

**Máster en Profesor/a de Educación Secundaria
Obligatoria y Bachillerato, Formación Profesional y
Enseñanza de Idiomas
Especialidad de Matemáticas**



**UNIVERSITAT
JAUME•I**

Trabajo Fin de Máster

**Herramientas para la redacción de textos
científicos en el ámbito del profesorado y
estudiantado de Educación Secundaria
Obligatoria y Bachillerato**

María Espinosa Arnau

Tutor: Pablo Gregori Huerta

Curso académico 2018/2019

RESUMEN

El presente Trabajo Final de Máster se engloba dentro de la modalidad de materiales didácticos. En él se presentan las herramientas básicas necesarias para redactar de forma eficiente textos de carácter científico-matemático que incluyan cálculos, código de algún lenguaje de programación, imágenes o incluso fórmulas matemáticas. Se trata pues de una guía tanto para el alumnado como para el profesorado, a través de la cual no solo se podrá aprender el lenguaje de escritura Markdown, sino también el lenguaje de programación R.

De este modo, los objetivos principales del trabajo giran en torno al desarrollo de las competencias y conocimientos necesarios para poder utilizar de forma adecuada las herramientas que en él se proporcionan. Además, supone una primera toma de contacto con un software y un lenguaje de composición de documentos del que podrán obtener aún más ventajas en su futura formación superior.

En el trabajo se explica cómo manejar dichos programas informáticos, así como también la adaptación de estos a los contenidos curriculares de cada uno de los cursos en los que se ha considerado apropiada la introducción del software.

ÍNDICE

1	INTRODUCCIÓN	1
2	MARCO TEÓRICO.....	2
2.1	Justificación.....	2
2.2	Lectura y tecnología.....	3
2.3	Visualización y edición de documentos.....	6
2.3.1	Visualización.....	6
2.3.2	Edición de documentos.....	6
2.3.2.1	Edición de textos científicos	10
3	CURRÍCULO: COMPETENCIAS CLAVE	14
3.1	Visión teórica.....	14
3.2	Competencias clave en las matemáticas	15
4	OBJETIVOS	17
5	DESTINATARIOS.....	18
5.1	Alumnado.....	18
5.2	Profesorado.....	19
6	REDACCIÓN DE TEXTOS CIENTÍFICOS: EJEMPLOS	20
6.1	Alumnado.....	20
6.2	Profesorado.....	26
7	CONCLUSIÓN.....	38
8	REFERENCIAS.....	40
9	ANEXO 1: SOFTWARE.....	44
9.1	LaTeX	44
9.2	R.....	45
9.2.1	Descarga e instalación de R.....	46
9.2.2	Notas sobre aspectos básicos.....	47
9.2.3	Conceptos y funciones básicas	48
9.3	Markdown.....	54
9.3.1	¿Qué es Markdown?	54
9.3.2	Sintaxis y conceptos básicos de Markdown.....	56
9.4	R Markdown.....	60
9.4.1	Presentaciones con RMarkdown	62
9.5	RStudio.....	64
9.5.1	Descarga e instalación de RStudio	64
10	ANEXO 2: CONTENIDO CURRICULAR Y APLICACIÓN EN RSTUDIO.....	69
10.1	Paquetes de R específicos.....	69
10.1.1	Paquete Ryacas.....	69
10.1.2	Paquete Polynom.....	71
10.1.3	Paquete Matlib	72

10.2	4º ESO: Matemáticas orientadas a las enseñanzas académicas y a las enseñanzas aplicadas.....	73
10.2.1	Bloque: Números y álgebra.....	73
10.2.2	Bloque: Geometría.....	78
10.2.3	Bloque: Análisis.....	80
10.2.4	Bloque: Estadística y probabilidad.....	82
10.3	1º Bachillerato: Matemáticas científicas y aplicadas a las CCSS.....	86
10.3.1	Bloque: Números y álgebra.....	86
10.3.2	Bloque: Geometría.....	90
10.3.3	Bloque: Análisis.....	90
10.3.4	Bloque: Estadística y probabilidad.....	92
10.4	2º Bachillerato: Matemáticas científicas y aplicadas a las CCSS.....	98
10.4.1	Bloque: Números y álgebra.....	98
10.4.2	Bloque: Geometría.....	100
10.4.3	Bloque: Análisis.....	100
10.4.4	Bloque: Estadística y probabilidad.....	101
11	ANEXO 3: CÓDIGOS DE LOS ARCHIVOS .RMD DEL CAPÍTULO 6.....	105
11.1	Código del Ejemplo Ejercicios.....	105
11.2	Código del Ejemplo Presentación Ejercicios.....	108
11.3	Código del Ejemplo Apuntes.....	111
11.4	Código del Ejemplo Presentación Apuntes.....	115
11.5	Código del Ejemplo de Examen.....	118

1 INTRODUCCIÓN

Uno de los objetivos del sistema educativo es mostrar cómo sacar el máximo partido a los recursos tecnológicos en las aulas. Para ello, no basta con aprender a utilizar dichos recursos, sino que resulta conveniente poner especial énfasis en saber manejarlos de la forma más adecuada y eficiente posible. En este sentido, aunque las nuevas generaciones ya se desenvuelven perfectamente en la era de la información y las nuevas tecnologías, la gran mayoría del alumnado no sabe cómo valerse de las herramientas disponibles.

En el área de las matemáticas, se dispone de gran cantidad de programas informáticos que facilitan la comprensión o visualización de determinados conceptos y, además, favorecen la participación del alumnado en su propio proceso de aprendizaje: el docente establece los objetivos o contenidos mínimos, y es el propio alumno quien decide el máximo que desea alcanzar. No obstante, en los cursos de la Educación Secundaria Obligatoria y Bachillerato resulta inviable mostrar y explicar todos ellos. Por eso se ha considerado el programa RStudio como centro de desarrollo de la propuesta didáctica.

En mi formación universitaria como matemática, he observado la importancia de la correcta redacción de textos científicos, así como del manejo de los lenguajes de programación. De este modo, el Trabajo Final de Máster se desenvuelve en esta dirección.

Es primordial que el alumnado, en los cursos de la Educación Secundaria Obligatoria y Bachillerato, desarrolle una capacidad de abstracción y un correcto razonamiento formal, es decir, que pueda construir relaciones lógico-deductivas. Todo ello se ve favorecido en gran medida por la utilización de un lenguaje de programación, en este caso particular, R; pues para poder usarlo de forma adecuada, primero se debe entender el concepto en su forma abstracta para después aplicarlo a casos particulares.

Asimismo, en la gran mayoría de grados de modalidad científica se requiere que el alumnado tenga habilidades de escritura científica y por ello se desarrolla este trabajo, pues tiene por objetivo facilitar y potenciar la redacción de este tipo de textos desde la formación más básica.

Por tanto, el trabajo se presenta como una primera aproximación al mundo que envuelve el lenguaje de programación y sus aplicaciones en los diferentes bloques conceptuales considerados en los currículos de la asignatura de matemáticas. Todo esto para dotar, principalmente al alumnado, de las capacidades o competencias necesarias para desarrollar con éxito la escritura científica de una forma más simple.

2 MARCO TEÓRICO

2.1 Justificación

La sociedad de hoy se caracteriza sustancialmente por el vertiginoso desarrollo científico y tecnológico que propician un cambio estructural a niveles económicos, políticos o culturales; principalmente debidos a la revolución de las tecnologías de la información y comunicación.

Las tecnologías de la información y la comunicación (TIC) están ya inmersas en el día a día de la gran mayoría de los centros educativos, y estos, deben estar preparados para poder hacer frente a los avances y retos que suponen. Es una realidad que la aparición de estos recursos ha generado un cambio notable en el proceso de enseñanza-aprendizaje del profesorado y alumnado, obligando así al cuerpo docente a adaptarse a estos nuevos procedimientos. No solo se trata de la simple introducción de las TIC en la enseñanza, sino de diseñar situaciones didácticas que permitan la correcta integración de estas, adaptándose en todo momento al alumnado y a los objetivos formativos de cada etapa (del Moral Pérez, 1997).

Es por esto por lo que, debido al fuerte impacto de las TIC en la educación, las instituciones educativas han tenido que reestructurar los enfoques formativos hacia una nueva línea de actuación que permite el desarrollo de un aprendizaje significativo para el alumnado con el uso de las nuevas tecnologías. Así pues, no solo depende de los centros educativos, sino también de la formación previa que reciba el docente, ya que solo así se podrán transmitir con éxito los conocimientos a través de dichas herramientas. Tal como indican Del Moral y Villalustre (2010), la formación del profesorado debe estar ligada a la integración de las TIC al sistema educativo, ya que posibilitan la implementación de nuevas metodologías de enseñanza; además de ofrecer pautas didáctico-metodológicas que facilitan experiencias enriquecedoras de aprendizaje. Así pues, el papel de las TIC resulta relevante, ya que permiten flexibilizar y mejorar ciertos procesos que influyen en el aprendizaje, la organización escolar o la comunicación con la comunidad (González-Pérez y De Pablos, 2015).

Por otro lado, adquirir dominio con las TIC implica directamente aumentar una de las competencias básicas: la competencia digital, que tal como define el Ministerio de Educación y Formación Profesional:

“Es aquella que implica el uso creativo, crítico y seguro de las tecnologías de la información y la comunicación para alcanzar los objetivos relacionados con el trabajo, la empleabilidad, el aprendizaje, el uso del tiempo libre, la inclusión y participación en la sociedad” (Ministerio de Educación y Formación Profesional-Gobierno de España, s.f.)

En el área que nos incumbe, tal como indica Rosero (2018) para poder contribuir al desarrollo de la competencia digital del docente desde el área de matemáticas, resulta necesario realizar aportaciones con utilidad práctica, que trasciendan las aportaciones teóricas. En muchos casos, los alumnos adquieren habilidades en el cálculo, pero no comprenden el verdadero de sentido de lo que aprenden (Barriuso, Gómez, Haro y Parreño, 2013). En esta línea, las nuevas tecnologías favorecen, en ciertos casos y bien utilizadas, una comprensión más consolidada de los conceptos matemáticos.

Además, en el ámbito de las matemáticas, los nuevos recursos permiten a profesores y estudiantes tener al alcance herramientas que favorecen el desarrollo de nuevas capacidades cognitivas, contribuyen a la realización de cálculos complicados y ayudan en el análisis de los procesos característicos de la resolución de problemas, tal y como señalan Caravallo y Zulema (2009).

De esta manera, el presente trabajo pretende ser una herramienta al alcance de los docentes y también del alumnado, para poder adaptar las TIC a la enseñanza de las matemáticas y poder ser un medio para optimizar y llegar al aprendizaje significativo del alumnado.

Se trata de una guía para la edición de documentos científicos, que incluyen texto normal combinado con cálculos numéricos, gráficas y fórmulas matemáticas, todo ello con un programa sencillo y gratuito (RStudio); de forma que el resultado final sea un documento de texto ordinario, pero redactado de forma más cómoda ya que la inclusión de elementos matemáticos resulta más sencilla. Trabajo que resultaría más costoso utilizando un editor de textos corriente, como por ejemplo Word. Por tanto, también se trabaja y potencia la redacción de textos de carácter científico-matemático, inculcando así la importancia de una correcta expresión escrita también en el área de las matemáticas. Así pues, se requerirá que el alumno esté dotado de una base matemática sólida y de una capacidad de abstracción desarrollada, ya que los programas que se utilizarán combinan partes de programación informática.

Se trabajará con el lenguaje de programación R para realizar los cálculos, con la sintaxis de LaTeX para poder escribir las fórmulas matemáticas cómodamente, y con la sintaxis de Markdown para estructurar de forma simplificada los textos. Todo ello a través de una interfaz sencilla llamada Rstudio, y gracias a una librería de R llamada *RMarkdown*, que permite fusionar todos estos aspectos en un sencillo flujo de trabajo, que sería muy enrevesado con cualquier otro recurso.

2.2 Lectura y tecnología

Las nuevas tecnologías no solo han aterrizado en el mundo educativo tal como comentamos en el apartado anterior, sino que también lo han hecho en otros ámbitos como el de la lectura. Además, el uso masivo de ordenadores y la conexión a Internet están propiciando un cambio por lo que respecta a las formas tradicionales de lectura, imponiendo así la lectura digital (Valencia, 2006).

Como señala Gutiérrez (2009), la idea de lectura se ha ido flexibilizando, ya que, de este modo, puede abarcar un conjunto de usos e intercambios de códigos diversos. Así pues, la lectura no es un hecho meramente racional, como indica Brunner (1988), sino que se ha transformado en un proceso en el que confluyen diferentes factores. De este modo, la lectura tradicional empieza a compartir su liderazgo con los textos electrónicos leídos en las pantallas de los ordenadores, y que día a día crecen exponencialmente (Valencia, 2006). Es evidente que la lectura digital presenta grandes ventajas, ya que el lector puede estructurar o incluso reestructurar a su gusto los textos gracias a las herramientas de edición que presentan los formatos digitales, también señalar, subrayar, anotar, etc.

Además, no importa el origen de dichos textos digitales, ya que la Red permite al usuario acceder a ellos cualquier día, a cualquier hora, sin importar su procedencia y el lugar en el que esté el lector en ese momento. Así, la “la digitalización permite la comunicación de textos a distancia, anulando la barrera física infranqueable entre el lugar del texto y el lugar del lector” (Valencia, 2006).

Claro está, que, así como la incorporación de las nuevas tecnologías en los centros educativos implicaba un cambio en las metodologías, así como también en la formación del docente; la incorporación de estas a la lectura también pide un cambio en el perfil del lector, ya que las habilidades requeridas y desarrolladas no serán las mismas que en el caso de la lectura más tradicional (entiéndase, lectura en papel impreso). El lector digital es capaz de buscar, asociar ideas y relacionarlas con múltiples recursos, manipular gran cantidad de información, seleccionar, reordenar e incluso descubrir por azar información útil. En este contexto, la Asociación Internacional de la Lectura señala que:

“Los textos electrónicos presentan nuevas ayudas y también nuevos retos que pueden tener gran impacto sobre la capacidad que tiene el individuo de comprender lo que lee. Internet en especial, ofrece nuevos formatos de texto, nuevos propósitos para la lectura y nuevas maneras de interactuar con la información, que pueden confundir y hasta abrumar a las personas acostumbradas a extraer significados únicamente de los impresos convencionales.” (Citado en Valencia, 2006)

Por tanto, resulta evidente que los formatos de texto que permitan una lectura desde cualquier dispositivo electrónico como móvil, ordenador, tableta e incluso e-book, han llegado para quedarse. No solo por su gran versatilidad sino también por su gran capacidad de adaptación a cualquier tipo de lector.

Aquí es donde entran en juego los diferentes formatos de los que disponemos para la lectura en formato digital. Por un lado, está el HTML. Este es un lenguaje de marcas de hipertexto (HyperText Markup Language) que se utiliza principalmente para la elaboración de páginas de Internet (Wikipedia: HTML, s.f). No incluye el diseño gráfico de las webs, sino que solo sirve para indicar como va ordenado el contenido que aparecerá (¿Qué es HTML?, s.f). Por tanto, es un lenguaje que se

caracteriza no solo por su legibilidad cuando se compila, ya que el resultado es un texto estructurado y agradable para leer; sino también por la navegabilidad que presenta, permitiendo al lector desplazarse a cualquier sitio web a través de los enlaces que podrá incluir. Además, existen numerosas aplicaciones que generan el código HTML automáticamente, de este modo, no es necesario saber programarlo. También es uno de los lenguajes más extendidos ya que todos los navegadores lo admiten (JLPM, s.f.).

Por otro lado, el formato EPUB (Electronic Publication) “es un formato redimensionable de código abierto para leer textos e imágenes” (Wikipedia, EPUB; s.f.). Asimismo, como se indica en la web del Ministerio (Ministerio de la presidencia, relaciones con las cortes e igualdad, s.f.), el formato EPUB es independiente en cuanto a contenido y forma, es decir, puede ser adaptado a cualquier dispositivo de lectura: ordenador, móvil, tableta o e-book. Además, cualquier formato puede convertirse a EPUB de forma gratuita a través de la herramienta Calibre (Wikipedia, Calibre; s.f.).

