v sea una matriz cuadrada en cuyo caso la operación NO se efectuará elemento a elemento, sino que tendrá otro significado. Así, v^k es la matriz v multiplicada por sí misma k veces.
- Para más información sobre el significado matemático de elevar un escalar a una matriz cuadrada, recomendamos este vídeo:

https://www.youtube.com/watch?v=8xxjyM2T2Dc&t=0s&ab_channel=matematicas-net

**Resumen**

El vídeo presenta una introducción a MATLAB, enfocándose en las operaciones que se pueden realizar entre un escalar y un vector o matriz. Se utiliza **k** para representar un escalar y **v** para un vector o matriz.

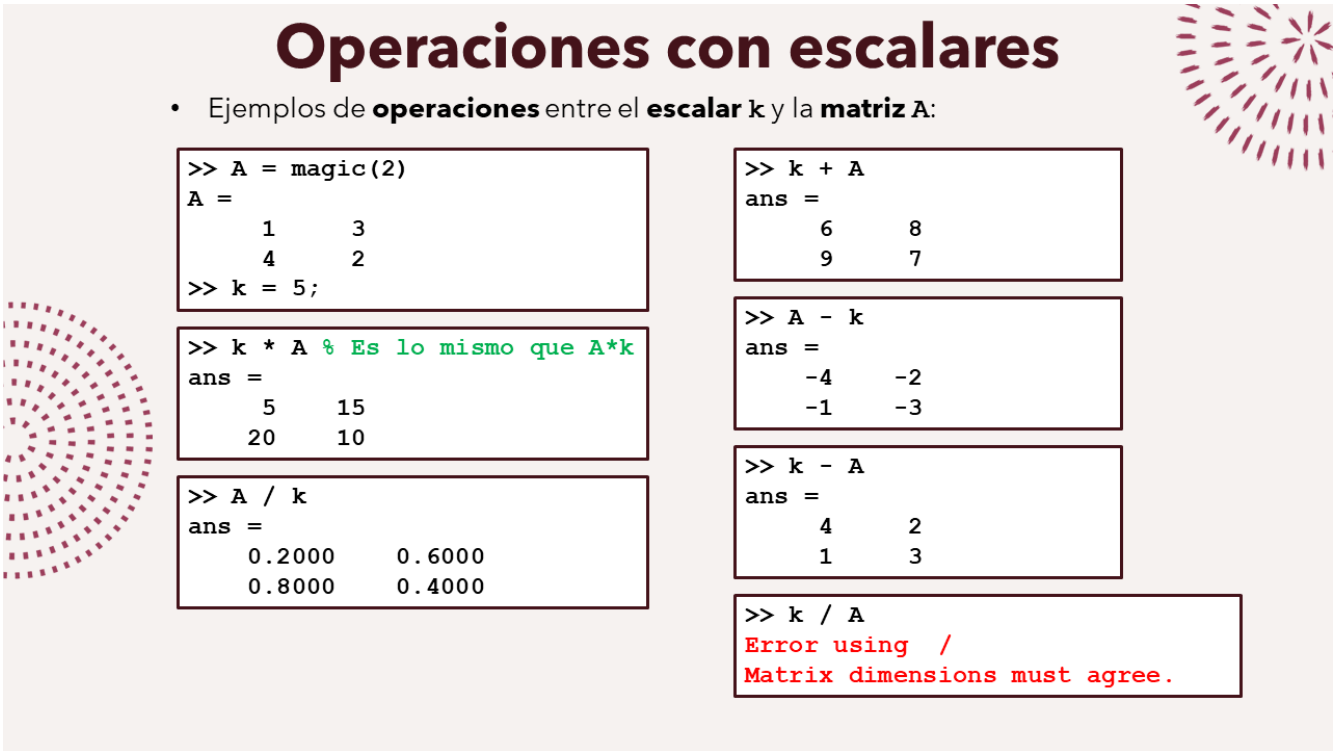


Figura 5.19 : Imagen de la miniatura usada en YouTube del vídeo: MATLAB: Operaciones de escalares con vectores y matrices

Se explican diferentes operaciones y su resultado:

- Suma **k+v**: Genera un nuevo vector o matriz del mismo tamaño que **v**. Cada elemento resultante es la suma del escalar **k** con cada elemento de **v**. Este resultado es idéntico al de **v+k**.
- Resta **k-v**: Produce un nuevo vector o matriz con cada elemento siendo el resultado de restar cada elemento de **v** al escalar **k**.
- Resta **v-k**: Similar a **k-v**, pero esta vez se resta el escalar **k** de cada elemento de **v**.
- Multiplicación **k*v**: Resulta en un nuevo vector o matriz donde cada elemento es la multiplicación del escalar **k** con cada elemento de **v**. Este resultado es igual al de **v*k**.
- División **v/k**: Crea un nuevo vector o matriz donde cada elemento es el resultado de dividir cada elemento de **v** entre el escalar **k**.

- División **k/v**: Esta operación dará un error, ya que no es coherente ni con las operaciones vectoriales ni matriciales.

Se proporcionan ejemplos de estas operaciones utilizando un vector **v** con los elementos 1, 2, 3, 4 y un escalar **k** asignado al valor 5. Además, se introducen operaciones elemento a elemento:

- División **k./v**: Produce un nuevo vector o matriz donde cada elemento es el resultado de dividir el escalar **k** por cada elemento de **v**.
- Potencia **v.^k**: Genera un nuevo vector o matriz donde cada elemento es el resultado de elevar cada elemento de **v** al escalar **k**. Sin el operador punto, esta operación solo es posible si **v** es una matriz cuadrada, generando una nueva matriz cuadrada resultante de multiplicar **v**, **k** veces por sí misma.
- Potencia **k.^v**: Crea un nuevo vector o matriz donde cada elemento es el resultado de elevar el escalar **k** a cada elemento de **v**. Sin el operador punto, solo es posible si **v** es una matriz cuadrada.

El vídeo muestra casos conflictivos, como elevar un vector fila **v** a **k** o **k** al vector fila **v** sin usar el operador punto, los cuales resultarán en un error.

Finalmente, se presenta un ejemplo con una matriz cuadrada **B** de tamaño 2x2, demostrando que todas las operaciones discutidas son posibles y resultan en otra matriz cuadrada del mismo tamaño.

## 5.1.20. MATLAB: Trasponer vectores y matrices con números reales y complejos

**Descripción de YouTube**

En este vídeo se explica una operación muy sencilla y muy útil cuando se trabaja con vectores y matrices: la operación de trasposición.

Recordemos que para MATLAB, un vector es un caso particular de matriz en la cual una de sus dos dimensiones es 1. Por eso, al trasponer un vector fila obtenemos un vector columna, y viceversa.

Los conceptos que se explican en este vídeo son:

- Concepto de trasposición y los dos operadores de MATLAB, que son:
  - ❖ Comilla simple ' -- Realiza la trasposición conjugada compleja.

- ❖ Comilla con punto .' -- Realiza la trasposición sin conjugación.

- Si en una matriz todos los elementos son reales (sin números complejos), ambos operadores son equivalentes.
- Si hay elementos complejos (aunque sea solo un número complejo), hay que tener en cuenta que la comilla simple cambia filas por columnas; y también calcula el conjugado de los números compuestos, es decir, cambia el signo de la parte imaginaria de cada número complejo (incluyendo los de la diagonal principal).

**Resumen**

El vídeo es una introducción a MATLAB, enfocada en la transposición de vectores y matrices.

La transposición es una operación que intercambia las filas por las columnas de un vector o una matriz. En MATLAB, esta operación se puede realizar con dos operadores: la comilla simple (carácter `'`) y la comilla precedida de un punto (caracteres `.'`), los cuales tienen diferentes propiedades.

El operador de comilla simple realiza la trasposición conjugada compleja. Es decir, transpone una matriz o vector y, si los elementos son complejos, también realiza su conjugación.

Por otro lado, el operador de comilla con punto realiza la transposición sin efectuar ninguna otra operación adicional, es decir, sin realizar la conjugación de los elementos complejos.

Se proporcionan ejemplos utilizando ambos operadores con vectores y matrices, tanto con elementos reales como complejos. En el caso de los vectores, se muestra cómo un vector fila se convierte en un vector columna al transponerlo y viceversa. Cuando se trata de matrices con elementos complejos, la transposición con el operador de comilla simple no solo intercambia las filas por las columnas, sino que también conjuga los elementos complejos.

Por otro lado, cuando se utiliza el operador de comilla con punto con matrices o vectores que contienen elementos complejos, simplemente intercambia las filas por las columnas sin realizar ninguna conjugación.

En resumen, para vectores y matrices de elementos exclusivamente reales, ambos operadores se comportan de la misma manera. Sin embargo, si alguno de los elementos es complejo, los resultados varían dependiendo del operador utilizado.