El formato más conocido es el PDF (Portable Document Format), que como bien su nombre indica, significa archivo con formato de documento portátil. Además, este formato es independiente del software, el hardware o el sistema operativo (Adobe, 2019). En sus comienzos, los archivos PDF no admitían enlaces, por que lo que su uso en la Red resultaba poco práctico. No obstante, esto ha cambiado (Porto y Gardey, 2009). Además, muchos programas informáticos, tanto libres como de pago, han adaptado sus versiones para poder visualizar, crear o modificar los archivos PDF, ya que se trata del tipo de documento por excelencia usado tanto en centros educativos, empresas, como en gobiernos (Wikipedia: PDF, s.f.).

Por último, no olvidar el formato *.doc*: el formato cerrado que utiliza Microsoft Word. Este tipo de archivo se creó como un documento de texto sin formato, con gran cantidad de funcionalidades. No obstante, esta extensión no resultó compatible con muchos otros procesadores de texto, es decir, “las ventajas funcionales que ofrecen los archivos DOC se convierten en inconvenientes de compatibilidad con otros programas debido al formato inadecuado por falta de ingeniería inversa” (DOC, Microsoft Word Binary File Format (.doc), s.f.)

Debido al factor compatibilidad y a la presión de la competencia de Open Office con su código abierto libre y formato abierto de documento (ODF), Microsoft se vio ante la necesidad de adoptar un nuevo estándar. Desarrollaron el “Office Open XML” (Checa, 2018), un formato de archivo abierto y estándar basado en el lenguaje XML (*Extensible Markup Language*), que supuso la mejora del archivo *.doc*. La extensión de los documentos Word paso a ser *.docx*. Además, estos archivos no presentan los problemas de compatibilidad que presentaban los *.doc*, ya que editores de texto gratuitos como Open Office, o Libre Office, e incluso herramientas en línea como Google Docs, son compatibles con él. Asimismo, son fácilmente convertibles a otros formatos como *.pdf, .txt, .doc, .html, .xml ...*

2.3 Visualización y edición de documentos

2.3.1 Visualización

Como se ha mencionado en el apartado anterior, existen diferentes formatos de visualización de documentos. Los PDF son inalterables en lo referente al formato, es decir, todos los elementos quedan fijados. De hecho, son el formato por excelencia para enviar trabajos, currículos, informes o cualquier documento de carácter oficial. A parte, no presentan problemas de compatibilidad con los diferentes sistemas operativos. En los archivos *.pdf* se permite modificar el zoom para ajustar el tamaño de la letra, también el modo de visualización ya sea una página o dos páginas; también se puede resaltar el texto, añadir notas e incluso firmar electrónicamente. No obstante, no se adaptan correctamente a la pantalla, por lo que el lector debe desplazarse por las páginas para leerlas en su totalidad, hecho que puede resultar tedioso. Por lo que respecta a los archivos *.docx*, son similares a los anteriores, pero con más facilidad para la edición. No obstante, pese a ser la versión mejorada de los archivos *.doc*, presentan más problemas que los anteriores, ya que, si un archivo se crea y se abre en versiones de Word diferentes, puede ocasionar errores en la presentación del documento, la colocación de las imágenes o tablas. Es decir, en el diseño. Aun así, ambos tipos de archivos se adaptan perfectamente para poder ser imprimidos sin ningún tipo de problema.

En lo referente a las páginas web, al contrario que con los dos formatos anteriores, estas adaptan su formato en función del dispositivo que se utilice: pantalla grande o pequeña. Sí es cierto que este ajuste puede presentar algún problema en lo que respecta a tablas o imágenes. Además, al igual que los documentos *.pdf* y *.docx*, pueden modificarse aspectos como el zoom, lo cual permite la adaptación a cualquier usuario.

No obstante, los archivos PDF y Word, ocupan, en general, más espacio de almacenamiento ya sea en la memoria del ordenador, de la tableta, teléfono o memoria portátil. Además, mediante los buscadores web, accedemos principalmente a páginas web, si bien es cierto que también encontramos en la red archivos PDF o Word. Aun así, las páginas web, no solo favorecen la adaptabilidad a los dispositivos, si no también la navegación a través de ellas o la interacción. Además, según un estudio realizado por Elsevier¹, la visualización mediante HTML se prefería para descubrir nuevo contenido, aprendizaje inmediato y para compartir información. Igualmente, las actualizaciones pueden publicarse rápidamente con una URL, hecho que garantiza que todos los usuarios lectores reciban simultáneamente la información más actualizada. (TerraXML, s.f.)

2.3.2 Edición de documentos

En cuanto a la edición de documentos, necesitamos disponer del software adecuado.

¹ Editorial de libros de medicina y literatura científica.

Archivos DOC

Para los archivos *.doc* o *.docx*, los más utilizados son Word o LibreOffice Writer, pues es de código abierto y gratuito. Igualmente existen otras herramientas o editores compatibles con este tipo de documentos como Word Online, la versión gratuita de Word, aunque no tan completa, Google Docs (dentro de Google Drive) que aun siendo más limitado que Word contiene los aspectos básicos y además, existe una extensión de Google Chrome que permite editar los textos sin necesidad de estar conectado a la red. También WPS Writer, editor gratuito, que forma parte de WPS Office y además su interfaz es muy similar a la de Word.

Es evidente, que la edición en Word resulta muy cómoda, ya que el usuario puede utilizarlo de forma totalmente intuitiva, pues dispone de botones que permiten modificar sin problemas cualquier aspecto del documento. Además, el archivo puede visualizarse previamente de diferentes formas y también se permite guardarlo en distintos formatos (*.doc*, *.docx*, *.pdf*, *.rtf*, *.txt*, *.html*, *.odt*, *.xml*, *.docm*...). El software comentado pertenece al grupo de editores de tipo WYSIWYG (*What You See Is What You Get: Lo que ves, es lo que obtienes*), es decir, “procesadores de texto que permiten escribir un documento mostrando directamente el resultado final” (Wikipedia, WYSIWYG; s.f)

Archivos PDF

Si lo que pretendemos es editar un PDF, la cosa se complica. Los archivos PDF suelen producirse a partir de otros formatos, como por ejemplo Word o LaTeX. Actualmente existen editores de documentos con extensión *.pdf*, no obstante, solo resultan cómodos si el objetivo es retocar algún aspecto mínimo. Si la intención es crear un documento de cero en formato PDF, no es nada práctico.

Así pues, lo habitual es crear documentos *.pdf*, bien a partir de un archivo *.doc* o *.docx* tal como se señala anteriormente, o bien a partir de LaTeX. Este último es un lenguaje que crea documentos PDF a partir de un archivo de texto con instrucciones y comandos especiales necesarios para estructurar de forma adecuada el documento (extensión *.tex*). En general, es una escritura costosa y poco intuitiva con una curva de aprendizaje lenta, ya que se requieren gran cantidad de comandos. Es por eso por lo que existen programas como Scientific Workplace que resultan un poco más amigables que LaTeX, pero su efectividad no es tanta como se espera. No obstante, utilizar LaTeX una vez se domina resulta 'cómodo' si se pretende incluir fórmulas o lenguaje matemático, ya que este editor es muy potente para la escritura matemática (véase Figura 2.1 y Figura 2.2). En el apartado 9.1 del Anexo 1, se incluye una descripción detallada de LaTeX.

```

39 \section{Exercicis.}
40
41 Amb la notació  $\nu = ak/h^2$ , considereu els mètodes
42 \bigskip
43
44 \begin{tabular}{l}
45 \textbf{Euler Explícit}& \\
46 \boxed{u_j^{n+1} =} & \\
47  $u_{(j)}^n + \nu(u_{(j+1)}^n - 2u_{(j)}^n + u_{(j-1)}^n)$  & \\
48 \textbf{Euler Implícit}& \\
49 \boxed{u_j^{n+1} =} & \\
50  $u_{(j)}^n + \nu(u_{(j+1)}^{n+1} - 2u_{(j)}^{n+1} + u_{(j-1)}^{n+1})$  & \\
51 \textbf{Crank-Nicolson}& \\
52 \boxed{u_j^{n+1} =} & \\
53 = & \\
54  $u_j^n +$  & \\
55  $0.5\nu(u_{(j+1)}^n - 2u_j^n + u_{(j-1)}^n$  & \\
56  $+ u_{(j+1)}^{n+1} - 2u_j^{n+1} + u_{(j-1)}^{n+1})$  & \\
57 \end{tabular}
58 \bigskip
59
60
61 per a la resolució del problema de valors inicials
62 $$
63  $u_t - au_{xx} = 0, u(x, 0) = u^0$ 
64 $$
65 $$
66 en el domini  $(0, 1) \times (0, T)$  amb condicions Dirichlet homogènies
67  $u(0, t) = u(1, t) = 0$ , les quals s'implementen com:
68 \begin{align*}
72 i on  $u_j^n \approx u(x_j, t_n)$ ,  $x_j = jh$ ,  $t_n = nk$ ,  $h = 1/(M+1)$ ,
73  $k = T/N$ .
74
75 Es demana:
76
77 \begin{enumerate}
78 \item Doneu la forma matricial de cadascun dels mètodes
79 \begin{equation} \label{eq:1}
80 u^{n+1} = H u^n, \quad u^n \in \mathbb{R}^M,
81 \end{equation}
82 amb  $H = H(A)$ , on  $A = A_M$  ve donada per
83 \begin{equation*}
84 A =
85 \begin{bmatrix}
86 -2 & 1 & 0 & \dots & 0 \\
87 1 & -2 & 1 & \dots & 0 \\
88 \vdots & \vdots & \vdots & \ddots & \vdots \\
89 0 & \dots & 1 & -2 & 1 \\
90 0 & \dots & 0 & 1 & -2
91 \end{bmatrix}.
92 \end{equation*}
93
94
95 \begin{itemize}
96 \item Mètode Euler Explícit \\
97 Utilitzant les condicions de Dirichlet,  $u_0 = 0$ ,
98  $u_{M+1} = 0$  i substituint en l'equació del mètode obtenim:
99  $u_1^{n+1} = u_1^n + \nu(u_2^n - 2u_1^n + u_0^n) = u_1^n + \nu(u_2^n - 2u_1^n)$ 
100  $u_i^{n+1} = u_i^n + \nu(u_{i+1}^n - 2u_i^n + u_{i-1}^n)$ ,  $\forall i = 2, \dots, M-1$ 


```

Figura 2.1 Còdigo LaTeX
Fuente: Elaboración propia



2 Exercicis.

Amb la notació $\nu = ak/h^2$, considereu els mètodes

Euler Explícit	$u_j^{n+1} = u_j^n + \nu(u_{j+1}^n - 2u_j^n + u_{j-1}^n)$
Euler Implícit	$u_j^{n+1} = u_j^n + \nu(u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1})$
Crank-Nicolson	$u_j^{n+1} = u_j^n + 0.5\nu(u_{j+1}^n - 2u_j^n + u_{j-1}^n + u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1})$

per a la resolució del problema de valors inicials

$$u_t - au_{xx} = 0, u(x, 0) = u^0$$

en el domini $(0, 1) \times (0, T)$ amb condicions Dirichlet homogènies $u(0, t) = u(1, t) = 0$, les quals s'implementen com:

$$u_0^n = 0$$

$$u_{M+1}^n = 0$$

i on $u_j^n \approx u(x_j, t_n)$, $x_j = jh$, $t_n = nk$, $h = 1/(M+1)$, $k = T/N$.

Es demana:

1. Doneu la forma matricial de cadascun dels mètodes
$$u^{n+1} = H u^n, \quad u^n, u^{n+1} \in \mathbb{R}^M, \tag{1}$$
amb $H = H(A)$, on $A = A_M$ ve donada per
$$A = \begin{bmatrix} -2 & 1 & 0 & \dots & 0 \\ 1 & -2 & 1 & \dots & 0 \\ \dots & \dots & \dots & \ddots & \dots \\ 0 & \dots & 1 & -2 & 1 \\ 0 & \dots & 0 & 1 & -2 \end{bmatrix}.$$
 - **Mètode Euler Explícit.**
Utilitzant les condicions de Dirichlet, ($u_0^n = 0, u_{M+1}^n = 0$) i substituint en l'equació del mètode obtenim:
$$u_1^{n+1} = u_1^n + \nu(u_2^n - 2u_1^n + u_0^n) = u_1^n + \nu(u_2^n - 2u_1^n)$$

1

Figura 2.2 Resultado PDF del código LaTeX
Fuente: Elaboración propia

Páginas Web (Archivos HTML)

Para las páginas web (o HTML), al igual que con los PDF, tenemos la alternativa de escribir en Word y guardarlo en este caso con el formato HTML que es el propio de las páginas web. Otra opción sería utilizar el lenguaje HTML directamente, opción poco recomendable, pues resulta un poco tedioso, ya que, para escribir un acento, resaltar una palabra utilizando la negrita o escribir un párrafo necesitamos introducir una instrucción poco intuitiva.

En la Figura 2.3 se observa el código HTML y en la Figura 2.4 el resultado que obtendríamos al compilar dicho código.

```
1 <html>
2 <head>
3   <title>
4     <h1> Título: Ejemplo lenguaje HTML </h1>
5   </title>
6 </head>
7
8 <body>
9 Hola mundo! Para insertar un salto de línea:
10 "br" .<br> Como vemos para poner un acento necesitamos
11 la instrucción "&acute;". <br>
12 Siguiete línea <br>
13 <br>
14 <h1> Encabezado tipo 1 </h1>
15 <h2> Encabezado tipo 2 </h2>
16 <h3> Encabezado tipo 3 </h3>
17
18 <p> Si queremos insertar un párrafo, deberemos
19 insertar entre los símbolos<> la letra 'p'. De esta
20 forma, las siguientes líneas se iniciarán como si
21 hubiéramos pulsado la tecla INTRO.
22 </p>
23
24
25
26 Para poder escribir en <b> negrita </b>, insertamos
27 la 'b'entre <>, y para hacerlo en
28 <i> italiana </i>,
29 la letra 'i' entre <>.
30 </p>
31
32
33 Para insertar una imagen con su título:
34
35 <br /><p>
38 class='caption'>Logo Universitat Jaume I</p>
39
40 Y para un hipervínculo:
41 <a href="https://www.uji.es/">
42 Enlace a la página de la UJI</a>
43
44 </body>
45 </html>
46
47
48
49
```

Figura 2.3 Código HTML
Fuente: Elaboración propia



Título: Ejemplo lenguaje HTML

Hola mundo! Para insertar un salto de línea: "br" .
 Como vemos para poner un acento necesitamos la instrucción "& acute".
 Siguiente línea

Encabezado tipo 1

Encabezado tipo 2

Encabezado tipo 3

Si queremos insertar un párrafo, deberemos insertar entre los símbolos<> la letra 'p'. De esta forma, las siguientes líneas se iniciarán como si hubiéramos pulsado la tecla INTRO.

Para poder escribir en **negrita** , insertamos la 'b'entre <>, y para hacerlo en *itálica* , la letra 'i' entre <>.

Para insertar una imagen con su título:



Logo Universitat Jaume I

Y para un hipervínculo: [Enlace a la página de la UJI](#)

Figura 2.4 Resultado del código HTML
 Fuente: Elaboración propia

2.3.2.1 Edición de textos científicos

Uno de los objetivos del trabajo recae en la redacción de textos de carácter científico. Estos textos se caracterizan principalmente por el uso de fórmulas matemáticas, desarrollo de expresiones, resolución de ecuaciones, código, cálculos varios, incluso gráficos de funciones o de un conjunto de datos. Para ello, tenemos los editores comentados en el apartado anterior.

En el caso de la edición de los formatos .doc, .docx y .html, , esencialmente utilizamos Word. En este, tal como se observa en la Figura 2.5, se dispone de un apartado específico llamado ' Ecuación' que permite insertar ecuaciones.

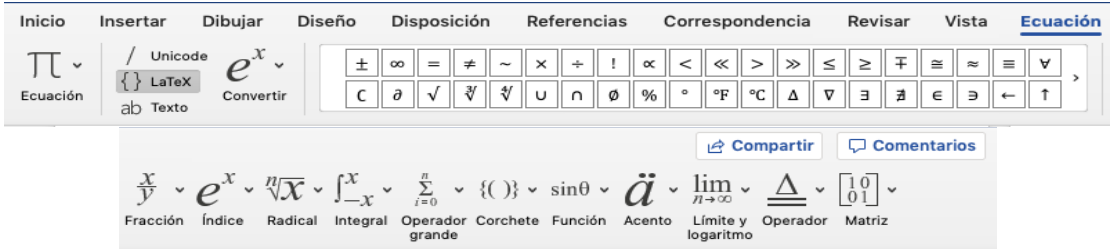


Figura 2.5 Opciones de Word para insertar Ecuaciones
 Fuente: Elaboración propia

No obstante, es poco práctico, pues para insertarlas se debe hacer 'click' sobre el símbolo que quiere insertarse y escribir en los recuadros blancos que quedan. Por tanto, si queremos incluir un número considerable de fórmulas resultará un proceso costoso.

Para la edición de PDF, se había considerado la utilización de LaTeX. En el caso de textos de carácter científico, es altamente recomendable, aunque la curva de aprendizaje sea lenta. Pues una vez se aprenden los comandos y funciones necesarias, la escritura con símbolos matemáticos resulta mucho más rápida. Lo verdaderamente costoso es dar formato al documento en LaTeX, por eso será conveniente utilizar la parte de lenguaje LaTeX para las ecuaciones mientras que, en lo referente al texto, a su estructura y escritura, utilizaremos otro editor que más adelante señalaremos. Para la escritura de fórmulas matemáticas con LaTeX, ver [apartado 9.1](#) del Anexo 1.

Por otro lado, podemos estar interesados en incluir cálculos, gráficas o análisis de datos numéricos realizados con otro software o programa. En este caso, se debe copiar el código del programa utilizado al editor de textos y también los resultados; y en el caso de las imágenes, guardarlas e incluirlas en el nuevo documento. Notar que este proceso a parte de ser lento puede producir un gran número de errores. Por tanto, sería muy apropiado encontrar la forma de poder comunicar el editor de textos con el programa que se usa para realizar los cálculos, y de esta forma, obtener de forma automática el código y sus resultados, bien numéricos o gráficos, en nuestro documento final.

Primeramente, para realizar los cálculos se utilizará el software estadístico R, un entorno diseñado principalmente para el tratamiento de datos, cálculo y desarrollo gráfico. Para ello se necesita conocer la sintaxis adecuada para poder llevar a cabo los diferentes cálculos y análisis de datos. En el [apartado 9.2](#) del Anexo 1 se detallan los aspectos más importantes de R, así como también como proceder a su instalación y los comandos principales.

Así, para poder llegar a la integración entre el código, sus salidas y el editor de textos, una de las opciones sería el paquete *Sweave* de R. Como se indica en el manual del propio paquete, “Sweave proporciona un marco flexible para mezclar texto y código R con el objetivo de generar documentos de forma automática”. Internamente lo que se hace es: en el documento en formato *.tex* (formato de LaTeX), se incluyen bloques identificados de código R con una estructura definida. Cuando se compila el documento, se llama al programa R que es el encargado de ejecutar los bloques de código que estén incluidos en él, y se genera un documento PDF que incluye texto, código y el resultado de este, bien numérico o bien en imagen. No obstante, este no será el elegido, pues se necesita conocer toda la sintaxis de LaTeX para estructurar el documento.

Llegados a este punto, ¿de qué forma podemos facilitar la escritura de texto plano, las fórmulas matemáticas, y la integración de código y sus salidas?

En cuanto a las fórmulas, símbolos, ecuaciones o desarrollos matemáticos, el lenguaje a utilizar, tal como se ha indicado anteriormente, será el de LaTeX. En referencia al texto, ¿existe algún lenguaje que permita escribir sin necesidad de utilizar muchas instrucciones, y que el resultado sea un texto bien estructurado y con los aspectos básicos? Markdown.

Markdown es un lenguaje de marcado ligero, que tiene por finalidad hacer que el usuario sea capaz de escribir usando un formato de texto fácil de leer y escribir, e incorporar la posibilidad de convertir el documento en un HTML estructuralmente válido (Wikipedia, Markdown; s.f) Es decir, como se observa en la Figura 2.6, permite redactar el texto correctamente y bien estructurado, visualizarlo casi en su forma final y facilitar su modificación y la salida en formato HTML sin necesidad de escribir en sintaxis de HTML (hecho que resulta artificioso). En el [apartado 9.3](#) del Anexo 1 se explica más detalladamente qué es Markdown, así como también su sintaxis.

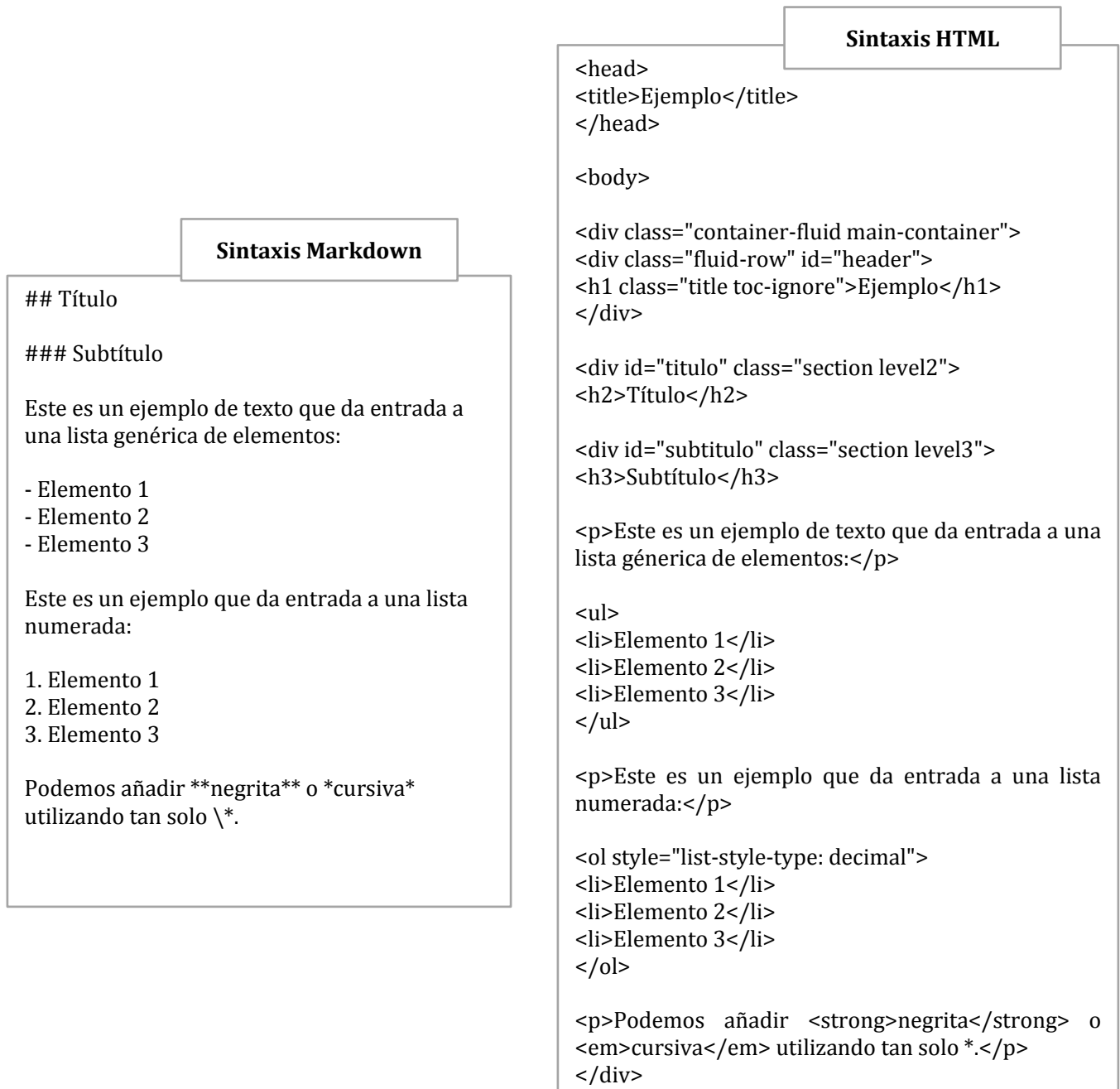
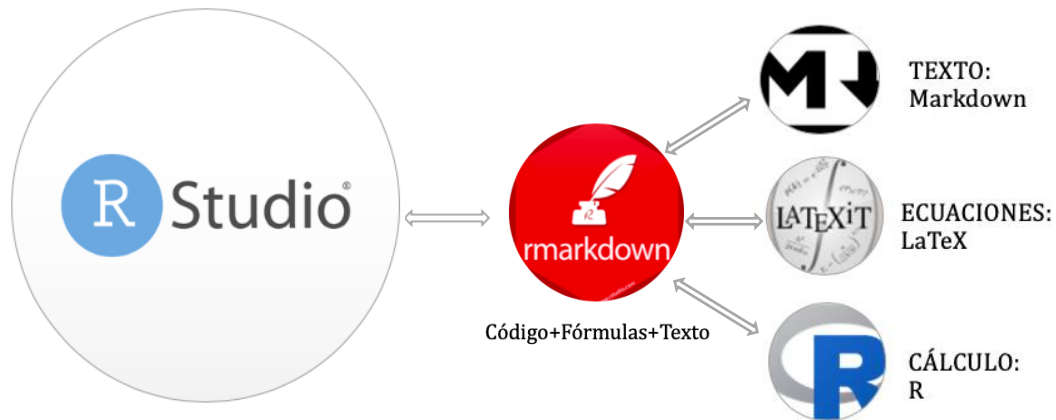


Figura 2.6 Comparación sintaxis Markdown (Izq.) con sintaxis HTML (Dcha)
Fuente: <https://markdown.es/>

Por tanto, para poder utilizar el lenguaje Markdown, con la sintaxis de LaTeX para las ecuaciones, necesitamos un paquete que permita la integración del código escrito en R con el texto plano y el resto de los elementos. Para ello, se trabajará con el paquete *RMarkdown* de R (ver Figura 2.7). Este paquete, permite generar documentos que integran texto, imágenes y código de R, más los resultados de la ejecución del código. Primero se crea un nuevo documento en formato. Rmd, escrito con lenguaje Markdown y con bloques de código de R. Se compila y se crean archivos HTML (convertibles a *.pdf*, o *.docx*) con el texto y los resultados del programa R integrados. En el [apartado 9.4](#) del Anexo 1 se especifica como introducir estos bloques de código juntamente con las diferentes opciones que presentan.

Finalmente, ¿qué software nos integra todo en uno? El gigante RStudio:

- Editor de archivos de texto
- Consola para el cálculo utilizando R
- Visor de páginas web
- Permite compilar archivos RMD
- Gran cantidad de funcionalidades



*Figura 2.7 RStudio, el programa que integra todo
Fuente: Elaboración propia. Imágenes: Wikipedia*

En el [apartado 9.5](#) del Anexo 1 se encuentra una explicación de RStudio, junto con el proceso de instalación y cómo utilizarlo.

3 CURRÍCULO: COMPETENCIAS CLAVE

3.1 Visión teórica

Desde la Unión Europea se recomienda a los estados miembros desarrollar un plan de integración curricular de las competencias clave para lograr que los ciudadanos alcancen un pleno desarrollo personal, social y profesional. Asimismo, a causa de la globalización, cada individuo deberá estar dotado de una amplia gama de competencias para poder desenvolverse y adaptarse con flexibilidad a las diferentes situaciones que se les planteen (Consejo de la Unión Europea, 2018).

Con todo ello, estamos ante un cambio notable en lo que al proceso de enseñanza-aprendizaje respecta, ya que este deberá basarse en el desarrollo competencial, y por tanto, deberá caracterizarse por la transversalidad, dinamismo y carácter integral (Orden ECD/65/2015, de 21 de enero). Además, la adquisición de las competencias no sucede de forma instantánea, sino de forma progresiva y con su continuo uso en la vida cotidiana, tanto dentro del contexto educativo como fuera, es decir, se trata de un conocimiento en la práctica.

Así pues, de acuerdo con lo regulado en la Orden ECD/65/2015, las competencias clave se definen como “aquellas que todas las personas precisan para su realización y desarrollo personal, así como para la ciudadanía activa, la inclusión social y el empleo”. En consecuencia, en nuestro sistema educativo se detallan y establecen las ocho competencias clave necesarias, y son:

- Comunicación Lingüística (CCL)
- Competencia Matemática y competencias básicas en Ciencia y Tecnología (CMCT)
- Competencia Digital (CD)
- Aprender a aprender (CPAA)
- Competencias Sociales y Cívicas (CSC)
- Sentido de la Iniciativa y Espíritu Emprendedor (SIE)
- Conciencia y Expresiones Culturales (CEC)

Todas ellas deben estar integradas en las propuestas curriculares de las diferentes materias que se impartan en los centros educativos. No son pues, un modelo pedagógico, sino más bien una guía para la educación, ya que como señala Tobón (2006), se focalizan en aspectos básicos como:

"1) La integración de los conocimientos, los procesos cognitivos, las destrezas, las habilidades, los valores y las actitudes en el desempeño ante actividades y problemas; 2) la construcción de los programas de formación acorde con los requerimientos disciplinares, investigativos, profesionales, sociales, ambientales y laborales del contexto; y 3) la orientación de la

educación por medio de estándares e indicadores de calidad en todos sus procesos"(Tobón, ,2006)

De este modo, la labor del profesorado consiste en adaptar metodologías, recursos y generar escenarios didácticos que favorezcan la adquisición de las diferentes destrezas que explicitan las competencias clave anteriores.

3.2 Competencias clave en las matemáticas

Resulta evidente que las matemáticas, o más concretamente, la asignatura de matemáticas, deberá contribuir al desarrollo y adquisición de las competencias clave descritas anteriormente. Así, según lo establecido en el Decreto 87/2015, de 5 de junio, del Consell, las matemáticas favorecen la adquisición y desarrollo de las diferentes competencias del siguiente modo:

- Competencia Lingüística (CCL), ya que se adquiere vocabulario nuevo y específico de las matemáticas y también se practica la comunicación oral y escrita. De esta forma, los alumnos deben aprender a transmitir sus ideas, ya sea por canal escrito u oral, con un lenguaje técnico adecuado.
- Competencia Matemática y competencias básicas en Ciencia y Tecnología (CMCT), ya que el alumnado llega a establecer en las diferentes tareas matemáticas o científicas una relación entre lo conceptual y lo procedimental. Además, favorecen en gran medida el desarrollo del pensamiento lógico-deductivo.
- Competencia Digital (CD), que se adquiere mediante el uso de herramientas informáticas y software específico como recurso didáctico y también para facilitar la resolución de problemas y el enfoque a futuras investigaciones.
- Competencia de Aprender a Aprender (CPAA), pues las matemáticas favorecen el análisis, la comprobación, el contraste o la extracción de conclusiones a partir de las informaciones que se reciben, para poder evaluar la fiabilidad de las fuentes y extraer conclusiones coherentes. Además, también contribuyen a la autonomía, perseverancia, reflexión crítica y sistematización.
- Competencia Social y Cívica (CSC) en tanto que el bloque referido al análisis de información, facilita que el alumnado analice los datos necesarios para tomar decisiones por lo que respecta a su participación social y como consumidor responsable de encuestas, sondeos, reportajes, gráficas... Además, el trabajo en grupo favorece el tener en cuenta las decisiones de los compañeros/as y a valorar de forma crítica su validez, escogiendo como solución aquella que más se adapte a los valores del sistema democrático y al bienestar de la sociedad.
- Sentido de la Iniciativa y Espíritu Emprendedor (SIE), implícito en la metodología de las matemáticas, ya que el alumnado puede llegar a sentirse

capaz de aprender, con autonomía y con esfuerzo propio; aumentando también el compromiso personal. En este sentido, es el alumno quien lleva la iniciativa y el que obtiene resultados a partir de sus decisiones y sus esfuerzos.