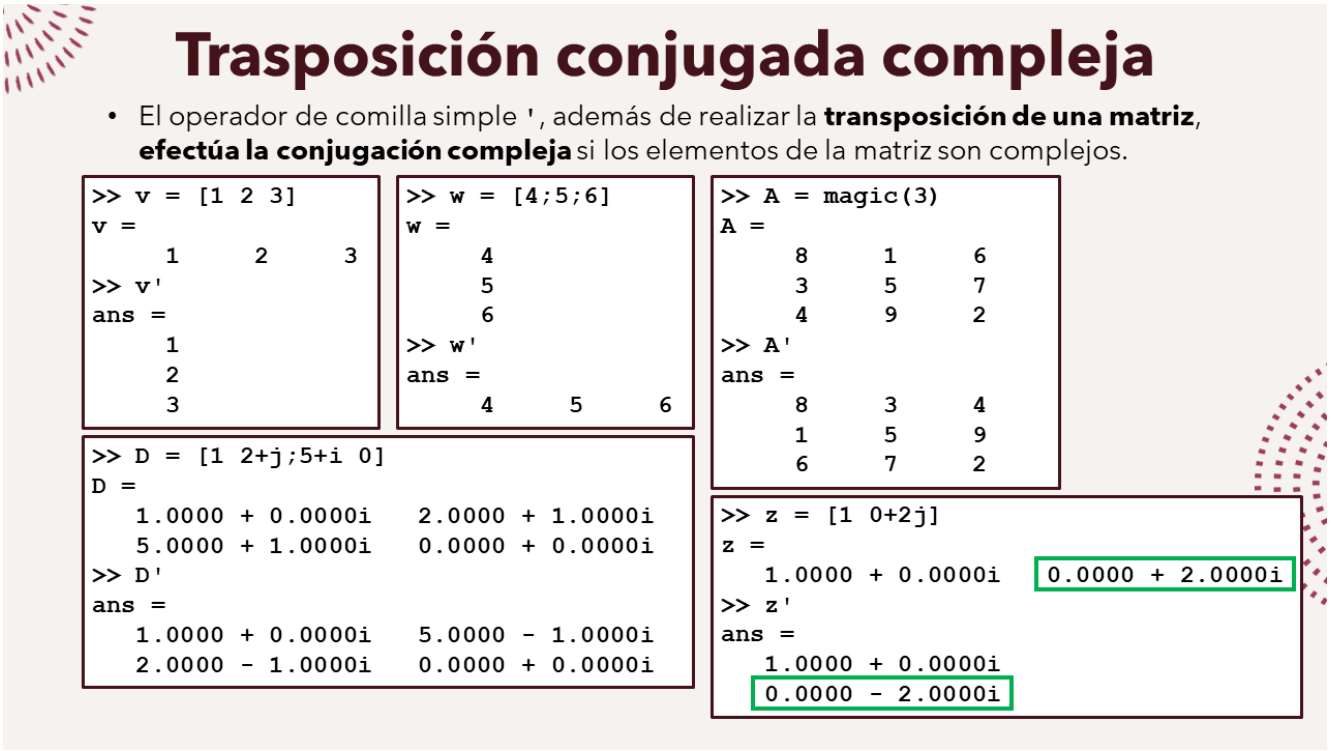


Figura 5.20 : Imagen de la miniatura usada en YouTube del vídeo: MATLAB: Trasponer vectores y matrices con números reales y complejos

## 5.1.21. Funciones de MATLAB para vectores y matrices

**Descripción de YouTube**

En este vídeo se explican varias funciones para realizar operaciones sobre los elementos de un vector o matriz en MATLAB. Resumiendo, el contenido de este vídeo incluye:

- Algunas funciones básicas: sum (suma), prod (producto), mean (media), median (mediana), max (máximo), y min (mínimo).
- Funciones de acumulación cumsum y cumprod, que generan un vector con sumas y productos acumulados, respectivamente.
- Se explican funciones estadísticas: var (varianza) y std (desviación estándar), y cómo ajustar la normalización de éstas.
- Función sort para ordenar un vector en forma ascendente o descendente, y cómo ordenar una matriz por filas o columnas.
- Función norm, que calcula la norma de un vector o la norma 2 de una matriz.
- Se aclara que algunas funciones, cuando se usan con matrices, actúan por columnas (por defecto), y cómo se puede operar por filas trasponiendo las matrices.

**Resumen**

El vídeo se enfoca en las funciones que realizan operaciones sobre los elementos de un único vector o matriz. Se detallan y explican varias funciones:

- **`sum`**: Suma todos los elementos de un vector.
- **`prod`**: Multiplica todos los elementos de un vector.
- **`mean`**: Calcula la media de los elementos de un vector. También se muestra una alternativa para calcular la media utilizando **`sum`** y **`length`**.
- **`median`**: Devuelve la mediana de los elementos de un vector.
- **`max`** y **`min`**: Devuelven el elemento más grande y más pequeño de un vector, respectivamente. También se demuestra cómo obtener la posición del máximo y mínimo elemento.
- **`cumsum`** y **`cumprod`**: Estas funciones devuelven un vector del mismo tamaño con las sumas y los productos acumulados de los elementos del vector original, respectivamente.
- **`var`** y **`std`**: Calculan la varianza y la desviación estándar de los elementos de un vector, respectivamente.
- **`sort`**: Ordena los elementos de un vector de forma ascendente o descendente.
- **`norm`**: Calcula la norma del vector.

Estas funciones se pueden aplicar también a matrices, operando por columnas. Para aplicar estas operaciones por filas, se necesita transponer la matriz. La excepción es la función **`norm`** que calcula la **norma 2** de la matriz, y por ende nos devuelve un escalar.

La función **`sort`** para matrices necesita un segundo argumento para ordenar las filas en lugar de las columnas.

El vídeo también aborda cómo las funciones **`var`** y **`std`** normalizan por defecto por **`n-1`** elementos. Para normalizar por n elementos, se usa **`var(v,1)`** y **`std(v,1)`**.

Finalmente, el vídeo concluye con un ejercicio para calcular la varianza de un vector sin usar la función **`var`** y comparar los resultados obtenidos con el uso de dicha función.

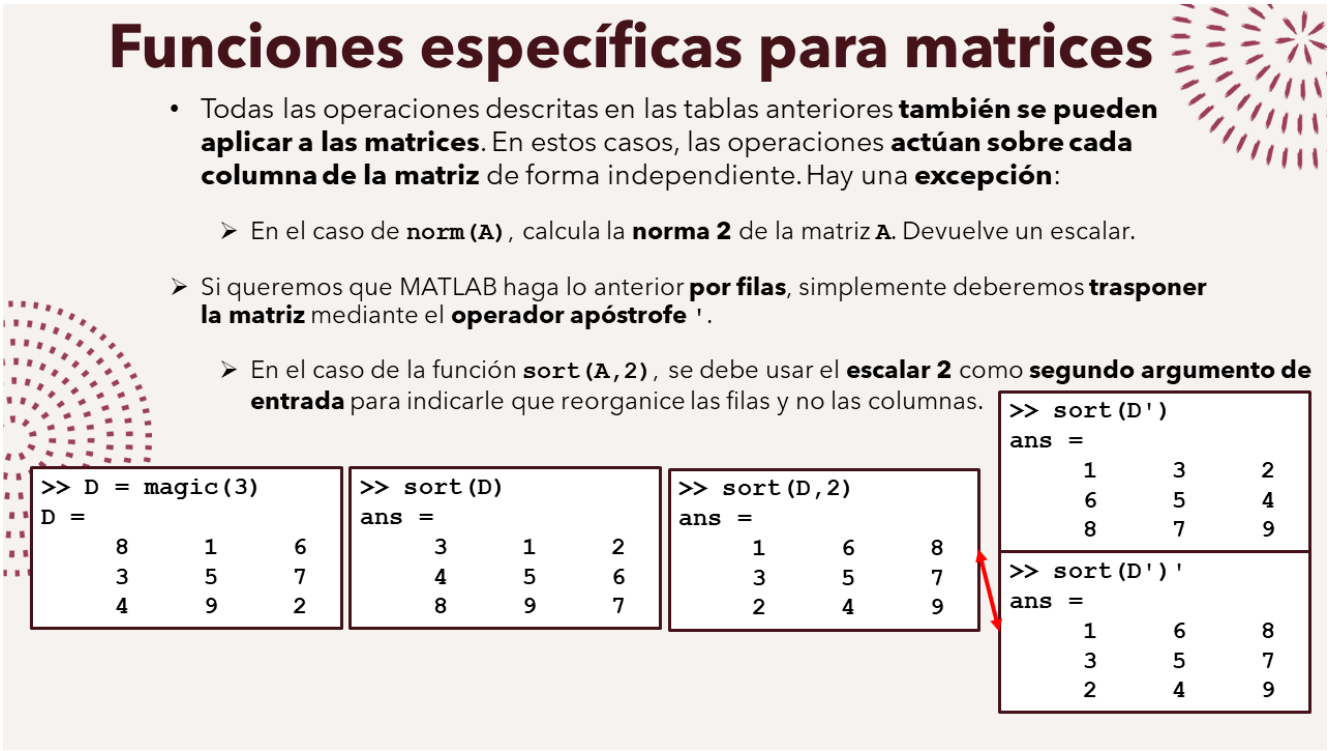


Figura 5.21 : Imagen de la miniatura usada en YouTube del vídeo: Funciones de MATLAB para vectores y matrices

## 5.1.22. MATLAB: Comandos sencillos, pero útiles, como `tic` y `toc`

### Descripción de YouTube

El vídeo explica algunos comandos simples de MATLAB, pero que pueden resultar útiles en muchas situaciones. Los comandos que tratamos son son:

- version: Muestra la versión de MATLAB.
- logo y membrane: Abren una ventana gráfica con el logo de MATLAB.
- computer: Devuelve el tipo de máquina y sistema operativo.
- memory: Proporciona información sobre el uso y los límites de la memoria.
- clock: Devuelve la fecha y la hora actuales en un vector de seis elementos.
- date: Devuelve la fecha actual como una cadena de texto.
- tic y toc: Comandos para medir el tiempo que MATLAB tarda en ejecutar un código.

El vídeo también muestra cómo utilizar los comandos tic y toc para comparar la eficiencia de distintos métodos para resolver un mismo problema. Se presenta un ejercicio en el que se genera una matriz pseudoaleatoria y un vector, y se resuelve un sistema lineal de 5000 ecuaciones con 5000 incógnitas utilizando dos enfoques diferentes. Los resultados

muestran que la elección del método puede tener un impacto significativo en el tiempo de ejecución, especialmente para problemas grandes y operaciones repetitivas. Esto se debe a que calcular la inversa de una matriz es una operación poco eficiente.

El vídeo enfatiza la importancia de tener suficiente memoria al trabajar con matrices grandes, y concluye destacando que los comandos tic y toc son útiles para el análisis de rendimiento y la optimización del código.

**Resumen**

El vídeo es un tutorial de MATLAB que se centra en explicar algunos comandos sencillos pero útiles.