- Conciencia y Expresiones Culturales (CEC), pues las matemáticas son una parte fundamental de la cultura, son la base de muchas de las ciencias y el lenguaje universal. Además, alcanzan su expresión sobretodo en áreas como la geometría, pasando incluso por la música.

Por tanto, las bases del currículo referidas a las competencias clave emanan de las instituciones políticas, tanto a nivel europeo, nacional y autonómico. No obstante, en la práctica son los docentes los que llevan a cabo la implementación de las diferentes tareas para conseguir este desarrollo competencial, ya que, el comunicador principal en el aula es el docente, y de este modo el encargado de que el proceso de enseñanza-aprendizaje tenga como base los criterios establecidos en la normativa.

4 OBJETIVOS

Con el presente trabajo, y siguiendo la línea de las competencias clave descritas en el apartado anterior, se pretende abordar una serie de objetivos fundamentales sobre los que se construye esta propuesta de trabajo:

- Adaptar el ámbito educativo al uso de las nuevas tecnologías.
- Beneficiar al alumnado y profesorado con las ventajas 'tanto presentes como futuras' que supone el uso de las herramientas informáticas propuestas.
- Desarrollar las competencias clave:
 - CCL, ya que implica la utilización correcta de léxico matemático y la redacción coherente del documento; así como también la comunicación eficaz de los resultados que se obtengan.
 - CMCT, pues se trabaja principalmente con conceptos matemáticos.
 - CD, debido al uso de las herramientas informáticas y de software específico para las matemáticas.
 - CPAA y SIE, ya que tanto el alumnado como el profesorado pueden ampliar los conocimientos básicos de forma autónoma y eficaz, adaptándoselos a su nivel.
 - CSC, pues favorece que el análisis de situaciones relacionadas con el entorno se plasme de forma que el trasvase de conocimientos sea inteligible para la sociedad.
- Impulsar la correcta utilización de software científico.
- Concienciar de la importancia de la redacción científica adecuada.
- Guiar al alumnado y profesorado en la redacción de textos científicos.
- Favorecer el desarrollo del aprendizaje constructivo.
- Convencer de la eficacia y la funcionalidad de RStudio.
- Iniciar al alumnado y al profesorado en el entorno de la programación.

5 DESTINATARIOS

A lo largo de los años, han sido muchos los modelos psicológicos propuestos para describir el proceso de enseñanza y aprendizaje. Sin embargo, como apuntan Martín y Navarro (2011, p.115), “no existe ninguna teoría que abarque todo el potencial del aprendizaje humano”.

Uno de los modelos que ha ganado auge en los últimos años es el modelo constructivista, un modelo en el cual el alumno es partícipe de su proceso de enseñanza-aprendizaje, es quien construye su propio conocimiento. El alumno, a partir de un ejemplo concreto debe ser capaz de llegar a la generalización, es decir, a la abstracción; todo ello a partir de conocimientos previos que tiene en su memoria.

En esta teoría el papel del docente es el de guía para el alumnado, porque tal y como señalan Ausubel y Bruner, dos de los autores de las teorías constructivistas más relevantes: “los alumnos aprenden cuando comprenden”. Ausubel defiende un aprendizaje significativo, es decir, “el que se produce cuando el alumno relaciona la nueva información con conocimientos previos ya almacenados en su estructura cognitiva” (Martín y Navarro, 2011, p.128), o más bien, un aprendizaje que se desarrolla de lo general a lo particular. Por el contrario, Bruner, defiende el aprendizaje por descubrimiento, de lo particular a lo general. Como indican Martín y Navarro (2011, p.127), para Bruner, que el alumno aprenda por descubrimiento no significa que descubra algo nuevo, sino que significa descubrir algo por sí mismo.

Por todo ello, por un lado, el foco del trabajo son los estudiantes, pues se pretende facilitarles una herramienta para que ellos mismos puedan incluso 'auto-aprender', favoreciendo la construcción de su propio conocimiento, tanto a nivel conceptual, por lo que a conceptos matemáticos respecta, como a nivel de los programas informáticos utilizados. Por otro lado, en este proceso, el alumno necesita una guía, por ello el profesorado también deberá conocer el funcionamiento del software. Además, también es una herramienta que les ayuda en la redacción de textos científicos, en su caso particular, exámenes, apuntes o incluso artículos de investigación.

5.1 Alumnado

Uno de los psicólogos más conocidos en el mundo de la educación es Piaget, pues dividió en varias etapas y sub-etapas el proceso del desarrollo cognitivo en niños y adolescentes. Así, como se señala en *Psicología para el profesorado de Educación Secundaria y Bachillerato* (Martín y Navarro, 2011, p.25), Piaget “concibe el desarrollo intelectual como un proceso continuo de organización y reorganización de estructuras en constante equilibrio que van desde la inteligencia sensomotora hasta la inteligencia formal”. Por tanto, es en esta última etapa que abarca desde los 12 años en adelante, donde comienza el desarrollo de las operaciones formales.

“Estas hacen posible la presencia del pensamiento científico. Es el momento en que el sujeto comienza a utilizar el pensamiento hipotético-deductivo, a teorizar y a manejar el pensamiento abstracto” (Martín y Navarro, 2011, p.26). Es decir, los adolescentes comienzan a no basar su conocimiento en experiencias reales, sino que son capaces de inventar situaciones verosímiles que les permitan razonar de forma lógica sobre ellas (Martín y Navarro, 2011, p.63). De este modo, los adolescentes al final de la etapa pueden pensar como un científico, plantear hipótesis, deducir y pensar con lógica y de forma coherente.

Es por esto por lo que se ha considerado pertinente dirigir el trabajo al alumnado de los cursos 4º de la ESO y posteriores, hasta 2º de Bachillerato: pues a partir de los 15-16 años, los adolescentes, en términos generales, han desarrollado ya su capacidad de abstracción. Por lo tanto, podrán ser capaces de entender los aspectos básicos relacionados con la programación de RStudio, así como también la escritura utilizando la sintaxis Markdown o LaTeX. De esta forma, el alumnado dispondrá de las herramientas necesarias para poder crear textos científicos y también textos relacionados con otras áreas, pues al tratarse de un editor de texto plano, podemos no incluir fórmulas o cálculos. Además, por otro lado, los contenidos matemáticos que se corresponden con estos cursos permiten sacar más provecho al software que se utilizará.

5.2 Profesorado

Con este trabajo no solo se pretende enseñar al alumnado a redactar de una forma más cómoda textos que incluyan cálculos, código, gráficas y fórmulas, sino que también pretende servir de instrumento para el cuerpo docente como alternativa más eficiente a otras herramientas de procesamiento de textos.

Al igual que el alumnado, el profesorado, más concretamente el de las áreas científicas, necesita de herramientas que le faciliten la escritura de textos científicos, como por ejemplo exámenes o apuntes. A través de esta guía, el profesorado podrá aprender a utilizar el software de cálculo RStudio y el lenguaje Markdown y LaTeX, y escribir apuntes para sus estudiantes, realizar los exámenes (pues los exámenes de matemáticas suelen incluir fórmulas y gráficos) e incluso, si decide ampliar su conocimiento acerca de estas herramientas, hacer presentaciones que incluyan fórmulas, gráficos o código. Además, el profesorado debe de estar formado adecuadamente para poder enseñar a sus alumnos todo lo relativo al software y lenguajes comentados, y para poder adaptarse de forma gradual al cambio que propician las nuevas tecnologías en el contexto educativo.

6 REDACCIÓN DE TEXTOS CIENTÍFICOS: EJEMPLOS

6.1 Alumnado

Como se comenta en apartados anteriores, la redacción de textos científicos tiene especial importancia en el alumnado, no solo porque se inician en la escritura específica de un área, sino porque también desarrollan la capacidad de razonar, pensar y deducir.

Así pues, pueden proponerse gran cantidad de ejercicios o trabajos para que el estudiantado pueda poner en práctica los conocimientos relativos a la escritura científica y al software que se utilizará. De este modo, el alumnado redactará una especie de informe respuesta resolviendo los ejercicios planteados, incluyendo las fórmulas necesarias, el código y su resultado, y gráficos si así se especifica. Además, también se incluye la posibilidad de realizar presentaciones utilizando el mismo programa informático.

Por otro lado, en el [Anexo 2](#), aparece una explicación detallada de la aplicación del lenguaje R a los contenidos curriculares de cada curso, separados en cuatro bloques: Números y álgebra, Análisis, Geometría y Estadística y probabilidad. Por tanto, puede servir de guía para el alumnado en el proceso de adquisición de los conceptos matemáticos que se exponen; y al profesor puede proporcionarle alternativas para abordar la visualización o resolución de tareas.

A continuación, se muestra un ejemplo de un informe que podría escribir un alumno respondiendo a los ejercicios planteados por el profesor; junto con una presentación.

Ejemplo de ejercicios

En primer lugar, se muestra el principio de la redacción que podría elaborar el alumnado en respuesta a unos ejercicios utilizando la sintaxis de *RMarkdown* (ver Figura 6.1), es decir, combinando lenguaje de escritura Markdown y código R, desde la consola de RStudio (el código completo se encuentra en el [Anexo 3](#)). Seguidamente aparece la versión que resultaría al procesar el documento con la redacción inicial.

```

---
title: "Ejercicios 4º ESO-Matemáticas orientadas
a las enseñanzas académicas"
author: "Alumno"
date: "6/26/2019"
output: html_document
        pdf_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

### Ejercicio-Álgebra
***1. Encuentra las raíces de los siguientes polinomios y factoriza***.

+  $p_1(x) = x^3 + 2x^2 - x - 2$ ,
+  $p_2(x) = 2x^3 + x^2 - 18x - 9$ ,
+  $p_3(x) = x^2 - \frac{13}{4}x - 3$ .

**Solución:**
Para encontrar las raíces de un polinomio  $p(x)$ , se iguala a cero, es decir  $p(x)=0$  y se resuelve la ecuación.
Por tanto, una vez obtenidas las raíces, podemos escribir el polinomio con factores del tipo  $(x-a)$ , siendo  $a$  una raíz.

+  $p_1(x) = x^3 + 2x^2 - x - 2$ 

```{r}
library(polynom)
p1=polynomial(coef=c(-2,-1,2,1))
raices_p1=solve(p1)
```

Por tanto, las raíces de  $p_1(x)$  son : `r raices_p1`.

Y la factorización será:
 $p_1(x) = (x+2)(x+1)(x-1)$ 

***2. Comprueba gráficamente que las raíces encontradas, lo son.***

**Solución:**
Para comprobar gráficamente, dibujamos el polinomio, y donde corte con el eje X, debe de coincidir con el valor de las raíces:

```{r}
plot(p1)
abline(h=0,lty=2,col="red") #Marcar el eje X
```

Vemos como el polinomio corta al eje en los puntos  $x=-2$ ,  $x=-1$  y  $x=1$ . Por tanto, queda comprobado.

```

Aparece por defecto al crear el documento R Markdown

Figura 6.1 Inicio código RMarkdown
Fuente: Elaboración propia

Una muestra del resultado en formato PDF ² (adecuado para versión imprimible) se muestra en la Figura 6.2:

Ejercicios 4.º ESO- Matemáticas orientadas a las enseñanzas académicas

Alumno

6/26/2019

Ejercicio-Álgebra

1. Encuentra las raíces de los siguientes polinomios y factoriza:

- $p_1(x) = x^3 + 2x^2 - x - 2$,
- $p_2(x) = 2x^3 + x^2 - 18x - 9$,
- $p_3(x) = x^2 - \frac{13}{4}x - 3$.

Solución:

Para encontrar las raíces de un polinomio $p(x)$, se iguala a cero, es decir $p(x) = 0$ y se resuelve la ecuación. Por tanto, una vez obtenidas las raíces, podemos escribir el polinomio con factores del tipo $(x - a)$, siendo a una raíz.

- $p_1(x) = x^3 + 2x^2 - x - 2$

```
library(polynom)
p1=polynomial(coef=c(-2,-1,2,1))
raices_p1=solve(p1)
```

Por tanto, las raíces de $p_1(x)$ son : -2, -1, 1.

Y la factorización será:

$$p_1(x) = (x + 2)(x + 1)(x - 1)$$

2. Comprueba gráficamente que las raíces encontradas, lo son.

Solución: Para comprobar gráficamente, dibujamos el polinomio, y donde corte con el eje X, debe de coincidir con el valor de las raíces:

```
plot(p1)
abline(h=0,lty=2,col="red") #Marcar el eje X
```

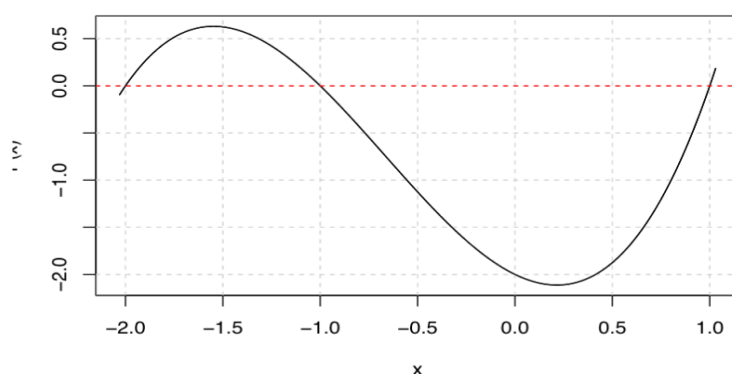


Figura 6.2 Formato PDF del archivo .Rmd
Fuente: Elaboración propia

Vemos como el polinomio corta al eje en los puntos $x = -2, x = -1$ y $x = 1$. Por tanto, queda comprobado.

² El documento .pdf completo se encuentra en el siguiente enlace:

<https://drive.google.com/file/d/1yIxyfadg18m010uzGSv7b8U7e17I1vCr/view?usp=sharing>

Y en formato HTML³, (adecuado para pantallas de todo tipo, por su capacidad de autoajuste) tal como se muestra en la Figura 6.3.

Ejercicios 4.º ESO- Matemáticas orientadas a las enseñanzas académicas

Alumno
6/26/2019

Ejercicio-Álgebra

1. Encuentra las raíces de los siguientes polinomios y factoriza:

- $p_1(x) = x^3 + 2x^2 - x - 2$,
- $p_2(x) = 2x^3 + x^2 - 18x - 9$,
- $p_3(x) = x^2 - \frac{13}{4}x - 3$.

Solución:

Para encontrar las raíces de un polinomio $p(x)$, se iguala a cero, es decir $p(x) = 0$ y se resuelve la ecuación. Por tanto, una vez obtenidas las raíces, podemos escribir el polinomio con factores del tipo $(x - a)$, siendo a una raíz.

- $p_1(x) = x^3 + 2x^2 - x - 2$

```
library(polynom)
p1=polynomial(coef=c(-2,-1,2,1))
raices_p1=solve(p1)
```

Por tanto, las raíces de $p_1(x)$ son : -2, -1, 1.