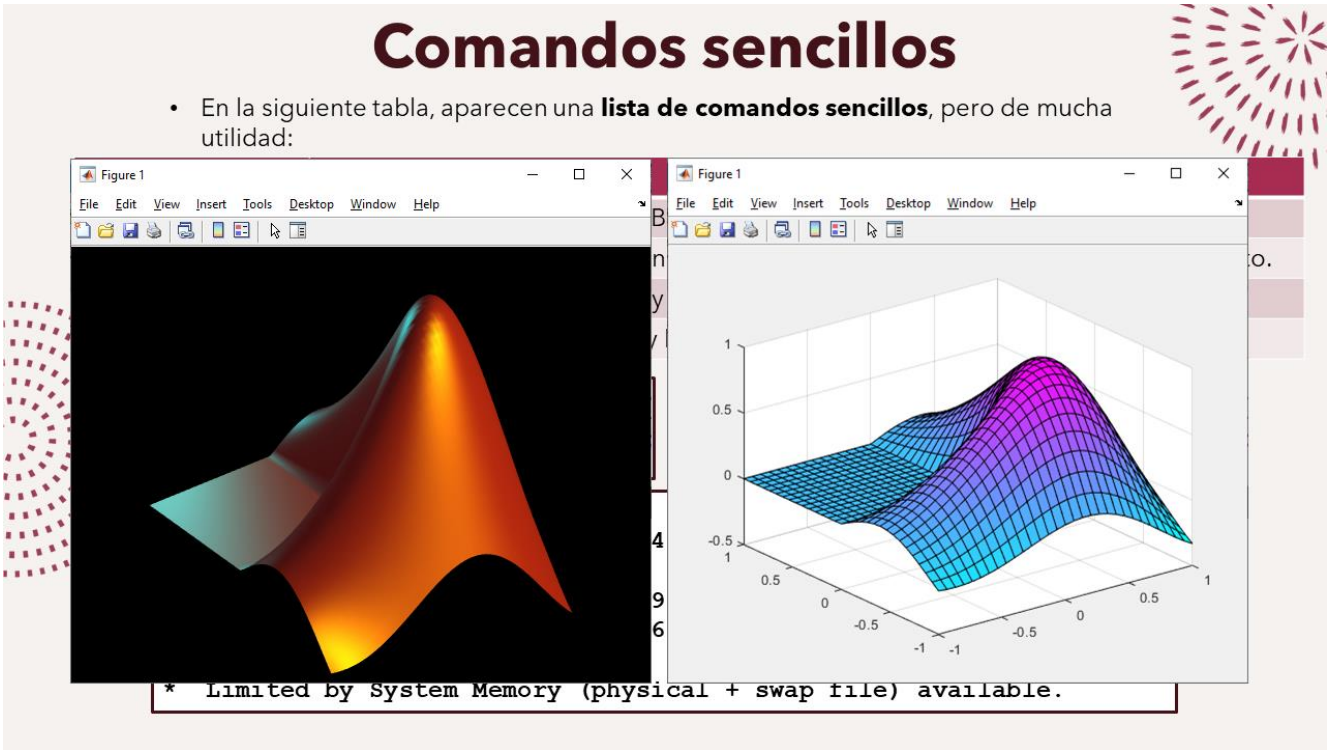


Figura 5.22 : Imagen de la miniatura usada en YouTube del vídeo: MATLAB: Comandos sencillos, pero útiles, como `tic` y `toc`

Los comandos cubiertos en este vídeo son:

- **`version`**: Muestra la versión de MATLAB.
- **`logo`** y **`membrane`**: Abren una ventana gráfica con el logo de MATLAB.
- **`computer`**: Devuelve el tipo de máquina y sistema operativo.
- **`memory`**: Proporciona información sobre el uso y los límites de la memoria.
- **`clock`**: Devuelve la fecha y la hora actuales en un vector de seis elementos.

- **`date`**: Devuelve la fecha actual como una cadena de texto.
- **`tic`** y **`toc`**: Se utilizan para medir el tiempo que MATLAB tarda en ejecutar el código.

El vídeo también muestra cómo utilizar los comandos **`tic`** y **`toc`** para comparar la eficiencia de distintos métodos para resolver un mismo problema. Se presenta un ejercicio en el que se genera una matriz pseudoaleatoria y un vector, y se resuelve un sistema lineal de 5000 ecuaciones con 5000 incógnitas utilizando dos enfoques diferentes. Los resultados muestran que la elección del método puede tener un impacto significativo en el tiempo de ejecución, especialmente para problemas grandes y operaciones repetitivas. Esto se debe a que calcular la inversa de una matriz es una operación poco eficiente.

El vídeo enfatiza la importancia de tener suficiente memoria al trabajar con matrices grandes, y concluye destacando que los comandos **`tic`** y **`toc`** son útiles para el análisis de rendimiento y la optimización del código.

### 5.1.23. MATLAB: Generación de números pseudoaleatorios con las funciones `rand`, `randn` y `randi`

**Descripción de YouTube**

Este vídeo profundiza en cómo podemos crear números pseudoaleatorios con MATLAB utilizando las funciones esenciales: rand, randn y randi. Los puntos clave abordados son:

- Función rand para generar números pseudoaleatorios con una distribución uniforme dentro del rango [0,1]. Esta puede ser modificada para generar matrices o vectores de diversos tamaños y rangos.
- Función randn para generar números pseudoaleatorios siguiendo una distribución normal, teniendo una media de 0 y una desviación estándar de 1. Esos dos valores pueden modificarse.
- Función randi para generar números enteros pseudoaleatorios dentro de un rango definido o bien una matriz multidimensional de estos números.
- Todas estas funciones se basan en una semilla inicial para la generación de números, que es un número que puede ser definido por el usuario.
- Se subraya que los números resultantes no son realmente aleatorios y, por eso se llaman pseudoaleatorios, ya que su generación está sujeta al algoritmo interno de MATLAB para cada función y a la semilla inicial seleccionada.

- 

**Resumen**

El vídeo se centra en cómo MATLAB genera matrices, vectores y escalares de números pseudoaleatorios. Aunque MATLAB ofrece varias funciones para este propósito, dependiendo del tipo de distribución deseada, el vídeo se enfoca en las funciones que generan distribuciones uniformes y normales: **`rand`**, **`randi`** y **`randn`**.

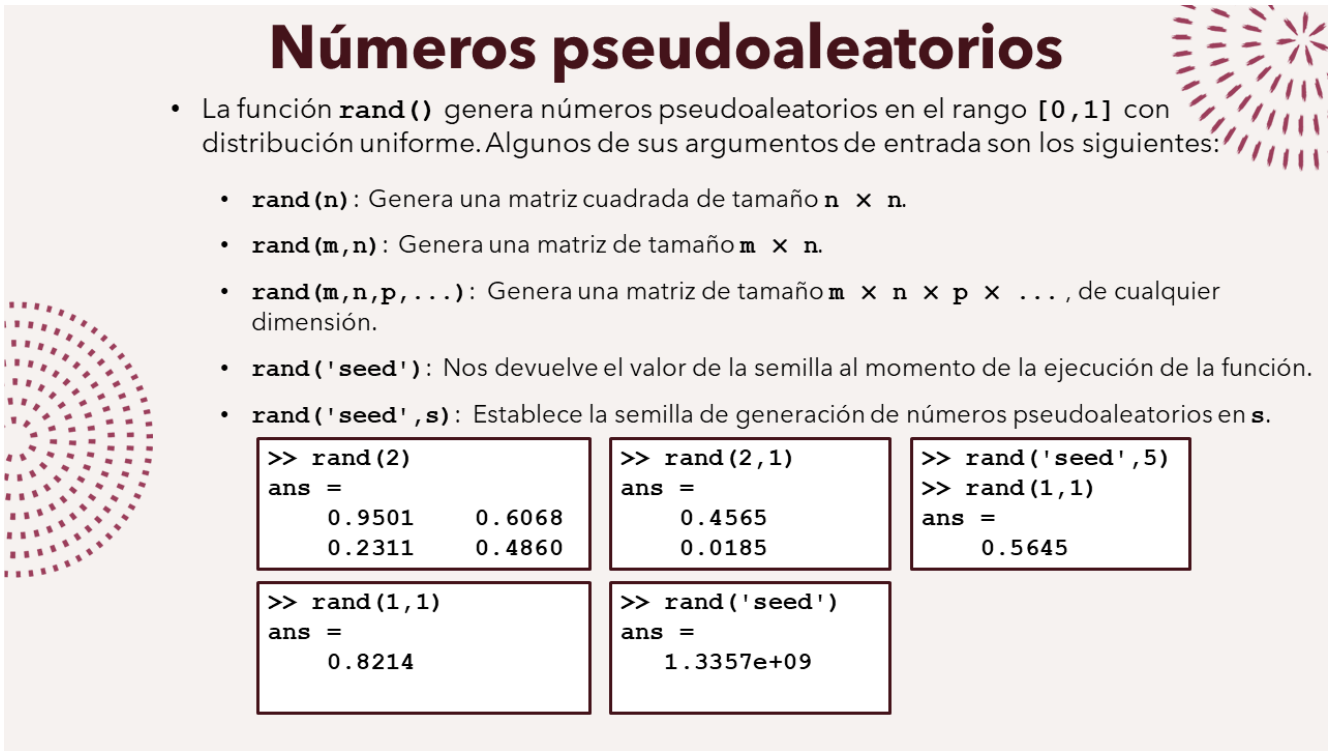


Figura 5.23 : Imagen de la miniatura usada en YouTube del vídeo: Generación de números pseudoaleatorios con MATLAB: Funciones `rand`, `randn` y `randi`

Estas funciones generan números pseudoaleatorios a través de algoritmos que usan una semilla inicial, que puede venir de diversas fuentes, como la temperatura del procesador o el valor del reloj interno del sistema en el momento de la ejecución de la función.

La función **`rand`** genera números pseudoaleatorios en el rango [0,1] con distribución uniforme. Sus argumentos de entrada pueden variar, permitiendo generar una matriz cuadrada, una matriz de tamaño m x n, una matriz de cualquier dimensión, devolver el valor de la semilla o establecer una semilla de generación de números pseudoaleatorios. Para cambiar el rango predefinido, se puede multiplicar la función **`rand`** por la diferencia entre el último y el primer valor del nuevo rango, y sumar al resultado el valor del inicio del nuevo rango.

La función **`randn`** genera números pseudoaleatorios con distribución normal, con media 0 y desviación estándar 1. No tiene un rango de valores definido por defecto, ya que su rango depende de la media y la desviación estándar. Comparte argumentos con la función **`rand`**.

La función **`randi`**, por otro lado, genera números enteros pseudoaleatorios. Sus argumentos de entrada permiten generar un número entero pseudoaleatorio en un rango específico, o una matriz de tamaño m x n x p x ... de números enteros pseudoaleatorios en un rango específico. También permite devolver o establecer la semilla de generación de números pseudoaleatorios.

El vídeo enfatiza que los números generados por estas funciones son pseudoaleatorios, y su generación depende del algoritmo utilizado y de la semilla inicial elegida.

## 5.1.24. MATLAB: Gráficas 2D con la función `plot`

**Descripción de YouTube**

En este vídeo aprenderemos a usar la función plot para hacer gráficas 2D. Particularmente, se explica lo siguiente:

- Comando plot y comandos afines para configurar la gráfica: poner un título, poner nombre a los ejes, poner una rejilla en el fondo y poner una leyenda.
- Comando hold (on/off) para permitir dibujar varias gráficas en la misma figura, superponiendo unas a otras, en el orden que deseemos.
- Comando plot para dibujar varias gráficas usando el comando una única vez.
- Comando figure para abrir ventanas gráficas. Esto nos permite trabajar con varios gráficos en paralelo.
- Comando subplot(f,c,n) para dibujar varias gráficas en la misma figura. Es decir, solo tendríamos una figura pero estaría compuesta de varias gráficas.
- Con subplot definimos una "matriz de gráficos" de f filas y c columnas. Es decir, que el número de subgráficos que contiene será la multiplicación f*c.
- El tercer argumento de subplot indica a qué gráfica nos referimos y ese valor debe estar comprendido entre 1 y f*c.

- Ejemplos: subplot(1,3,1) indica que pintaremos 1 fila de tres gráficos y a partir de ese comando todo lo que pintemos se pintará en la primera gráfica (la que está más a la izquierda). Por otra parte, subplot (3,1,3), indica que pintaremos 3 gráficos en 3 filas distintas (usando una sola columna). Es decir, los 3 gráficos estarán uno encima de otro y a partir de ese comando todo lo que pintemos se pintará en la última gráfica (la tercera, que será la que está abajo del todo).

**Resumen**

Este vídeo es una introducción a la creación de gráficos 2D en MATLAB utilizando la función **plot**. Se menciona que los gráficos son una herramienta esencial en muchas áreas científicas para visualizar y analizar datos. MATLAB ofrece una variedad de funciones para crear diferentes tipos de gráficos y permite personalizarlos para adaptarse a necesidades específicas.

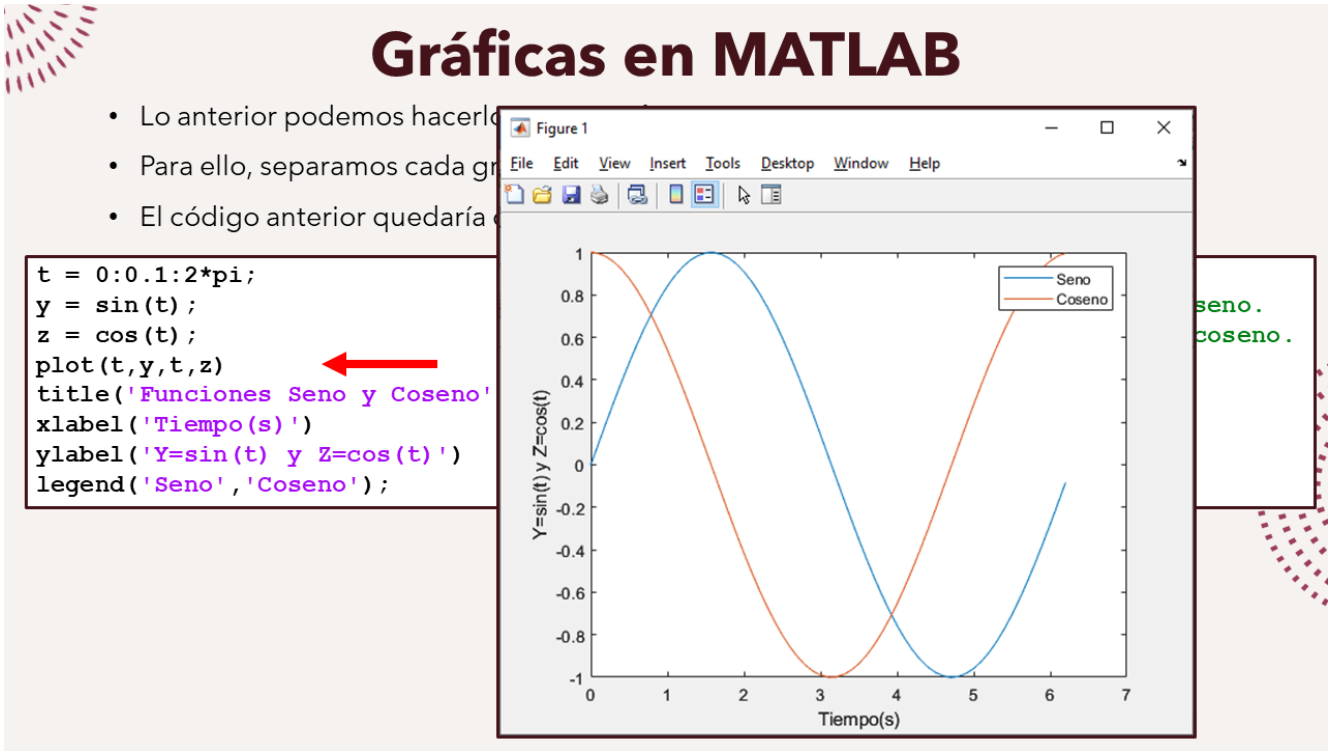


Figura 5.24 : Imagen de la miniatura usada en YouTube del vídeo: MATLAB: Gráficas 2D con la función `plot`

El proceso para crear gráficos 2D en MATLAB es sencillo. Primero, se definen los vectores de valores para los ejes **X** e **Y**. Luego, se utiliza la función **`plot`** para graficar los puntos y unirlos. Se pueden agregar títulos a los gráficos utilizando la función **`title`**, y etiquetas a los ejes con **`xlabel`** e **`ylabel`**. Para agregar una cuadrícula al gráfico, se utiliza el comando **`grid on`**. Se puede agregar una leyenda utilizando la función **`legend`**.

El vídeo proporciona un ejemplo específico de cómo trazar la función seno en función del tiempo. Se crea el vector **t** para los valores del eje **X** y el vector **y** para los valores del eje **Y**, y luego se utiliza **plot** para trazar y unir los puntos. Se agrega un título al gráfico con **title**, etiquetas a los ejes con **xlabel** y **ylabel**, y se activa la cuadrícula con **grid on**.

Se explica que MATLAB borra una gráfica al crear una nueva, pero este comportamiento se puede cambiar con el comando **hold on**, que permite superponer gráficos. Para dejar de superponer gráficos, se puede utilizar el comando **hold off**.

Se da un ejemplo adicional donde se traza la función coseno en la misma gráfica que el seno. Después de activar el modo de retención con **hold on**, se crea un nuevo vector **z** para los valores del eje Y de la función coseno. Se traza esta función con **plot** y se agrega una leyenda con **legend**.

Se menciona que todas las gráficas se pueden trazar en una única llamada a **plot**, separando cada gráfica con comas. También se pueden dibujar en diferentes ventanas de gráficos utilizando el comando **figure(n)**, donde **n** es el número de la figura.

Finalmente, se puede dividir una figura en varias subtramas utilizando el comando **subplot(m,n,p)**, donde **m** es el número de filas, **n** es el número de columnas y **p** es la posición de la gráfica.

## 5.1.25. Ejercicio MATLAB: Modelizar y graficar la población de los Estados Unidos

### Descripción de YouTube

En este ejercicio mostramos cómo usar MATLAB para hacer un gráfico de la población de los Estados Unidos entre 1790 y 2020:

- En un script en MATLAB crearemos dos vectores: years (representando los años) y USA_population (representando los valores de población).
- Se utiliza la función plot con dos argumentos para indicar los puntos de la gráfica: (years, USA_population), que representan los valores de los puntos en los ejes X e Y respectivamente. Obviamente, ambos vectores deben tener el mismo número de elementos, ya que ambos representan puntos en 2D.
- Usando otros dos argumentos de plot ajustamos el grosor de la línea.
- Con 'axis tight' conseguimos que la gráfica quede del tamaño justo, sin márgenes por ningún lado.

- Con 'grid on' visualizamos una cuadrícula de color claro en el fondo de la gráfica. Esto facilita ver los valores aproximados que toma la gráfica en cada punto.
- El resultado es un gráfico en forma de S que predice que la población de los Estados Unidos tuvo un crecimiento abrupto pero tiende a crecer más lentamente a partir de 2020.

NOTA: Evidentemente, la gráfica es aproximada y no se puede saber el desarrollo futuro de la población, lo cual depende de multitud de factores.

**Resumen**

Este vídeo muestra cómo resolver un problema relacionado con la modelización de la población de los Estados Unidos desde 1790 hasta 2020. El problema implica crear un script en MATLAB para mostrar un gráfico de la población de los Estados Unidos cada 10 años durante este período, utilizando una fórmula dada y ajustando los ejes al rango de datos con el comando **`axis tight`**.