Y la factorización será:

$$p_1(x) = (x + 2)(x + 1)(x - 1)$$

2. Comprueba gráficamente que las raíces encontradas, lo son.

Solución: Para comprobar gráficamente, dibujamos el polinomio, y donde corte con el eje X, debe de coincidir con el valor de las raíces:

```
plot(p1)
abline(h=0,lty=2,col="red") #Marcar el eje X
```

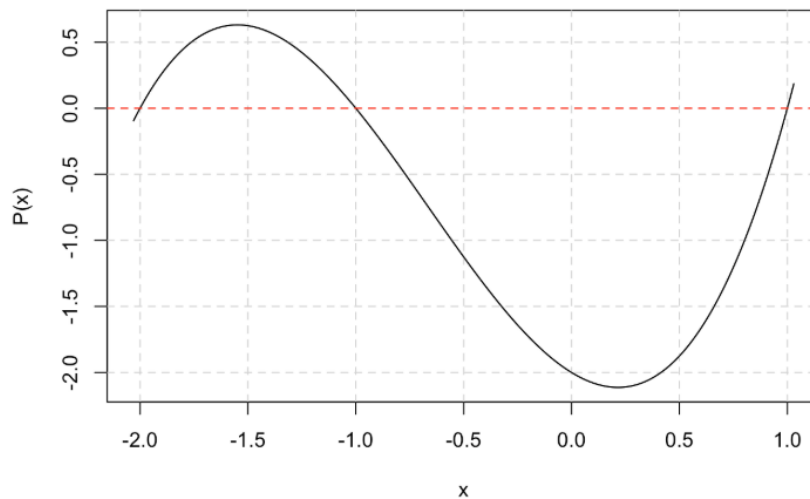


Figura 6.3 Formato HTML del documento .Rmd
Fuente: Elaboración propia

³ El documento .html completo se encuentra en el siguiente enlace:
<http://rpubs.com/mariaespinosa/509909>

En el Anexo 1 se puede consultar la escritura de fórmulas con LaTeX, la escritura de texto con Markdown, los comandos de R, y también cómo integrarlo todo usando el software RStudio.

Ejemplo de presentación

Crear presentaciones utilizando R Markdown también es posible. En el Anexo 1 (apartado 9.4.1), aparece la explicación más detallada. Si en lugar de realizar un informe, se les pidiera a los alumnos elaborar una presentación de los ejercicios para explicarlos a la clase, el código sería prácticamente el mismo (ver Figura 6.4), tan solo teniendo en cuenta que cada título que se añada, se corresponderá a una diapositiva diferente (véase [Anexo 3](#) para el código completo):

```

---
title: "Corrección de ejercicios"
author: "Alumno"
date: "6/27/2019"
output:
  ioslides_presentation: default
  beamer_presentation: default
---
```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = FALSE)
```

```

Aparece por defecto al crear el documento R Markdown

```

## Ejercicio de Álgebra

```

Nueva diapositiva con título

```

***1. Encuentra las raíces del siguiente polinomio y factoriza***:

+ $p_1(x)= x^3+2x^2-x-2$

Para encontrar las raíces de un polinomio $p(x)$, se iguala a cero, es decir, $p(x)=0$ y se resuelve la ecuación.
Por tanto, una vez obtenidas las raíces, podemos escribir el polinomio con factores del tipo $(x-a)$, siendo $a$ una raíz.

```{r echo=TRUE }
library(polynom)
p1=polynomial(coef=c(-2,-1,2,1))
raices_p1=solve(p1)
raices_p1
```

```

Nueva diapositiva sin título

```

***

```

Por tanto, las raíces de \$p_1(x)\$ son : `r raices_p1`.
Y la factorización será:

$$p_1(x) = (x+2)(x+1)(x-1)$$

Figura 6.4 Inicio código RMarkdown
Fuente: Elaboración propia

Una parte del resultado en formato *ioslides*⁴ se muestra en la Figura 6.5.

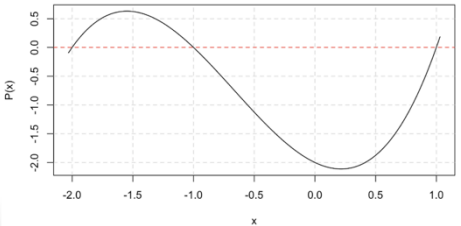
| | |
|--|--|
| <p style="text-align: center;">Corrección de ejercicios</p> <p>Alumno
6/27/2019</p> | <p style="text-align: center;">Ejercicio de Álgebra</p> <p>1. Encuentra las raíces del siguiente polinomio y factoriza:</p> <p>$p_1(x) = x^3 + 2x^2 - x - 2$</p> <p>Para encontrar las raíces de un polinomio $p(x)$, se iguala a cero, es decir, $p(x) = 0$ y se resuelve la ecuación. Por tanto, una vez obtenidas las raíces, podemos escribir el polinomio con factores del tipo $(x - a)$, siendo a una raíz.</p> <pre>library(polynom) p1<-polynomial(coef=c(-2,-1,2,1)) raices_p1<-solve(p1) raices_p1</pre> <pre>## [1] -2 -1 1</pre> <p style="text-align: right;">2/15</p> |
| <p>Por tanto, las raíces de $p_1(x)$ son : -2, -1, 1.</p> <p>Y la factorización será:</p> $p_1(x) = (x + 2)(x + 1)(x - 1)$ <p style="text-align: right;">3/15</p> | <p>2. Comprueba gráficamente que las raíces encontradas, lo son.</p> <p>Para comprobar gráficamente, dibujamos el polinomio, y donde corte con el eje X, debe de coincidir con el valor de las raíces:</p>  <p style="text-align: right;">4/15</p> |

Figura 6.5 Formato Ioslides (HTML) del código .Rmd
Fuente: Elaboración propia

Y también en formato *beamer*⁵, como se observa en la Figura 6.6.

| | |
|---|---|
| <p style="text-align: center;">Corrección de ejercicios</p> <p style="text-align: center;">Alumno</p> <p style="text-align: center;">6/27/2019</p> | <p style="text-align: center;">Ejercicio de Álgebra</p> <p>1. Encuentra las raíces del siguiente polinomio y factoriza:</p> <p>▶ $p_1(x) = x^3 + 2x^2 - x - 2$</p> <p>Para encontrar las raíces de un polinomio $p(x)$, se iguala a cero, es decir, $p(x) = 0$ y se resuelve la ecuación. Por tanto, una vez obtenidas las raíces, podemos escribir el polinomio con factores del tipo $(x - a)$, siendo a una raíz.</p> <pre>library(polynom) p1<-polynomial(coef=c(-2,-1,2,1)) raices_p1<-solve(p1) raices_p1</pre> <pre>## [1] -2 -1 1</pre> |
|---|---|

Figura 6.6 Formato Beamer (PDF) del archivo .Rmd
Fuente: Elaboración propia

⁴ Archivo Ioslides (HTML): <http://rpubs.com/mariaespinosa/509794>

⁵ Archivo Beamer (PDF): https://drive.google.com/file/d/1s3Z0UiR6mKXKypiqCWuwT0tuX_kmO6OE/view?usp=sharing

6.2 Profesorado

En este TFM proponemos fomentar la redacción de textos científicos entre los estudiantes. Sin embargo, los profesores suelen redactar sus apuntes, presentaciones y exámenes, y para ellos también proponemos esta alternativa a las herramientas más comúnmente utilizadas (Word, LaTeX puro, etc.), por las ventajas de unir en el mismo flujo de trabajo el texto (verbal y la simbología matemática) y los cálculos que pueda involucrar dicho texto. A continuación, aportamos ejemplos demostrativos.

Ejemplo de apuntes

Primeramente, se muestra el código en sintaxis *RMarkdown* de unos apuntes relativos al curso de 2º de Bachillerato de algunos conceptos de las áreas de álgebra y análisis (véase Figura 6.7). Así pues, aparece texto, junto con código de R, fórmulas matemáticas escritas en LaTeX, y los gráficos pertinentes, todo ello elaborado desde RStudio. Seguidamente, en las Figuras 6.7 y 6.8 aparece la versión que resultaría al procesar el documento con la redacción inicial en versión PDF y HTML, respectivamente. En el [Anexo 3](#) aparece el código en su totalidad.

```

---
title: "Apuntes 2º Bachillerato"
author: "Profesor"
date: "6/27/2019"
output:
  html_document: default
  pdf_document: default
---

```${r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

```

Aparece por defecto al crear el documento R Markdown

Álgebra - Interpretación geométrica de un sistema de ecuaciones

Sistemas de dos ecuaciones con dos incógnitas

Una vez explicada la forma matricial de un sistema, es importante recalcar la **interpretación geométrica de las ecuaciones** que forman nuestro sistema. Recordar, que en un sistema con dos ecuaciones y dos incógnitas, no son más que dos rectas, que pueden:

- + Ser **secantes**, es decir, cortarse en un punto. En este caso el sistema es Compatible Determinado (S.C.D)
- + Ser **coincidentes**. En este caso el sistema es Compatible Indeterminado (S.C.I), pues existen infinitas soluciones.
- + Ser **paralelas**, es decir, no cortarse en ningún punto. En este caso el sistema es Incompatible (S.I).

Ejemplo. Sea el sistema:

```


$$\begin{pmatrix} 1 & -2 \\ 2 & -1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -4 \\ 1 \end{pmatrix}$$


```

Si lo resolvemos, obtenemos por solución:

```

```${r echo=FALSE}
library(matlib)
A=matrix(c(1,-2,2,-1),nrow=2,ncol =2,byrow = TRUE)
b = c(-4,1)
solucion_sistema=Solve(A, b, fractions = TRUE)
```

```

Figura 6.7 Inicio código RMarkdown
Fuente: Elaboración propia

Una muestra del resultado en formato PDF ⁶ de uno de los apartados se muestra en la Figura 6.8.

Apuntes 2.º Bachillerato

Profesor

6/27/2019

Álgebra - Interpretación geométrica de un sistema de ecuaciones

Sistemas de dos ecuaciones con dos incógnitas

Una vez explicada la forma matricial de un sistema, es importante recalcar la **interpretación geométrica de las ecuaciones** que forman nuestro sistema. Recordar, que en un sistema con dos ecuaciones y dos incógnitas, no son más que dos rectas, que pueden:

- Ser **secantes**, es decir, cortarse en un punto. En este caso el sistema es Compatible Determinado (S.C.D)
- Ser **coincidentes**. En este caso el sistema es Compatible Indeterminado (S.C.I), pues existen infinitas soluciones.
- Ser **paralelas**, es decir, no cortarse en ningún punto. En este caso el sistema es Incompatible (S.I).

Ejemplo. Sea el sistema:

$$\begin{pmatrix} 1 & -2 \\ 2 & -1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -4 \\ 1 \end{pmatrix}$$

Si lo resolvemos, obtenemos por solución:

```
## x1 = 2
## x2 = 3
```

Figura 6.8 Formato PDF del archivo .Rmd
Fuente: Elaboración propia

De forma similar, la salida en formato HTML⁷ del código se expone en la Figura 6.9:

Apuntes 2.º Bachillerato

Profesor

6/27/2019

Álgebra - Interpretación geométrica de un sistema de ecuaciones

Sistemas de dos ecuaciones con dos incógnitas

Una vez explicada la forma matricial de un sistema, es importante recalcar la **interpretación geométrica de las ecuaciones** que forman nuestro sistema. Recordar, que en un sistema con dos ecuaciones y dos incógnitas, no son más que dos rectas, que pueden:

- Ser **secantes**, es decir, cortarse en un punto. En este caso el sistema es Compatible Determinado (S.C.D)
- Ser **coincidentes**. En este caso el sistema es Compatible Indeterminado (S.C.I), pues existen infinitas soluciones.
- Ser **paralelas**, es decir, no cortarse en ningún punto. En este caso el sistema es Incompatible (S.I).

Ejemplo. Sea el sistema:

$$\begin{pmatrix} 1 & -2 \\ 2 & -1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -4 \\ 1 \end{pmatrix}$$

Si lo resolvemos, obtenemos por solución:

```
## x1 = 2
## x2 = 3
```

Figura 6.9 Formato HTML del archivo .Rmd
Fuente: Elaboración propia

⁶ El documento .pdf completo se encuentra en el siguiente enlace:

<https://drive.google.com/file/d/1CArXKg7ktFEtc7ZLhtHaXiV2-9Lt2QjI/view?usp=sharing>

⁷ El documento .html completo se encuentra en el siguiente enlace:

<https://rpubs.com/mariaespinosa/509922>

Ejemplo de presentación

Como se indica en el apartado dedicado al alumnado, también es posible crear presentaciones desde RStudio, y exportarlas a formato *beamer*, *ioslides* e incluso *power point*.

Así pues, el docente puede considerar presentar los apuntes anteriores en modo presentación. Para ello, siguiendo lo explicado en el Anexo 1, se crea la presentación (ver Figura 6.10), utilizando de forma similar el texto en sintaxis R Markdown (véase [Anexo 3](#) para el código completo):

```
---
title: "Apuntes 2º Bachillerato"
author: "Profesor"
date: "6/27/2019"
output: ioslides_presentation
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = FALSE)
```
```

Aparece por defecto al crear el documento R Markdown

```
## Álgebra - Interpretación geométrica de un sistema de ecuaciones
### **Sistemas de dos ecuaciones con dos incógnitas**
```

Nueva diapositiva con título

Una vez explicada la forma matricial de un sistema, es importante recalcar la ****interpretación geométrica de las ecuaciones**** que forman nuestro sistema. Recordar, que en un sistema con dos ecuaciones y dos incógnitas, no son más que dos rectas, que pueden:

```
***
```

Nueva diapositiva sin título

- + Ser ****secantes****, es decir, cortarse en un punto. Por tanto, el sistema es Compatible Determinado (S.C.D)
- + Ser ****coincidentes****. Por tanto, el sistema es Compatible Indeterminado (S.C.I), pues existen infinitas soluciones.
- + Ser ****paralelas****, es decir, no cortarse en ningún punto. Por tanto, el sistema es Incompatible (S.I).

```
***
```

Nueva diapositiva sin título

****Ejemplo****. Sea el sistema:

$$\begin{pmatrix} 1 & -2 \\ 2 & -1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -4 \\ 1 \end{pmatrix}$$

Si lo resolvemos, obtenemos por solución:

```
```{r echo=FALSE}
library(matlib)
A=matrix(c(1,-2,2,-1),nrow=2,ncol=2,byrow=TRUE)
b=c(-4,1)
solucion_sistema=Solve(A,b,fractions=TRUE)
```
```

Figura 6.10 Inicio código RMarkdown para presentaciones
Fuente: Elaboración propia

Se puede apreciar la salida en formato *ioslides*⁸ en la Figura 6.11.

Apuntes 2º Bachillerato
Profesor
6/27/2019

Álgebra - Interpretación geométrica de un sistema de ecuaciones

Sistemas de dos ecuaciones con dos incógnitas
Una vez explicada la forma matricial de un sistema, es importante recalcar la **interpretación geométrica de las ecuaciones** que forman nuestro sistema. Recordar, que en un sistema con dos ecuaciones y dos incógnitas, no son más que dos rectas, que pueden:

- Ser **secantes**, es decir, cortarse en un punto. Por tanto, el sistema es Compatible Determinado (S.C.D)
- Ser **coincidentes**. Por tanto, el sistema es Compatible Indeterminado (S.C.I), pues existen infinitas soluciones.
- Ser **paralelas**, es decir, no cortarse en ningún punto. Por tanto, el sistema es Incompatible (S.I).

Ejemplo. Sea el sistema:

$$\begin{pmatrix} 1 & -2 \\ 2 & -1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -4 \\ 1 \end{pmatrix}$$

Si lo resolvemos, obtenemos por solución:

```
## x1 = 2
## x2 = 3
```

Si además, dibujamos las ecuaciones, observamos que son dos rectas secantes que se cortan en un punto:

Figura 6.11 Formato *ioslides*(HTML) del archivo *.Rmd*
Fuente: Elaboración propia

Y también en formato *beamer*⁹, en la Figura 6.12.

Sistemas de dos ecuaciones con dos incógnitas

Una vez explicada la forma matricial de un sistema, es importante recalcar la **interpretación geométrica de las ecuaciones** que forman nuestro sistema. Recordar, que en un sistema con dos ecuaciones y dos incógnitas, no son más que dos rectas, que pueden:

- ▶ Ser **secantes**, es decir, cortarse en un punto. Por tanto, el sistema es Compatible Determinado (S.C.D)
- ▶ Ser **coincidentes**. Por tanto, el sistema es Compatible Indeterminado (S.C.I), pues existen infinitas soluciones.
- ▶ Ser **paralelas**, es decir, no cortarse en ningún punto. Por tanto, el sistema es Incompatible (S.I).