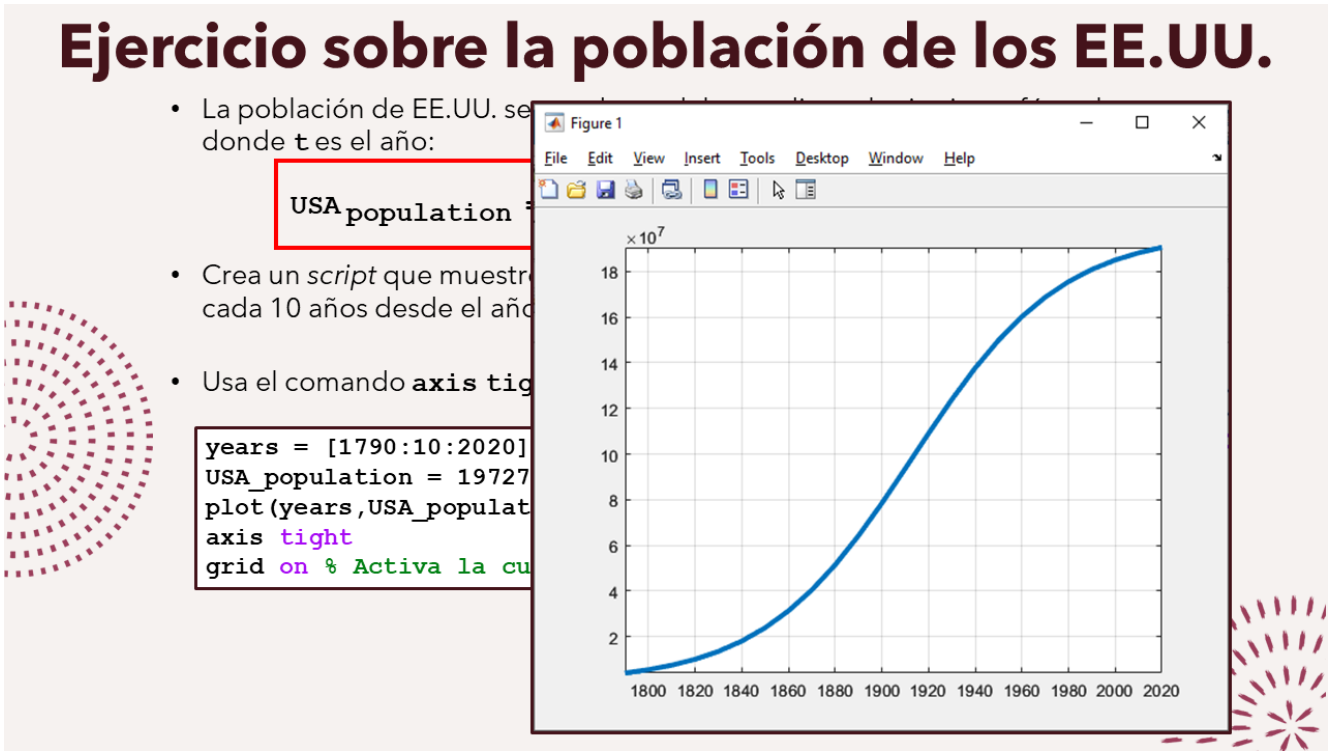


Figura 5.25 : Imagen de la miniatura usada en YouTube del vídeo: MATLAB: Ejercicio MATLAB: Modelizar y graficar la población de los Estados Unidos

El primer paso para resolver este problema es crear un vector **`years`** que vaya del valor 1790 al valor 2020 en incrementos de 10 usando el operador dos puntos. Este vector representa los años. El siguiente paso es crear un segundo vector, **`USA_population`**, que contiene los valores de la fórmula de población para cada uno de los elementos del vector

**`years`**. Esta operación involucra varias operaciones elemento a elemento, ya que es una operación entre escalares y un vector.

Luego, el vídeo muestra cómo usar la función **`plot`** para crear el gráfico. La función **`plot`** toma como primer argumento el vector **`years`**, que representa la coordenada **X**, y como segundo argumento el vector **`USA population`**, que representa la coordenada **Y**. También se añade un tercer argumento para modificar el grosor de la línea del gráfico, que se establece en 3. Por último, se utiliza el comando **`axis`** con el atributo **`tight`** para ajustar los ejes de las coordenadas a los valores máximos y mínimos de la curva.

El vídeo concluye mostrando el resultado final del código, que es el gráfico de la población de los Estados Unidos desde 1790 hasta 2020.

## 5.1.26. MATLAB: Operaciones entre vectores y matrices

### Descripción de YouTube

El vídeo cubre algunas de las operaciones básicas que se pueden realizar entre vectores y matrices en MATLAB, usando operadores y algunas funciones. Se enfatiza que estas operaciones deben seguir las reglas matemáticas para no incurrir en errores. Por ejemplo, para sumar dos matrices ambas deben ser del mismo tamaño.

Distinguimos dos tipos de operaciones:

1. Operaciones ELEMENTO A ELEMENTO: son aquellas en las que cada elemento del primer operando se opera con su homólogo del segundo operando. Por tanto, en este tipo de operaciones los dos operandos deben tener exactamente la misma dimensión. Ejemplo: suma, resta... de vectores o matrices.
2. Operaciones que NO son elemento a elemento: Son operaciones que siguen otro algoritmo matemático y, por tanto, los operandos no tienen que tener el mismo tamaño. Ejemplo: la multiplicación de matrices (que es distinta a hacer la multiplicación elemento a elemento).

Los contenidos explicados en el vídeo incluyen:

- Operaciones elemento a elemento entre vectores/matrices: suma, resta, multiplicación, división por la derecha y por la izquierda y potencia.
- Valores especiales que pueden surgir al hacer operaciones: Inf (infinito) y NaN (no es un número, Not a Number).

- Función dot para calcular el producto escalar de dos vectores y dos formas más de efectuar esa operación.
- Función cross para calcular el producto vectorial de dos vectores tridimensionales.
- Se describen las operaciones entre matrices que no son elemento a elemento: producto matricial, inversión de matriz, la elevación de una matriz a una potencia, y las operaciones de división por la izquierda y por la derecha.
- Función inv para calcular la inversa de una matriz cuadrada.
- Se explican los operadores / y \ que permiten realizar operaciones de división sin calcular estrictamente la matriz inversa. En el fondo, el operador \ se aplica para resolver sistemas de ecuaciones, sin que el número de incógnitas deba coincidir con el número de ecuaciones.

**Resumen**

El vídeo cubre las operaciones que se pueden realizar entre vectores y matrices en MATLAB, incluyendo los operadores y funciones para llevar a cabo estas operaciones. Se enfatiza que estas operaciones deben seguir las leyes matemáticas que rigen las operaciones matriciales, como que las matrices deben ser del mismo tamaño para sumarse.

Las operaciones elemento a elemento se describen en detalle, incluyendo suma, resta, multiplicación, división por la derecha e izquierda y elevación. Se muestra que MATLAB devolverá errores si las dimensiones de los vectores o matrices no coinciden para estas operaciones, y se explican los significados de **`Inf`** (infinito) y **`NaN`** (no es un número) como posibles resultados.

Se detallan ejemplos de operaciones elemento a elemento en matrices del mismo tamaño y se demuestra que MATLAB devolverá un error si las dimensiones de las matrices no coinciden.

Se explica la función **`dot`** para calcular el producto escalar de dos vectores, y la función **`cross`** para calcular el producto vectorial de dos vectores. Estas funciones requieren que los vectores tengan el mismo tamaño y sean tridimensionales, respectivamente, o MATLAB devolverá un error.

Finalmente, se describen las operaciones entre matrices que no son elemento a elemento, como el producto matricial, la operación de inversión de matriz, la elevación de una matriz a una potencia, y las operaciones de división por la izquierda y la derecha. Estas

operaciones requieren que las matrices cumplan ciertas condiciones, como ser cuadradas o tener el mismo número de columnas o filas, o MATLAB devolverá un error.

Se discuten también la función **`inv`** para calcular la inversa de una matriz cuadrada y los operadores **`/`** y **`\`** que permiten realizar operaciones de división sin calcular estrictamente la matriz inversa.

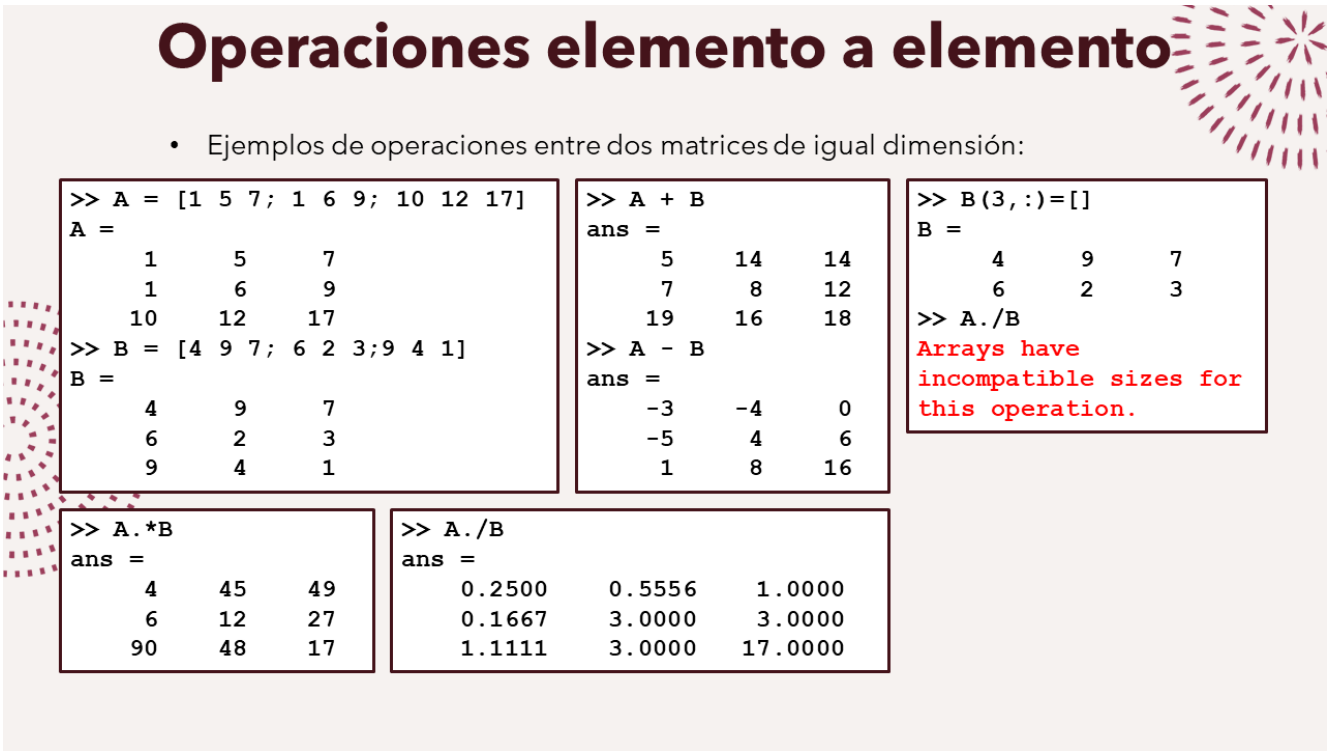


Figura 5.26 : Imagen de la miniatura usada en YouTube del vídeo: MATLAB: Operaciones entre vectores y matrices

## 5.1.27. MATLAB: Atributos de la función `plot` para personalizar las gráficas

### Descripción de YouTube

En este vídeo aprenderemos a personalizar gráficas 2D en MATLAB usando atributos en la función plot y otras funciones relacionadas.

- Los atributos se definen tras los vectores que representan los valores de los puntos en los ejes X e Y.
- En cada gráfica podemos modificar el tipo de línea que une los puntos, el color y el tipo de marca que pone en cada punto (si queremos remarcar cada punto).
- Los atributos que afectan a todas las curvas se escriben al final en la función plot.
- Función axis para modificar los límites de los ejes X e Y.

- Comando text para agregar texto a las gráficas.
- Veremos cómo usar letras griegas en las funciones que permitan poner texto.
- Configurar el grosor de las líneas con LineWidth.
- Configurar el tamaño de las fuentes con FontSize.
- Uso de la función set(gca, ...) para modificar características de los ejes de una gráfica.

**Resumen**

El vídeo trata sobre la personalización de gráficas 2D en MATLAB. Primero, menciona que los gráficos se pueden configurar usando atributos de la función **`plot`**. Los atributos se definen en forma de cadenas de caracteres tras los dos vectores que representan los valores de los ejes **X** e **Y** de cada curva.

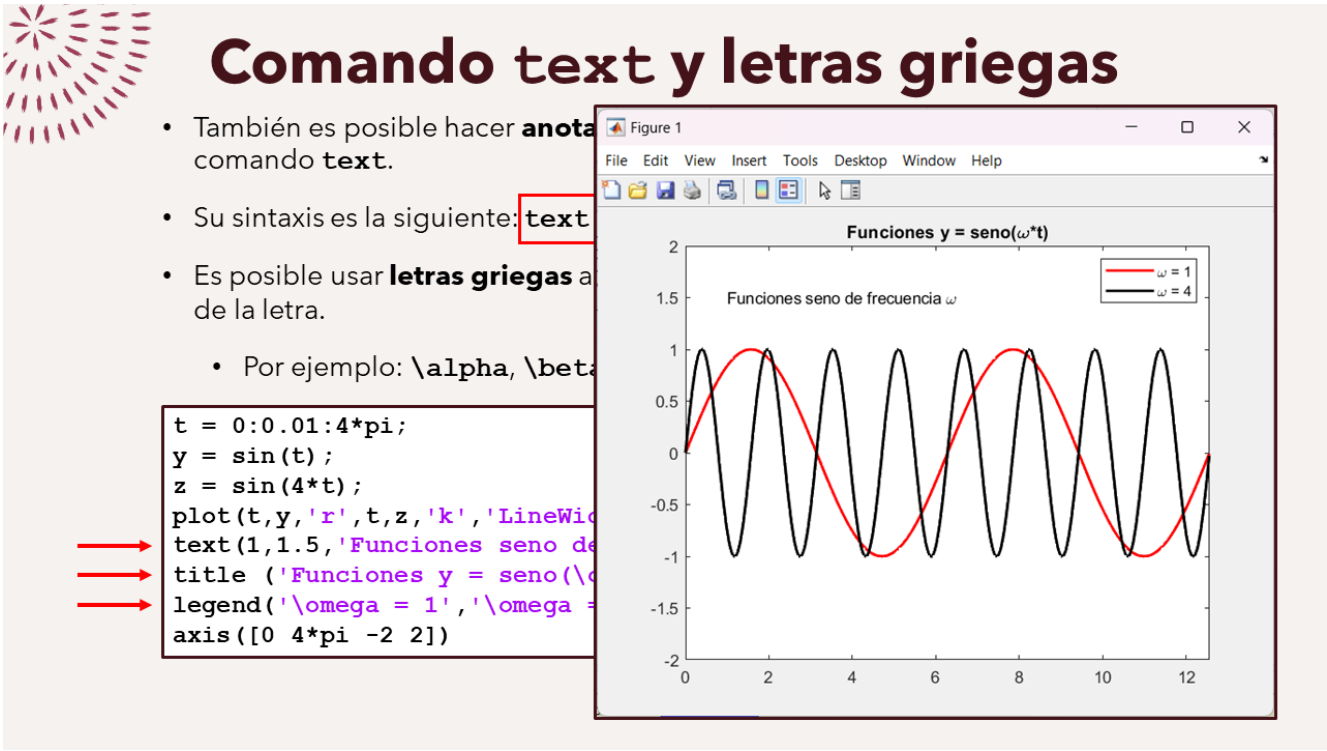


Figura 5.27 : Imagen de la miniatura usada en YouTube del vídeo: MATLAB: Atributos de la función plot para personalizar las gráficas

Existen atributos que se usan una única vez y afectan a todas las curvas. Estos se escriben como últimos argumentos de entrada del **`plot`**. Se pueden combinar ambos tipos de atributos en un único **`plot`**. Los atributos se usan después de los dos vectores que representan los puntos de cada curva.

En una demostración más detallada, el vídeo presenta cómo se crean vectores que representan valores de tres funciones distintas: un seno de amplitud 3, fase 0 y frecuencia igual a 1, una recta y una parábola. Luego usa la función **`plot`** para graficar estas funciones con distintos atributos.

MATLAB selecciona automáticamente los límites de las gráficas en los ejes **X** e **Y**, pero se pueden modificar con la función **`axis`**. El vídeo muestra cómo usar esta función para limitar el rango de los ejes.

Además, se pueden hacer anotaciones en las gráficas de MATLAB usando el comando **`text`**, que coloca texto en una posición específica. También es posible usar letras griegas en las anotaciones.

Otras funciones para personalizar las gráficas incluyen **`LineWidth`** para cambiar el tamaño de las líneas y **`FontSize`** para el tamaño de las fuentes. Estos atributos se pueden usar en funciones que generan texto o líneas.

La función **`set(gca,atributo1,atributo2,…)`** permite modificar los ejes de una gráfica. Con esta función, se puede cambiar el grosor de los ejes y el tamaño de la fuente numeración.

Finalmente, el vídeo sugiere revisar la documentación online de MATLAB para aprender más sobre la personalización de gráficos 2D, ya que es un tema muy extenso para cubrir por completo en un curso de introducción.

# 5.2. Listas de reproducción

Como se ha comentado al principio de este capítulo, se han creado varias listas de reproducción. Estas listas comprenden la organización de los vídeos publicados, garantizando a los seguidores una secuencia temática que asegure una correcta asimilación de los conceptos. Se han creado 5 listas de reproducción, una de ellas con todos los vídeos ordenados según el temario de la asignatura Sistemas Informáticos, las demás contienen los vídeos organizados según la temática.

Las listas creadas se pueden ver en la primera columna de las Tabla 5.1, 5.2, 5.3, 5.4 y 5.5, mientras que la segunda columna de estas tablas muestra los vídeos que conforman la lista organizados por orden a como se deberían visualizar por parte de un alumno que quisiera seguir el curso completo, es decir aquellos que ocupan filas superiores son los

primeros vídeos que habría que visualizar y aprender. La tercera columna muestra la duración de cada vídeo.

Al final se muestra la duración total de cada lista. La primera lista comprende todos los vídeos realizados, por lo que se ve que el material audiovisual desarrollado tiene una duración total de 2 horas, 55 minutos y 47 segundos.

| Nombre de la lista | Vídeos ordenados por su orden didáctico | Duración |
|---|---|---|
| **Curso de MATLAB desde cero** | Qué es MATLAB | 6:52 |
| | Entorno de MATLAB y conceptos básicos | 12:13 |
| | MATLAB: Conceptos básicos sobre variables y ficheros .mat | 10:25 |
| | Scripts o ficheros .m de MATLAB: creación y ejecución | 6:30 |
| | MATLAB aplicado a la astrofísica: ¿En cuánto tiempos se consumirá toda la masa del SOL? | 7:25 |
| | Formatos de salida de MATLAB y notación científica | 5:02 |
| | Nombres permitidos y comando `help` en MATLAB | 5:28 |
| | Comandos sencillos, pero útiles, como `tic` y `toc` | 3:25 |
| | Números complejos en MATLAB | 7:49 |
| | Funciones matemáticas elementales en MATLAB | 8:06 |
| | MATLAB: Crear vectores y matrices | 5:58 |
| | MATLAB: Composición de vectores y matrices | 6:59 |
| | MATLAB: Direccionar y modificar vectores y matrices | 4:43 |
| | MATLAB: Operador dos puntos (`:`) | 5:11 |
| | MATLAB: Transponer vectores y matrices con números reales y complejos | 3:05 |
| | MATLAB: Operaciones de escalares con vectores y matrices | 4:59 |
| | MATLAB: Generación de números pseudoaleatorios con las funciones `rand`, `randn` y `randi` | 6:12 |
| | Funciones de MATLAB para vectores y matrices | 7:00 |
| | MATLAB: operaciones entre vectores y matrices | 7:06 |
| | Polinomios en MATLAB | 9:56 |
| | MATLAB: Ajuste polinomial a partir de un conjunto de puntos | 5:55 |

| | | |
|---|---|---|
| | MATLAB: Cálculo simbólico para solucionar ecuaciones | 11:39 |
| | Resolviendo ecuaciones lineales con MATLAB | 12:22 |
| | MATLAB: Problema de cálculo simbólico sobre el crecimiento bacteriano | 3:53 |
| | MATLAB: Gráficas 2D con la función `plot` | 6:41 |
| | MATLAB: Cómo personalizar gráficas con las funciones `plot`, `text`, `axis` y otras | 7:52 |
| | Ejercicio MATLAB: Modelizar y graficar la población de los Estados Unidos | 1:55 |
| **Duración total** | | 3:04:41 |
| **Duración promedio** | | 6:50 |