Figura 6.12 Formato *Beamer* (PDF) del archivo *.Rmd*
Fuente: Elaboración propia

⁸ Archivo *ioslides* (HTML): <http://rpubs.com/mariaespinosa/509917>

⁹ Archivo *Beamer* (PDF): https://drive.google.com/file/d/1AlsXv1sH0g5eUciXaTvIT8_iCpyWk8h0/view?usp=sharing

Ejemplo de examen

Son muchas las veces que los profesores de matemáticas o de cualquier asignatura de carácter científico, deben elaborar exámenes, incluyendo fórmulas y gráficos. Puede resultar una tarea tediosa si no se tiene el dominio necesario de ciertos programas. Por eso, se recomienda hacer uso de esta guía, pues facilita en gran medida dicho cometido. De este modo, los profesores podrán redactar exámenes utilizando la sintaxis de *RMarkdown*, y a su vez resolverlo utilizando el RStudio.

No obstante, no solo puede servir para la mera redacción y resolución del examen, sino también para crear exámenes personalizados. Es decir, modelos de exámenes en los que tan solo cambiarán los números o parámetros que aparezcan en los enunciados. Así pues, podemos evitar la copia, ya que a través de un código que a continuación se mostrará, se pueden crear tantos modelos de exámenes como determinemos junto con su solución: puede ser un modelo diferente por alumno, o el número de modelos diferentes que se consideren apropiados.

A continuación, se muestra la redacción de un posible ejercicio de examen desglosada en diferentes partes para poder explicar la estructura que sigue el documento ([Anexo 3](#) para ver el documento íntegro): el enunciado del examen que se muestra en la Figura 6.13, y su solución en la Figura 6.14.

Seguidamente, se escribe la solución entre estas líneas de código:

```
`r c('<!--', '')[1+solucion]`  
---  
Solución: .....  
---  
`r c('-->', '')[1+solucion]`
```

Variable definida que determinará si incluir las soluciones o no en el documento

Pues si la solución es *FALSE*, esta aparece como comentario y no se muestra; mientras que, si es *TRUE*, aparece en la versión del documento.

```
`r c('<!--', '')[1+solucion]`  
---  
**Solución:**  
1. En este caso, para construir la tabla de frecuencias, se necesitan las frecuencias absolutas y las relativas:  
  
+ Frecuencias absolutas  
  
```{r}  
table(faltas_ortografia)
```  
  
+ Frecuencias relativas  
  
```{r}  
prop.table(table(faltas_ortografia))
```  
  
2. Para saber el porcentaje de `r falt1` faltas, debemos mirar la tabla de frecuencias relativas y multiplicar por 100:  
  
```{r}  
prop.table(table(faltas_ortografia))[falt1+1] *100
```  
  
3. Para calcular el número de alumnos, que ha hecho `r falt2` faltas o más, vamos a la tabla de frecuencias absolutas, y sumamos todos los valores que queden por encima de `r falt2`, este inclusive:  
  
```{r}  
table(faltas_ortografia)[falt2+1:max(faltas_ortografia)]
```  
  
4. La representación es un diagrama de barras:  
  
```{r}  
barplot(table(faltas_ortografia),main="Diag.de barras")
```  
---  
`r c('-->', '')[1+solucion]`
```

Figura 6.14 Redacción Solución del examen
Fuente: Elaboración propia

Para poder crear las diferentes versiones de exámenes a partir del documento anterior, se ejecutará el código R que se muestra en la Figura 6.15. Este *script* genera tantos archivos *.pdf* como modelos indiquemos en el contador, junto con su solución si así lo señalamos. Además, cada documento que se obtiene tiene por título: "Archivo-i", siendo *i* cada una de las versiones.

```

setwd("~/Desktop/MARÍA/TFM/Examen") #Para estar en la carpeta adecuada
archivo = "ExamenEstad-1920" → Plantilla genérica (sin sufijo .Rmd)

solucion = FALSE #TRUE para mostrar soluciones

set.seed(20190628) → Semilla para la repetibilidad del azar (se puede poner cualquier valor)

require(knitr) → Nº de modelos diferentes
for(iii in 1:5) {

  #Nombre archivo compilado según la versión
  archivo_i= paste(archivo, "-", iii, c('', 'solucion')[1+solucion],
                  ".pdf", sep='')

  #Compila el archivo Rmd desde código
  rmarkdown::render(input=paste(archivo, '.Rmd', sep=''),
                    output_format="pdf_document",
                    output_file=archivo_i) → Compila el documento .Rmd

}

```

Figura 6.15 Código de R que genera los diferentes modelos de exámenes
Fuente: Elaboración propia

Es importante señalar que el directorio de trabajo debe ser el mismo que la ubicación de los archivos *.R* y *.Rmd*. Destacar también que el nombre que se le pasa a "*archivo*", debe de ser el mismo que el nombre del documento *.Rmd*.

Por tanto, si ejecutamos el *script* anterior, este nos genera 5 modelos de exámenes diferentes¹⁰:

¹⁰ En el siguiente enlace se encuentra una carpeta con 5 modelos y sus soluciones:
<https://drive.google.com/drive/folders/1QyI IGilC69Ov6KioJlpN1sqV47jNXKV?usp=sharing>

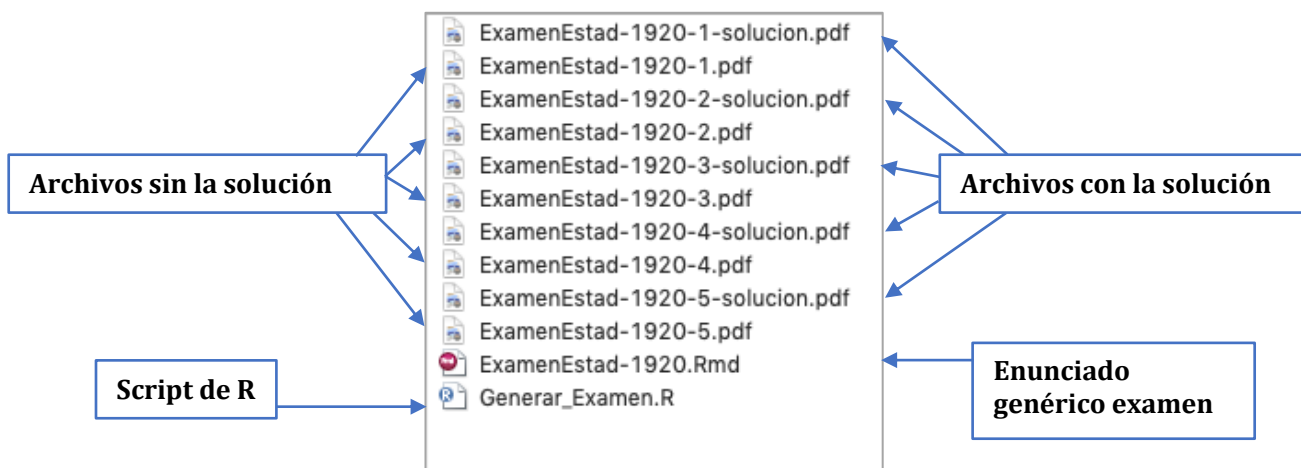


Figura 6.16 Modelos de exámenes generados y su solución
Fuente: Elaboración propia

Nota: En el caso de indicar "`solución=TRUE`", aparecerían los archivos con la solución de la forma en la que se muestra en la figura anterior.

Como se observa en la Figura 6.17 (Versión 1 del examen) y en la Figura 6.18 (Versión 2 del examen), los datos iniciales del problema son diferentes para cada una de las versiones, así como también los datos involucrados en las preguntas del ejercicio.

Examen 4º ESO - Matemáticas orientadas a las enseñanzas aplicadas

Estadística Descriptiva

IES Gilabert de Centelles

Versión 1: Nombre y Apellidos _____

Problema 1 (2 pt)

La profesora de castellano ha contabilizado las faltas de sus alumnos en los exámenes, y ha obtenido los siguientes resultados:

0, 5, 6, 5, 3, 1, 2, 6, 1, 6, 3, 8, 4, 1, 3, 8, 0, 0, 0, 4, 5, 3, 5, 3, 7

A partir de estos:

1. Construye una tabla de frecuencias para la variable **faltas de ortografía**.
2. ¿Qué porcentaje de alumnos ha hecho 2 faltas? Indica qué frecuencia se utiliza para el cálculo y **detalla los cálculos**.
3. ¿Cuántos alumnos han hecho 7 faltas o más? Indica qué frecuencia se utiliza para el cálculo y **detalla los cálculos**.
4. Representa con el gráfico adecuado. ¿Qué nombre recibe este gráfico?

Solución:

1. En este caso, para construir la tabla de frecuencias, se necesitan las frecuencias absolutas y las relativas:

- Frecuencias absolutas

```
## faltas_ortografia
## 0 1 2 3 4 5 6 7 8
## 4 3 1 5 2 4 3 1 2
```

- Frecuencias relativas

```
## faltas_ortografia
## 0 1 2 3 4 5 6 7 8
## 0.16 0.12 0.04 0.20 0.08 0.16 0.12 0.04 0.08
```

2. Para saber el porcentaje de 2 faltas, debemos mirar la tabla de frecuencias relativas y multiplicar por 100:

*Figura 6.17 Versión 2 del examen generada con el código anterior, junto con solución.
Fuente: Elaboración propia*

Versión 2: Nombre y Apellidos_____

Problema 1 (2 pt)

La profesora de castellano ha contabilizado las faltas de sus alumnos en los exámenes, y ha obtenido los siguientes resultados:

3, 8, 2, 2, 0, 5, 6, 0, 8, 8, 7, 1, 7, 2, 6, 6, 7, 2, 4, 4, 5, 5, 8, 4, 3

A partir de estos:

1. Construye una tabla de frecuencias para la variable **faltas de ortografía**.
2. ¿Qué porcentaje de alumnos ha hecho 6 faltas? Indica qué frecuencia se utiliza para el cálculo y **detalla los cálculos**.
3. ¿Cuántos alumnos han hecho 0 faltas o más? Indica qué frecuencia se utiliza para el cálculo y **detalla los cálculos**.
4. Representa con el gráfico adecuado. ¿Qué nombre recibe este gráfico?

Solución:

1. En este caso, para construir la tabla de frecuencias, se necesitan las frecuencias absolutas y las relativas:

- Frecuencias absolutas

```
## faltas_ortografia
## 0 1 2 3 4 5 6 7 8
## 2 1 4 2 3 3 3 3 4
```

- Frecuencias relativas

```
## faltas_ortografia
## 0 1 2 3 4 5 6 7 8
## 0.08 0.04 0.16 0.08 0.12 0.12 0.12 0.12 0.16
```

2. Para saber el porcentaje de 6 faltas, debemos mirar la tabla de frecuencias relativas y multiplicar por 100:

*Figura 6.18 Versión 2 del examen generada con el código anterior, junto con solución.
Fuente: Elaboración propia*

7 CONCLUSIÓN

La llegada de las nuevas tecnologías ha supuesto un antes y un después en todos los ámbitos en los que se han incorporado. En el ámbito educativo, las nuevas tecnologías llegan para quedarse, propiciando cambios significativos en las metodologías que se implementan día a día en las aulas. No obstante, no solo basta con conocerlas y saber manejarlas, sino que es importante que el profesorado adquiera las habilidades necesarias para introducirlas de forma correcta y sobretodo, eficiente.

En el caso concreto de las matemáticas, existe gran cantidad de programas informáticos dedicados a trabajar conceptos matemáticos, pero, en este trabajo, la base ha sido RStudio, un programa que comenzó como una interfaz gráfica de usuario para el programa de estadística R, aportando algunas comodidades, y que con el paso de los años, y el desarrollo de algunas librerías, como *RMarkdown*, se ha convertido en una seria alternativa para la redacción de textos que incluyen el análisis de datos con R, simplificando el flujo de trabajo y dando gran versatilidad.

Sin embargo, el objetivo del presente trabajo va más allá de la simple introducción de software matemático: el eje central recae en la reflexión sobre una eficiente redacción de textos de carácter científico para el profesorado y para el estudiantado. Así pues, por un lado, se trata de una guía para el alumnado, ya que propicia la introducción a la programación, la explicación de conceptos de forma innovadora a través del software utilizado y sobretodo, ayuda y favorece la redacción de textos que incluyen fórmulas, comandos o pequeños bloques de programación con sus salidas e imágenes.

Por otro lado, considero que también puede resultar un material útil para el profesorado, pues es una alternativa muy interesante para la escritura de apuntes y exámenes, y como hemos visto, también la elaboración de exámenes personalizados. Adicionalmente, se han presentado y ejemplificado diversas funciones del lenguaje R que permiten nuevas formas de abordar los contenidos curriculares de matemáticas de 4º de la ESO hasta 2º de Bachillerato. Todo este material puede servir como guía para el profesorado interesado.

Aun así, pese a poder trabajar la gran mayoría de los contenidos curriculares de la asignatura, en el bloque de geometría, el software utilizado presenta un punto débil, pues no se han desarrollado paquetes para poder sustituir la facilidad de visualización geométrica que presentan otros programas, como por ejemplo, Geogebra.

A través de la sintaxis sencilla que presenta el lenguaje Markdown, la programación a través del lenguaje R, y la escritura de fórmulas mediante LaTeX, todo ello englobado en *RMarkdown* e incorporado al programa informático RStudio; se consigue que tanto alumnado como profesorado puedan elaborar textos científicos cargados de notación matemática, con una calidad tipográfica profesional y sin un coste de aprendizaje excesivo. Todo ello, teniendo en cuenta la importancia que recae en la visualización del documento, pues a parte de los formatos más

conocidos como los PDF o los Docx, los archivos escritos a través de RMarkdown, se visualizan fácilmente en formato HTML, y este se caracteriza por su gran adaptabilidad a todo tipo de pantallas gracias a su capacidad de autoajuste.

No solo se hace hincapié en la importancia de la utilización de herramientas informáticas, sino también se recalca la gran importancia de la redacción científica, pensando en el futuro profesional del alumnado.

A pesar de todo lo descrito, es importante remarcar que el trabajo está desarrollado a nivel teórico y constituye una propuesta para redactar textos científicos que no se ha llevado aún a la práctica. Espero en un futuro próximo implementarlo en las aulas con total normalidad, pues es una herramienta de enseñanza de las matemáticas que va totalmente de la mano con las nuevas tecnologías. Además, resulta fundamental que el profesorado se sumerja en el entorno de la programación y el software matemático, y de este modo, incitar al alumnado a pensar, a imaginar. Que puedan entender las matemáticas como un proceso creativo y no como un proceso mecánico.

Desde que empecé el grado de matemáticas ya sabía que quería convertirme en docente, no solo para enseñar los conocimientos teóricos, si no para poder transmitir a mis alumnos la fascinación que pueden despertar y el maravilloso mundo que se esconde tras las matemáticas.

*-La vida es buena por solo dos cosas, descubrir y enseñar las matemáticas.
Simeon Poisson.*

8 REFERENCIAS

Adobe. (2019). *PDF: Tres letras que cambiaron el mundo*. Obtenido de Adobe Document Cloud:

<https://acrobat.adobe.com/es/es/acrobat/about-adobe-pdf.html>

Barriuso, J., Gómez, V., Haro, J., & Parreño, F. (2013). Introducción a la Estadística con R. *SUMA*, 72, 17-30.

Bruner Jerome (1988) *Realidad mental y mundos posibles: los actos de la imaginación que dan sentido a la experiencia*. Barcelona.Gedisa.

Caraballo, H., & Zulema González, C. Z. (2009). Herramientas para la enseñanza y el aprendizaje de la matemática: Software libre. Universidad Nacional de La Plata. Facultad de Humanidades y Ciencias de la Educación. Departamento de Ciencias Exactas y Naturales.

Carles, J. (Abril de 2019). *Aprender Markdown y Multimarkdown de forma sencilla y rápida*. Obtenido de <https://geekland.eu/aprender-markdown-en-minutos/>

Checa, S. (Abril de 2018). *¿Cuál es la diferencia entre un archivo .DOC y .DOCX en Microsoft Word?* Obtenido de Locura informática:

<https://www.locurainformaticadigital.com/2018/04/30/diferencia-archivo-doc-y-docx-word/>

Conesa, D. (s.f.). *Computación y programación en R: Tema 1*. Obtenido de <https://www.uv.es/conesa/Cursor/material/handout-sesion1.pdf>

Consejo de la Unión Europea (2018): «Recomendación del Consejo de 22 de mayo de 2018, relativa a las competencias clave para el aprendizaje permanente». Diario Oficial de la Unión Europea (4 junio).