Tabla 5.1 : Lista de reproducción Curso de MATLAB desde cero

| **Nombre de la lista** | **Vídeos ordenados por su orden didáctico** | **Duración** |
|---|---|---|
| **Introducción a MATLAB** | Qué es MATLAB | 6:52 |
| | Entorno de MATLAB y conceptos básicos | 12:13 |
| | MATLAB: Conceptos básicos sobre variables y ficheros .mat | 10:25 |
| | Scripts o ficheros .m de MATLAB: creación y ejecución | 6:30 |
| | MATLAB: Comandos sencillos, pero útiles, como `tic` y `toc` | 3:25 |
| | Formatos de salida de MATLAB y notación científica | 5:02 |
| | Nombre permitidos y comando `help` en MATLAB | 5:28 |
| | MATLAB aplicado a la astrofísica: ¿En cuánto tiempo se consumirá toda la masa del Sol? | 7:25 |
| **Duración total** | | 57:20 |
| **Duración promedio** | | 7:10 |

Tabla 5.2 : Lista de reproducción Conceptos básicos sobre MATLAB

| | | |
|---|---|---|
| **MATLAB: Vectores y matrices** | MATLAB: Crear vectores y matrices | 5:58 |
| | MATLAB: Composición de vectores y matrices | 6:59 |
| | MATLAB: Direccionar y modificar vectores y matrices | 4:43 |
| | MATLAB: Operador dos puntos (`:`) | 5:11 |
| | MATLAB: Transponer vectores y matrices con números reales y complejos | 3:05 |
| | MATLAB: Operaciones de escalares con vectores y matrices | 4:59 |
| | Funciones de MATLAB para vectores y matrices | 7:00 |
| | MATLAB: operaciones entre vectores y matrices | 7:06 |
| | MATLAB: Generación de números pseudoaleatorios con las funciones `rand`, `randn` y `randi` | 6:12 |
| **Duración total** | | 51:13 |
| **Duración promedio** | | 5:41 |

Tabla 5.3 : Lista de reproducción Vectores y matrices en MATLAB

| | | |
|---|---|---|
| **MATLAB: Operaciones simples** | Números complejos en MATLAB | 7:49 |
| | Funciones matemáticas elementales en MATLAB | 8:06 |
| | Polinomios en MATLAB | 9:56 |
| | MATLAB: Ajuste polinomial a partir de un conjunto de puntos | 5:55 |
| | MATLAB: Cálculo simbólico para solucionar ecuaciones | 11:39 |
| | Resolviendo ecuaciones lineales con MATLAB | 12:22 |
| | MATLAB: Problema de cálculo simbólico sobre el crecimiento bacteriano | 3:53 |
| **Duración total** | | 59:40 |
| **Duración promedio** | | 8:31 |

Tabla 5.4 : Lista de reproducción Números complejos, funciones matemáticas, polinomios, ecuaciones, sistemas de ecuaciones.

| | | |
|---|---|---|
| **MATLAB: Gráficos** | MATLAB: Gráficas 2D con la función `plot` | 6:41 |
| | MATLAB: Cómo personalizar gráficas con las funciones `plot`, `text`, `axis` y otras | 7:52 |
| | Ejercicio MATLAB: Modelizar y graficar la población de los Estados Unidos | 1:55 |
| **Duración total** | | 16:28 |
| **Duración promedio** | | 5:29 |

Tabla 5.5 : Lista de reproducción Crear y personalizar gráficas 2D en MATLAB con `plot`

# Capítulo 6: Conclusiones, objetivos alcanzados y futuros desarrollos

En el presente capítulo, se propondrá un análisis crítico detallado y se expondrán las conclusiones derivadas de este TFG.

En primer lugar, se llevará a cabo una evaluación del contenido generado durante la realización del proyecto. Con ello se buscará identificar tanto los puntos fuertes como las áreas de mejora en lo que respecta a la producción de material audiovisual para la enseñanza de MATLAB.

Finalmente, se presentarán una serie de sugerencias y propuestas orientadas a la evolución futura de este trabajo. Dichas recomendaciones surgieron de la experiencia adquirida durante la realización de este TFG y están destinadas a orientar y enriquecer las futuras investigaciones y desarrollos en el campo de la enseñanza de MATLAB a través de material audiovisual.

# 6.1. Conclusiones

Este TFG ha permitido consolidar y expandir el conocimiento adquirido previamente por el autor en esta disciplina. Este contenido multimedia no solo ha servido como material de aprendizaje para el autor, sino que también se emplea como un medio para la difusión del conocimiento, brindando a otros la posibilidad de adquirir habilidades y comprender conceptos técnicos fundamentales relacionados con MATLAB. Como resultado, este proyecto, cumple los objetivos que nos marcamos inicialmente y, además, los logros obtenidos seguirán siendo relevantes y aportando beneficios en los años por venir.

En cuanto a la calidad de los vídeos, esta se ha logrado gracias a la supervisión del tutor de esta asignatura, teniendo que modificar detalles constantemente, hasta el último momento en cada vídeo.

Con respecto a las dos estrategias de creación de vídeos, descritas en el Capítulo 4, nuestras conclusiones son que el uso de PowerPoint para animar diapositivas y luego generar un vídeo resultó ser más conveniente en la mayoría de los casos, ya que permite mayor claridad en las explicaciones. Sin embargo, el uso del programa OBS resultó ser una mejor opción para explicar conceptos básicos sobre el entorno y los tipos de archivos de MATLAB, ya que permite capturar la pantalla y el cursor para guiar mejor a la audiencia durante estas explicaciones.

Respecto a la calidad del audio, se logró crear grabaciones de calidad sin la presencia de ruidos molestos que pudieran dificultar la comprensión, y con un volumen aceptable y estable. Para lograrlo, se tuvieron que aprender a manejar las herramientas de edición de audio descritas en el Capítulo 3. Como se mencionó en el Capítulo 4, debido a la falta de habilidades en la edición de audio al principio del proyecto, se utilizó la herramienta de inteligencia artificial Adobe Enhance, presentada en la Sección 3.5. A pesar de que esta herramienta ofrecía un resultado aceptable, el autor considera que modifica su entonación, ya que fue entrenada principalmente para el idioma inglés, y por tanto, no cree que el resultado sea adecuado. Por ello, el autor tuvo que mejorar sus habilidades de grabación y edición para lograr resultados óptimos sin recurrir a dicha herramienta de inteligencia artificial. De los 27 vídeos que componen este trabajo, solo se utilizó Adobe Enhance en 5 de ellos, específicamente en los vídeos titulados: “Entorno de MATLAB y conceptos básicos”, “Conceptos básicos sobre variables y archivos .mat en MATLAB”, y “Scripts o archivos .m de MATLAB: creación y ejecución”.

## 6.2. Objetivos alcanzados

A lo largo del proceso de desarrollo de este TFG, se ha hecho un esfuerzo constante por lograr los objetivos propuestos al comienzo del proyecto (Capítulo 1). En términos generales, se puede afirmar que los objetivos se han cumplido con éxito. Particularmente nos gustaría destacar lo siguiente:

1. Se ha proporcionado contenido educativo de calidad a través de 27 vídeos bien organizados y atractivos, con un enfoque claro en temas específicos de MATLAB con estos datos generales:
    - Duración media de los vídeos (ver Sección 5.2): 6 minutos y 50 segundos.
    - Duración total de todos los vídeos (ver Sección 5.2): 3 horas, 4 minutos y 41 segundos.
    - Tiempo estimado de trabajo por cada segundo de vídeo final (ver Sección 4.2): 204 segundos.
    - Tiempo invertido para la elaboración del contenido audiovisual (aproximación, ver Sección 4.2): 627 horas.
2. Cada vídeo ha sido cuidadosamente diseñado para facilitar la comprensión y mejorar la experiencia de aprendizaje de los estudiantes.
3. Se ha conseguido mantener la duración de cada vídeo entre 3 y 12 minutos, a excepción de algunos casos en los que se han superado por poco esos doce minutos.
4. Se han cuidado numerosos detalles para asegurar una presentación efectiva y profesional. Cada vídeo cuenta con una carátula, una despedida, y un título preciso, así como una descripción en modo texto que sirve como un índice de contenidos. También se cumplen las normas de calidad que se establecieron en el Capítulo 1.
5. Se ha garantizado un audio claro, sin ruidos de fondo y a un volumen adecuado. Se han cuidado aspectos tales como la pronunciación, el ritmo del habla, entre otros.

6. Los términos técnicos se han presentado de manera escrita la primera vez que se utilizan, y se ha hecho uso de diversas herramientas visuales para resaltar las partes más importantes del contenido, en el momento en el que la explicación lo requería.
7. Los metadatos de cada vídeo se han seleccionado cuidadosamente para atraer a la audiencia adecuada. Los títulos son atractivos y representativos del contenido de los vídeos. Las palabras clave se han utilizado de manera efectiva para proporcionar información relevante sobre el contenido de los vídeos. Cada vídeo incluye tarjetas de sugerencia de otros vídeos del mismo curso, recomendaciones finales y un botón de suscripción, siguiendo las posibilidades que ofrece YouTube [3].
8. En lo que respecta a los objetivos de temario abarcado en los vídeos, es posible afirmar que han sido alcanzados, pues se ha cubierto prácticamente todo el Tema 2 de la asignatura Sistemas Informáticos citada anteriormente.