Cristóbal, J. (s.f.). *Sintaxis Markdown*. Obtenido de Markdown.es: <https://markdown.es/sintaxis-markdown/>

Del Moral, M. E. (1997): “La actualización docente en nuevas tecnologías ante las exigencias de su integración en los diseños curriculares”. *Aula Abierta*, volumen 70, pp. 77-93.

Del Moral Pérez, M. E., & Martínez, L.V. (2010). Formación del profesor 2.0: desarrollo de competencias tecnológicas para la escuela 2.0. *Magister: Revista miscelánea de investigación*, (23), 59-69

DOC, Microsoft Word Binary File Format (.doc). (s.f.). Obtenido de Online-Convert: <https://www.online-convert.com/es/formato-de-archivo/doc>

Gardey, J. P. (2009). *Definición de PDF*. Obtenido de Definición de.: <https://definicion.de/pdf/>

Gutiérrez, E. (2009). Leer digital la lectura en el entorno de las nuevas Tecnología de la información y la comunicación. *Signo y pensamiento*, 28(54), 144-163.

Febrero, M., Galeano, P., González, J., & Pateiro, B. (2008). Prácticas de estadística en R. *España, Universidad de Santiago de Compostela*.

González Pérez, A., & Pablos Pons, J. D. (2015). Factores que dificultan la integración de las TIC en las aulas. *Revista de Investigación Educativa*, 33 (2), 401-417.

JLPM. (s.f.). *Principales lenguajes de programación web, ventajas y desventajas*.

Obtenido de Registro de dominios:

<https://www.registrodominiosinternet.es/2013/08/lenguajes-programacion-web-ventajas.html>

Martín, C., & Navarro, J. I. (2011). Psicología para el profesorado de Educación Secundaria y Bachillerato. *Madrid, España: Editorial Pirámide*.

Ministerio de Educación y Formación Profesional-Gobierno de España. (s.f.).

Competencia Digital. Obtenido de LOMCE:

<https://www.educacionyfp.gob.es/educacion/mc/lomce/el-curriculo/curriculo-primaria-eso-bachillerato/competencias-clave/competencias-clave/digital.html>

Ministerio de la presidencia, relaciones con las cortes e igualdad. (s.f.). *¿Qué es el formato ePUB?* Obtenido de Agencia Estatal Boletín Oficial del Estado:

https://www.boe.es/diario_boe/preguntas_frecuentes/formato_epub.php

Pérez, M. E. (1997). La actualización docente en nuevas tecnologías ante las exigencias de su integración en los diseños curriculares. *Aula Abierta*(70), 77-93.

Platas, A. B., & García, R. P. Ñ. (2017). TEI down: Uso de Markdown extendido para el marcado automático de documentos TEI. *Revista de humanidades digitales*, 1, 57-75.

Porto, J. P., & Gardey, A. (2009). *Definición de PDF*. Obtenido de Definición de.:

<https://definicion.de/pdf/>

¿Qué es HTML? (s.f.). Obtenido de Programar es el futuro:

<https://codigofacilito.com/articulos/que-es-html>

RStudio. (2016-2018). *R Markdown*. Obtenido de RStudio:

<https://rmarkdown.rstudio.com/lesson-1.html>

RStudio. (s.f.). *RStudio*. Obtenido de RStudio: <https://www.rstudio.com/>

RStudio. (s.f.). *RStudio Features*. Obtenido de RStudio:

<https://www.rstudio.com/products/rstudio>

Rosero, J. R. (2018). Impacto del uso de las TIC como herramientas para el aprendizaje de la matemática de los estudiantes de educación media. *Cátedra*, 1(1), 70-91.

Santana, A., & Hernandez, C. N. (2016). *Introducción a R*. Obtenido de Introducción a R:
<http://www.dma.ulpgc.es/profesores/personal/stat/cursoR4ULPGC/index.html>

Santana, A., & Hernandez, C. N. (2016). *Creación de documentos con R Markdown*. Obtenido de Introducción a R:
<http://www.dma.ulpgc.es/profesores/personal/stat/cursoR4ULPGC/20-Rmarkdown.html>

Santana, A., & Hernandez, C. N. (2016). *Librerías en R*. Obtenido de Introducción a R:
<http://www.dma.ulpgc.es/profesores/personal/stat/cursoR4ULPGC/5-librerias.html>

Simpkin, S. (2017). *Introducción a Markdown*, traducido por Víctor Gayol. Obtenido de The Programming Historian:
<https://programminghistorian.org/es/lecciones/introduccion-a-markdown>

TerraXML. (s.f.). *Is HTML or PDF Better for Technical Documentation?* Obtenido de:
<http://terraxml.com/is-html-or-pdf-better-for-technical-documentation/>

The R Foundation. (s.f.). *The R Project for Statistical Computing*. Obtenido de R:
<https://www.r-project.org/>

Tobón, S. (2006). Aspectos básicos de la formación basada en competencias. *Talca: Proyecto Mesesup*, 1, 1-15.

Zubcoff, J. (2017). Curso Herramientas básicas para Data Science. Módulo 5. Informes automáticos con RMarkdown.

VALENCIA, A. G. (2006). E-Reading, la nueva revolución de la lectura: del texto impreso al ciber-texto.

Wikipedia. (s.f.). Calibre. Obtenido de Wikipedia:
[https://es.wikipedia.org/wiki/Calibre_\(software\)](https://es.wikipedia.org/wiki/Calibre_(software))

Wikipedia. (s.f.). EPUB. Obtenido de Wikipedia:
<https://es.wikipedia.org/wiki/EPUB>

Wikipedia. (s.f.). HTML. Obtenido de Wikipedia:
<https://es.wikipedia.org/wiki/HTML>

Wikipedia. (s.f.). *LaTeX*. Obtenido de Wikipedia:
<https://es.wikipedia.org/wiki/LaTeX>

Wikipedia. (s.f.). *Markdown*. Obtenido de Wikipedia:
<https://es.wikipedia.org/wiki/Markdown>

Wikipedia. (s.f.). PDF. Obtenido de Wikipedia:
<https://es.wikipedia.org/wiki/PDF>

Wikipedia. (s.f.). WYSIWYG. Obtenido de Wikipedia:
<https://es.wikipedia.org/wiki/WYSIWYG>

9 ANEXO 1: SOFTWARE

9.1 LaTeX

LaTeX es un “sistema de composición de textos, orientado a la creación de documentos escritos que presenten una alta calidad tipográfica” (Wikipedia, s.f.). Suele usarse para la redacción de libros científicos o artículos que incluyen gran cantidad de expresiones matemáticas. Aprender a utilizarlo es un trabajo costoso, pero una vez se aprende es difícil dejar de manejarlo.

No obstante, para lo que nos atañe, tan solo se mostrarán los comandos necesarios para poder introducir fórmulas matemáticas en el documento .Rmd. Todas las fórmulas o símbolos LaTeX que se utilicen deben incluirse entre dos símbolos de dólar: \$fórmula\$, o bien entre cuatro para que la fórmula aparezca en el siguiente párrafo \$\$fórmula\$\$.

Recogemos las principales funciones en la siguiente tabla:

| | LaTeX | Resultado |
|---------------|---|--|
| Superíndice | <code>x^{2-x}</code> | x^{2-x} |
| Subíndice | <code>x_{i+1}</code> | x_{i+1} |
| Fracción | <code>\frac{2}{x-1}</code> | $\frac{2}{x-1}$ |
| Matriz | <code>\begin{pmatrix} a & b \\ c & d \end{pmatrix}</code> | $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ |
| Determinante | <code>\begin{vmatrix} a & b \\ c & d \end{vmatrix}</code> | $\begin{vmatrix} a & b \\ c & d \end{vmatrix}$ |
| Raíz cuadrada | <code>\sqrt{5}</code> | $\sqrt{5}$ |
| Raíz n-ésima | <code>\sqrt[n]{9}</code> | $\sqrt[n]{9}$ |
| Sumatorio | <code>\sum_{i=1}^n (x_i^2)</code> | $\sum_{i=1}^n x_i^2$ |
| Integral | <code>\int_a^b f(x) dx</code> | $\int_a^b f(x) dx$ |
| Límite | <code>\lim_{n \to \infty} (1 + \frac{1}{n})^n</code> | $\lim_{n \rightarrow \infty} \left(1 + \frac{1}{n}\right)^n$ |

Símbolos y letras

| | LaTeX | Resultado |
|----------------------|--|---|
| Existe | <code>\exists</code> | \exists |
| No existe | <code>\nexists</code> | \nexists |
| Pertenece | <code>\in</code> | \in |
| No pertenece | <code>\notin</code> | \notin |
| Para todo | <code>\forall</code> | \forall |
| Menor o igual | <code>\leq</code> | \leq |
| Mayor o igual | <code>\geq</code> | \geq |
| Más-menos | <code>\pm</code> | \pm |
| Punto
multiplicar | <code>\cdot</code> | \cdot |
| Paréntesis | <code>\(, \)</code> | $(,)$ |
| Corchete | <code>\{ , \}</code> | $\{, \}$ |
| Flechas | <code>\leftarrow, \rightarrow, \uparrow, \downarrow</code> | $\leftarrow, \rightarrow, \uparrow, \downarrow$ |
| Alpha | <code>\alpha</code> | α |
| Beta | <code>\beta</code> | β |
| Gamma | <code>\gamma</code> | γ |
| Mu | <code>\mu</code> | μ |
| Pi | <code>\pi</code> | π |
| Infinito | <code>\infty</code> | ∞ |
| Barra superior | <code>\bar{letra}</code> | \bar{x} |

[\(Volver a la lectura\)](#)

9.2 R

R es un entorno diseñado principalmente para el tratamiento de datos, cálculo y desarrollo gráfico. Su diseño viene influenciado principalmente por dos lenguajes de programación existentes: S y Scheme. El resultado es un lenguaje muy similar a S pero la semántica se deriva a Scheme (Conesa, s.f.). Es un lenguaje que permite trabajar fácilmente con vectores y matrices y ofrece diversas herramientas para el análisis de datos. Además, es un software libre, esto es, que permite al usuario copiar, distribuir, estudiar, cambiar e incluso mejorar dicho software.

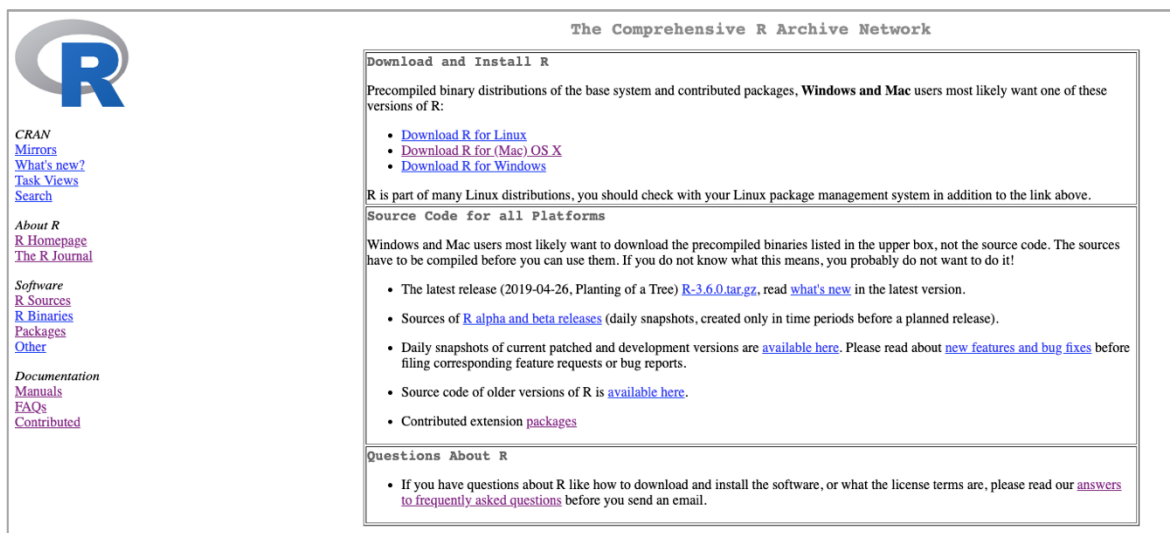
También es uno de los lenguajes de programación más utilizados en investigación clínica, siendo popular en minería de datos, investigación biomédica o bioinformática, por tanto, si sabemos manejarnos con R seremos capaces de entender mucha de la literatura científica más actual.

Tal como se indica en la web del software (The R Foundation, s.f.) ,uno de los puntos fuertes de R es la facilidad con la que se pueden producir gráficos de alta calidad junto con publicaciones que incluyen símbolos matemáticos y fórmulas. Otra

de las ventajas que posee R es la gran cantidad de webs y foros que existen dedicados a mejorar este software y dispuestos a ayudar al usuario en cualquier duda que se le plantee. Asimismo, no solo se trata de un sistema/lenguaje estadístico, sino más bien de un entorno que además de implementar técnicas estadísticas, puede extenderse muy fácilmente a otras áreas a través de paquetes (disponibles en <https://cran.r-project.org/>). Destacar que R puede utilizarse también como calculadora científica ordinaria.

9.2.1 Descarga e instalación de R

En la web de R (The R Foundation, s.f.) se encuentra a disposición de cualquier usuario toda la información relativa a R, incluyendo una gran cantidad de manuales ¹¹. Como se trata de un software libre, puede descargarse un ejecutable a través de la CRAN (Comprehensive R Archive Network)

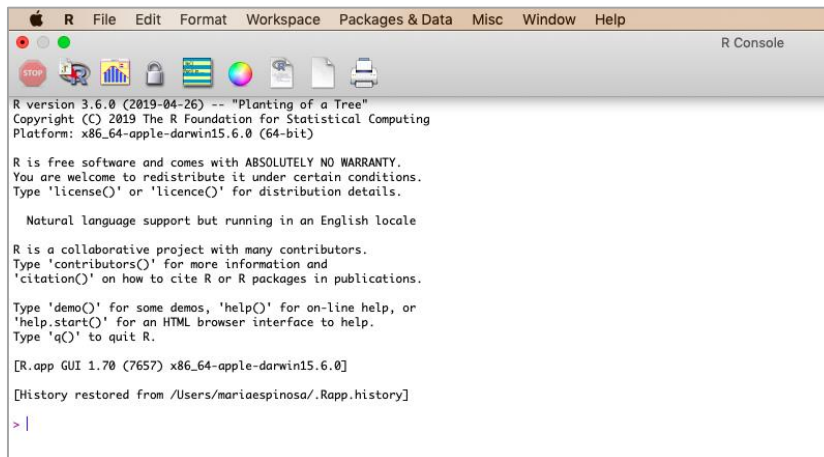


The screenshot shows the CRAN website interface. On the left, there is a navigation menu with links for CRAN, Mirrors, What's new?, Task Views, Search, About R, R Homepage, The R Journal, Software, R Sources, R Binaries, Packages, Other, Documentation, Manuals, FAQs, and Contributed. The main content area is titled "The Comprehensive R Archive Network" and contains a section "Download and Install R". This section provides instructions for downloading precompiled binary distributions for Windows and Mac users, and source code for all platforms. It includes a list of download links for Linux, Mac OS X, and Windows. Below this, there is a section "Source Code for all Platforms" which explains that source code must be compiled and provides links to the latest release, alpha and beta releases, and older versions. A "Questions About R" section at the bottom offers a link to frequently asked questions.

Figura 9.1 Descarga del ejecutable para la instalación de R
Fuente: Elaboración propia

Se selecciona el archivo .exe descargado y se siguen los pasos que indicará el ejecutable. Una vez finalizado el proceso de instalación, ejecutamos R y nos aparecerá una ventana tal y como se muestra. Esta ventana es la consola de R y es donde se escriben los comandos.