## 6.3. Futuros desarrollos

Los trabajos futuros pueden centrarse en la continua creación de vídeos educativos sobre MATLAB, específicamente completando los conceptos no abordados del Tema 2 de la asignatura Sistemas Informáticos. Además, sería estupendo generar contenido audiovisual docente para el resto de los temas de dicha asignatura.

Por supuesto, sería fantástico que otras asignaturas afines fueran desarrollando material audiovisual que completara la docencia presencial. Estos podrían incluir, por ejemplo, asignaturas de Matemáticas o de Automática y Control, donde MATLAB se ha convertido en una herramienta esencial.

Otro tipo de desarrollos futuros que se pueden proponer son:

- Añadir otros idiomas a los vídeos, particularmente el inglés, pero también otros como el francés o el árabe.
- Añadir subtítulos a los vídeos ya creados para que el material pueda ser aprovechado mejor.
- Extender el contenido a otros temas interesantes tales como algoritmos específicos, como pueden ser algoritmos genéticos, redes neuronales, etc.

# Anexo A:
# Enlaces de vídeos creados y publicados

[1] Qué es MATLAB. https://youtu.be/HwI6DrdpEXo

[2] Entorno de MATLAB y conceptos básicos. https://youtu.be/vJ9MWlt7E-Q

[3] MATLAB: Conceptos básicos sobre variables y ficheros .mat. https://youtu.be/5GClQ4TtTTA

[4] Scripts o ficheros .m de MATLAB: creación y ejecución. https://youtu.be/ZOuObL73plA

[5] MATLAB aplicado a la astrofísica: ¿En cuánto tiempos se consumirá toda la masa del SOL? https://youtu.be/8MFWgCNFuSk

[6] Formatos de salida de MATLAB y notación científica. https://youtu.be/VrzVNoHKCOI

[7] Nombres permitidos y comando `help` en MATLAB. https://youtu.be/xTxSQVBXN4M

[8] Números complejos en MATLAB. https://youtu.be/wP8yKY-XZIY

[9] Funciones matemáticas elementales en MATLAB. https://youtu.be/ZQgLDw6jMdo

[10] Resolviendo ecuaciones lineales con MATLAB. https://youtu.be/UNnKeL8TcQs

[11] MATLAB: Cálculo simbólico para solucionar ecuaciones. https://youtu.be/jwfxnhxI3cg

[12] Polinomios en MATLAB. https://youtu.be/9TgZEehsoJ0

[13] MATLAB: Problema de cálculo simbólico sobre el crecimiento bacteriano. https://youtu.be/ZAukxPvrucs

[14] MATLAB: Crear vectores y matrices. https://youtu.be/Vq08gKCfP9Q

[15] MATLAB: Composición de vectores y matrices. https://youtu.be/P8Hz__-UjNU

[16] MATLAB: Direccionar y modificar vectores y matrices. https://youtu.be/_YxnLdphIVY

[17] MATLAB: Operador dos puntos (`:`). https://youtu.be/dYzSmQpUHMc

[18] MATLAB: Ajuste polinomial a partir de un conjunto de puntos. https://youtu.be/PN0CmJFrcPs

[19] MATLAB: Operaciones de escalares con vectores y matrices. https://youtu.be/DuL4_LTTJjA

[20] MATLAB: Transponer vectores y matrices con números reales y complejos. https://youtu.be/PxsJryn9pC0

[21] Funciones de MATLAB para vectores y matrices. https://youtu.be/1nOO2EJ0SnY

[22] Comandos sencillos, pero útiles, como `tic` y `toc`. https://youtu.be/n_H3qld2BmM

[23] MATLAB: Generación de números pseudoaleatorios con las funciones `rand`, `randn` y `randi`. https://youtu.be/J7HMadKjkQU

[24] MATLAB: Gráficas 2D con la función `plot`. https://youtu.be/4xJd96QK5y0

[25] Ejercicio MATLAB: Modelizar y graficar la población de los Estados Unidos. https://youtu.be/MQltr37ERII

[26] MATLAB: operaciones entre vectores y matrices. https://youtu.be/mg2rkn06f-U

[27]MATLAB: Cómo personalizar gráficas con las funciones `plot`, `text`, `axis` y o

t

# Referencias

[1] Menéndez Dávila J., Galeas Guijarro E., & Avilez Merino R. (2018). Las tecnologías de la información y las comunicaciones y su impacto en el rendimiento financiero de las empresas. apuntes didácticos. Opuntia Brava, 9(2), 236-244, p.6. Recuperado a partir de https://opuntiabrava.ult.edu.cu/index.php/opuntiabrava/article/view/165

[2] Ortiz Zambrano J., Sangacha Tapia L., & Alarcón Santillán J. (2018). Importancia de la programación en la formación de los ingenieros de sistemas computacionales. Opuntia Brava, 9(4), 94-100, p. 3. Recuperado a partir de https://opuntiabrava.ult.edu.cu/index.php/opuntiabrava/article/view/212

[3] YouTube. (web oficial). https://www.youtube.com/

[4] Canal de YouTube “Aprende conmigo Informática”, del Dr. José Galindo Gómez de la Universidad de Málaga: https://www.youtube.com/@aprendeconmigoinformatica/videos

[5] Khan Academy (web oficial). https://es.khanacademy.org/

[6] TED-Ed (web oficial). https://ed.ted.com/

[7] Coursera (web oficial). https://www.coursera.org/

[8] edX (web oficial). https://www.edx.org/

[9] Udacity (wed oficial). https://www.udacity.com/

[10] MiriadaX (web oficial). https://miriadax.net/

[11] Lenguaje de programación Python (web oficial). https://www.python.org/

[12] Biblioteca de Python Sympy (web oficial). https://www.sympy.org/en/index.html

[13] Biblioteca de Python Numpy (web oficial). https://numpy.org/

[14] Lenguaje de programación Java (web oficial). https://docs.oracle.com/javase/7/docs/technotes/guides/language/

[15] Lenguaje de programación C (web oficial). https://www.open-std.org/jtc1/sc22/wg14/

[16] Lenguaje de programación C++ (web oficial). https://isocpp.org/

[17] Lenguaje de programación C# (web oficial). https://learn.microsoft.com/en-us/dotnet/csharp/

[18] Lenguaje de programación JavaScript (web oficial). https://www.javascript.com/

[19] Lenguaje de programación Ruby (web oficial). https://www.ruby-lang.org/en/

[20] Lenguaje de programación PHP (web oficial). https://www.php.net/

[21] Lenguaje de programación R (web oficial). https://www.r-project.org/

[22] Lenguaje de programación MATLAB (web oficial). https://es.mathworks.com/products/matlab.html?s_tid=hp_products_matlab

[23] Lenguaje de programación Octave (web oficial). https://octave.org/

[24] Lenguaje de programación Julia (web oficial). https://julialang.org/

[25] Lenguaje de programación Mathematica (web oficial). https://www.wolfram.com/mathematica/

[26] Lenguaje de programación Maple (web oficial). https://www.maplesoft.com/

[27] Lenguaje de programación SageMath (web oficial). https://www.sagemath.org/

[28] Lenguaje de programación Fortran (web oficial). https://fortran-lang.org/es/index

[29] Lenguaje de programación Scilab (web oficial). https://www.scilab.org/

[30] Lenguaje de programación IDL (web de la nasa). https://www.nas.nasa.gov/hecc/support/kb/interactive-data-language-(idl)_119.html

[31] Entorno de programación LabVIEW (web oficial). https://www.ni.com/es-es/shop/software/products/labview.html

[32] Sistema de algebra computacional Maxima (web oficial). https://maxima.sourceforge.io/

[33] Sistema computacional para el algebra discreta GAP (web oficial). https://www.gap-system.org/

[34] Lenguaje de programación Haskell (web oficial). https://www.haskell.org/

[35] Galindo, J., Galindo, P., Corral, J.M.R. (2020). Multimedia System for Self-learning C/C++ Programming Language. In: Serrhini, M., Silva, C., Aljahdali, S. (eds) Innovation in Information Systems and Technologies to Support Learning Research. EMENA-ISTL 2019. Learning and Analytics in Intelligent Systems, vol 7. Springer, Cham. https://doi.org/10.1007/978-3-030-36778-7_7

[36] Universidad Nacional a Distancia (web oficial). https://www.uned.es/universidad/inicio/conocenos.html

[37] MATLAB Central (web oficial). https://es.mathworks.com/matlabcentral/

[38] MATLAB Centro de ayuda (web oficial). https://es.mathworks.com/help/matlab/

[39] Guía docente de la asignatura Sistemas Informáticos del grado en Ingeniería Eléctrica (web oficial). https://sara.uma.es/ht/2022/ProgramasAsignaturas_Titulacion_5044_AsigUMA_50960.pdf

[40] Microsoft PowerPoint (web oficial). https://www.microsoft.com/es-es/microsoft-365/powerpoint

[41] Audacity (web oficial). https://www.audacityteam.org/

[42] OBS Open Broadcaster Software (web oficial). https://obsproject.com/es

[43] Shotcut (web oficial). https://shotcut.org/

[44] Adobe enhance speech (web oficial). https://podcast.adobe.com/enhance

[45] Configuraciones de tasas de bits para emisiones en directo en YouTube (web Google support). https://support.google.com/youtube/answer/2853702?hl=es

[46] Audacity en SourceForge (web oficial). https://sourceforge.net/projects/audacity/

[47] Fernández Casasola S. (2023). Vídeos docentes para el aprendizaje de programación C/C++ en Ingeniería Industrial. TFG para el Grado de… de la Universidad de Málaga.