¹¹ <https://cran.r-project.org/manuals.html>

The image shows a screenshot of the R Console window on a Mac. The window title is "R Console" and the menu bar includes "R", "File", "Edit", "Format", "Workspace", "Packages & Data", "Misc", "Window", and "Help". The console output displays the R version (3.6.0), copyright information, and various help messages. The prompt ">|" is visible at the bottom.

```
R version 3.6.0 (2019-04-26) -- "Planting of a Tree"
Copyright (C) 2019 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin15.6.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[R.app GUI 1.70 (7657) x86_64-apple-darwin15.6.0]
[History restored from /Users/mariaespinosa/.Rapp.history]

> |
```

Figura 9.2 Consola de R
Fuente: Elaboración propia

9.2.2 Notas sobre aspectos básicos

- **Ayuda:** R dispone de ayuda que permite al usuario consultar off-line cualquier duda que tenga sobre cómo utilizar alguno de los comandos. Por ejemplo, si queremos consultar como usar la función *plot*, bastará con escribir en la consola: *help(solve)*, ' ?' (*solve*). Para búsquedas más avanzadas, *help.search(plot)*.
- **Case sensitivity:** esto es, el lenguaje R distingue entre mayúsculas y minúsculas.
- **Importar código:** si el usuario guarda código o funciones que ha creado en un fichero externo *comandos.R*, estos se pueden volver a cargar en cualquier sesión de R introduciendo en la consola *source("comandos.R")*.
- **Exportar código:** el comando *sink("data")* almacena la salida de los comandos posteriores en el archivo "data". Se usa principalmente para guardar salidas de datos en archivos de texto.
- **Workspace:** o "sitio de trabajo", nos indica donde estamos trabajando y donde se guardan los ficheros que creamos. Con el comando *getwd()* podemos consultar el directorio de trabajo; y con *setwd()* podemos modificarlo. Al iniciar R, se abre de forma automática un espacio de trabajo en el cual se van almacenando todos los datos, las funciones, resultados... que se usan durante una determinada sesión. Todo ello puede guardarse para su posterior utilización en un archivo con extensión ".RData" (menú file). Si queremos cargar el archivo "R.Data", usaremos el comando *load()*. No obstante, es recomendable guardar las funciones y los objetos en archivos propios (Febrero, Galeano, González y Pateiro; 2008)

- **Paquetes y librerías:** la instalación de R ya viene con determinadas funciones básicas, no obstante, lo que caracteriza este software es la facilidad con la que se pueden ampliar añadiendo nuevas funciones capaces de abordar tareas diferentes. Así, los paquetes (*packages*) son “colecciones de funciones, datos y código que se almacenan en una carpeta conforme a una estructura bien definida, fácilmente accesible para R” (Santana y Hernández, 2016). En primer lugar, mediante la instrucción `install.packages(“nombre.del.paquete”)` descargamos el paquete en nuestro ordenador y lo instalamos. No obstante, para poder acceder a los elementos que se encuentran en él, deberemos cargarlo mediante el comando `library(“nombre.del.paquete”)` o bien `nombre.del.paquete::función.del.paquete`.
- **Listado:** con el comando `ls()`, obtenemos una lista de todos los objetos que tenemos cargados en nuestro directorio.
- **Borrar:** para eliminar objetos de nuestro espacio de trabajo, introducimos en la consola de R la instrucción `rm (objeto)`.
- **Historial de comandos:** si queremos consultar, o volver a ejecutar algún comando ya ejecutado, podemos acceder a ellos con la flecha del teclado \uparrow . También, con `history()` se obtiene el historial de todos los comandos y funciones utilizados en la sesión en un archivo de texto. Si queremos corregir una línea de código, nos podemos desplazar sobre ella con las flechas $\leftarrow \rightarrow$ del teclado.

9.2.3 Conceptos y funciones básicas

Ficheros-Scripts

Tal como se ha comentado, los comandos y las órdenes se escriben en la consola de R. No obstante, si queremos realizar más de una acción, es recomendable hacerlo en un fichero de comandos editable o *script*, es decir, se trabaja en una pantalla a parte, pero dentro de R. Además, podemos modificar el fichero y añadir comentarios con el símbolo '#'. De esta forma, todo el código que se pretenda utilizar quedará en un fichero externo, fácilmente ejecutable. Para crearlo, podemos hacerlo desde la pestaña 'Archivo' y seleccionando 'Nuevo documento'.

Clases de objeto

R es un lenguaje orientado a objetos, y estos son fácilmente modificables con operadores lógicos, aritméticos y comparativos; y también con funciones (que a su vez son también objetos). Cualquier asignación en R se realiza con '`<-`' o con '`=`'. Las clases de objetos que existen son:

- **Vectores:** son el objeto más básico de R. Se crean con la función `c()`, que concatena los elementos que aparecen dentro del paréntesis. Para poder acceder a un elemento del vector, se indica el nombre del vector seguido de la posición a la cual queremos acceder entre corchetes. Por ejemplo, creamos el vector numérico de nombre `vector1` y queremos acceder al elemento situado en la tercera posición:

```
vector.num= c(1,2,3,4) #Creamos el vector
vector.num[3] #Accedemos al elemento situado en la 3ª posición
```

La longitud del vector la podemos obtener mediante `length(vector)`. Además, es necesario que los elementos del vector sean todos del mismo tipo.

Podemos crear también factores, un tipo específico de vector que podemos entender como una variable categórica con un número finito de valores o niveles (Santana y Hernández, 2016). La instrucción será `factor()`. Por ejemplo, tenemos la variable 'sexo':

```
sexo= c('H', 'H', 'M', 'H', 'M' )
```

Para convertirla en factor:

```
sexo=factor(sexo)
sexo
[1] H H M H M
Levels: H M
```

Tal como bien se indica en la web 'Introducción a R' (Santana y Hernández, 2016) en muchas ocasiones, los niveles de los factores resultan poco ilustrativos por lo que a su significado respecta. Así pues, para clarificar este aspecto, la función `factor` permite especificar estos aspectos: para explicitar los niveles del factor, incluiremos *levels*; mientras que para asignar las etiquetas a cada uno de los niveles lo haremos mediante *labels*. Por ejemplo,

```
sexo
=factor(sexo, levels=c("H", "M"), labels=c("Hombre", "Mujer"))

sexo
[1] Hombre Hombre Mujer Hombre Mujer
Levels: Hombre Mujer
```

De esta forma, cuando tengamos que utilizar este factor los nombres que aparecerán serán los asignados en las etiquetas (*labels*).

Funciones vectores

| | |
|--|---|
| <code>v=c(, , ,)</code> | Crear un vector concatenado |
| <code>v[i]</code> | Acceder a la componente i del vector v |
| <code>v=numeric()</code> | Crear un vector numérico vacío |
| <code>v=character()</code> | Crear un vector de caracteres vacío |
| <code>v=complex()</code> | Crear un vector complejo vacío |
| <code>v=factor()</code> | Crear un factor vacío |
| <code>as.factor(v)</code> | Coerción de un vector v a ser factor |
| <code>factor(x, levels=c(), labels=c())</code> | Crear un factor con niveles y etiquetas |
| <code>length(v)</code> | Longitud del vector v |
| <code>x : y</code> | Secuencia del número x hasta el número y |
| <code>seq()</code> | Función para generar secuencias más complejas |

- **Matrices:** son una generalización multidimensional de los vectores. Para crear una matriz A, se utiliza la función `matrix()` a la cual se le pasan diferentes argumentos:

```
A= matrix (ncol=3, nrow=3, c(1,2,3,4,5,6,7,8,9), byrow=TRUE)
```

El argumento `ncol` nos indica el número de columnas, y `nrow` el número de filas de la matriz. A continuación, se introducen los datos de nuestra matriz, y con `byrow=TRUE` indicamos que la matriz se rellene por filas, de forma que quedará:

```
A
      [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

En la siguiente tabla quedan recogidas las principales operaciones y funciones que pueden llevarse a cabo en R con la estructura de matriz:

Funciones matrices

| | |
|---|--|
| <code>matrix(ncol, nrow, c(), byrow=TRUE/FALSE)</code> | Crear una matriz de dimensión nrow x ncol |
| <code>A[i, j]</code> | Elemento (i,j) de la matriz A |
| <code>A[, j]</code> | Selecciona todos los elementos de la columna j |
| <code>A[i,]</code> | Selecciona todos los elementos de la fila i |

| | |
|--------------------------|--|
| <code>A[i:j, i:j]</code> | Selecciona las filas de la i a la j, y las columnas de la i a la j |
| <code>ncol(A)</code> | Número de columnas de A |
| <code>nrow(A)</code> | Número de filas de A |
| <code>diag(A)</code> | Vector con la diagonal de la matriz A |
| <code>diag(v)</code> | Crea una matriz de diagonal el vector v |
| <code>t(A)</code> | Matriz transpuesta de A |
| <code>cbind(...)</code> | Combina secuencias de vectores por columnas |
| <code>rbind(...)</code> | Combina secuencias de vectores por filas |

- **Listas:** son una colección ordenada de objetos de cualquier tipo y se construye con la función `list()`. En este trabajo no se trabajará con estas, así pues, no se profundiza en su estructura.
- **Data frame:** un *data frame* podría entenderse como una matriz de datos donde las columnas se corresponden con cada una de las variables y las filas con cada uno de los individuos. Es una forma de organización de los datos propia de R que facilita el tratamiento de estos. Para crear un *data frame* se hará mediante el comando `data.frame(variable1, variable 2, ...)` donde 'variable 1', 'variable 2' ... no son más que vectores (numéricos o de tipo factor) creados tal como se explica en uno de los apartados anteriores.

```
años =c(14,13,13,28,19) #Variable numérica
sexo=factor(c(' H' , ' H' , ' M' , ' H' , ' M' )) #Variable de tipo
factor
ejemplo.dataframe= data.frame (años,sexo)
```

Para visualizar el *data frame*, tan solo hay que escribir el nombre en la consola de R:

```
ejemplo.dataframe
```

```
  años sexo
1   14   H
2   13   H
3   13   M
4   28   H
5   19   M
```

Si queremos manipular nuestro conjunto de datos, en primer lugar, con la función `attach(nombre.del.dataframe)`, indicamos que vamos a acceder al

data frame. Esta operación resulta necesaria, ya que si no se realiza, no se puede trabajar con ninguna de las variables del conjunto de datos. Así pues, nos bastará con escribir el nombre de la variable que queramos trabajar o bien con *nombre.del.dataframe\$nombre.de.la.variable*. Es decir,

```
attach(ejemplo.dataframe)
años #Escribimos nombre de la variable
[1] 14 13 13 28 19
ejemplo.dataframe$años #Accedemos a través del símbolo $
[1] 14 13 13 28 19
```

Con la instrucción *str(nombre.del.dataframe)*, podemos visualizar la estructura del conjunto de datos o el tipo de variables que lo forman:

```
str(ejemplo.dataframe)
' data.frame' : 5 obs. of 2 variables:
 $ años: num 14 13 13 28 19
 $ sexo: Factor w 2 levels "H","M": 1 1 2 1 2
```

Con la función *detach()* realizamos la operación inversa de *attach()*, y así desvinculamos del espacio de trabajo el data frame.

Funciones *data frame*

| | |
|----------------------------------|--|
| <code>data.frame()</code> | Crear un data frame |
| <code>attach ()</code> | Acceder al data frame |
| <code>detach()</code> | Desvincular el data frame |
| <code>str()</code> | Estructura del data frame |
| <code>names()</code> | Nombre de las variables del data frame |
| <code>dataframe\$variable</code> | Acceder a una variable del data frame |

Nota: cuando se generan o crean objetos y se les asigna un nombre, para poder verlos, tan solo se ha de escribir el nombre del objeto en la consola de R y pulsar la tecla Intro. Es decir:

```
> v<-c(1:10) #Crear vector
> v #Visualizar vector
[1] 1 2 3 4 5 6 7 8 9 10
> A <- matrix(v,nrow=2,ncol=5,byrow=TRUE) #Crear matriz
> A #Visualizar matriz
      [,1] [,2] [,3] [,4] [,5]
[1,]  1   2   3   4   5
[2,]  6   7   8   9  10
>
```

Figura 9.3 Visualización de los objetos creados
Fuente: Elaboración propia

Funciones básicas

R no solo es un programa especializado en cálculo estadístico, sino que también puede utilizarse como calculadora científica. En las siguientes tablas quedan recogidas las principales operaciones y funciones que se pueden realizar:

| Operadores | |
|---------------------|----------------------|
| Aritméticos | Comparativos |
| + Adición | < Menor que |
| - Sustracción | > Mayor que |
| * Multiplicación | <= Menor o igual que |
| / División | >= Mayor o igual que |
| ^ Potencia | = = Igual a |
| % % División entera | != Diferente |

| Funciones matemáticas | |
|-----------------------|---------------------------------------|
| exp() | Exponencial |
| log() | Logaritmo natural(ln) |
| log10() | Logaritmo en base 10 |
| factorial() | Factorial de un número |
| sqrt() | Raíz cuadrada |
| sin() | Seno |
| cos() | Coseno |
| tan() | Tangente |
| Re() | Parte real número imaginario |
| Im() | Parte imaginaria
número imaginario |
| Arg() | Argumento número imaginario |
| Mod() | Módulo número imaginario |
| Conj() | Conjugado número imaginario |
| pi | Número Π |
| exp(1) | Número e |

En capítulos posteriores se incidirá en otras funciones más específicas de las diferentes áreas matemáticas.

[\(Volver a la lectura\)](#)

9.3 Markdown

9.3.1 ¿Qué es Markdown?

Markdown es un “lenguaje de marcado ligero que trata de conseguir la máxima legibilidad y facilidad de publicación tanto en su forma de entrada como de salida” (Wikipedia, s.f.). Este lenguaje fue creado por John Gruber en 2004, junto con Aaron Swartz. La finalidad que perseguían era hacer que el usuario pudiese escribir usando un formato de texto fácil de leer y fácil de escribir, incorporando la posibilidad de convertir su documento en HTML estructuralmente válido (Wikipedia, s.f.).

Platas y García (2017) señalan que una de las ideas más interesantes que presenta Markdown es el hecho de la creación de documentos o archivos de texto planos fácilmente legibles sin necesidad de tener apariencia de haber sido marcados con elementos de formateo que hacen más difícil su lectura. Además, con Markdown pueden producirse archivos que son legibles como texto plano y que a su vez pueden utilizarse en otros programas o plataformas (Simpkin, 2017).

Sintaxis Markdown

Título

Subtítulo

Este es un ejemplo de texto que da entrada a una lista genérica de elementos:

- Elemento 1
- Elemento 2
- Elemento 3

Este es un ejemplo que da entrada a una lista numerada:

1. Elemento 1
2. Elemento 2
3. Elemento 3

Podemos añadir **negrita** o *cursiva* utilizando tan solo *.

Sintaxis HTML

```
<head>  
<title>Ejemplo</title>  
</head>
```

```
<body>
```

```
<div class="container-fluid main-container">  
<div class="fluid-row" id="header">  
<h1 class="title toc-ignore">Ejemplo</h1>  
</div>
```

```
<div id="titulo" class="section level2">  
<h2>Título</h2>
```

```
<div id="subtitulo" class="section level3">  
<h3>Subtítulo</h3>
```

```
<p>Este es un ejemplo de texto que da entrada a una lista genérica de elementos:</p>
```

```
<ul>  
<li>Elemento 1</li>  
<li>Elemento 2</li>  
<li>Elemento 3</li>  
</ul>
```

```
<p>Este es un ejemplo que da entrada a una lista numerada:</p>
```

```
<ol style="list-style-type: decimal">  
<li>Elemento 1</li>  
<li>Elemento 2</li>  
<li>Elemento 3</li>  
</ol>
```

```
<p>Podemos añadir <strong>negrita</strong> o <em>cursiva</em> utilizando tan solo *.</p>  
</div>
```

Figura 9.4 Comparación sintaxis Markdown y sintaxis HTML
Texto: <https://markdown.es/>

Y el resultado sería:

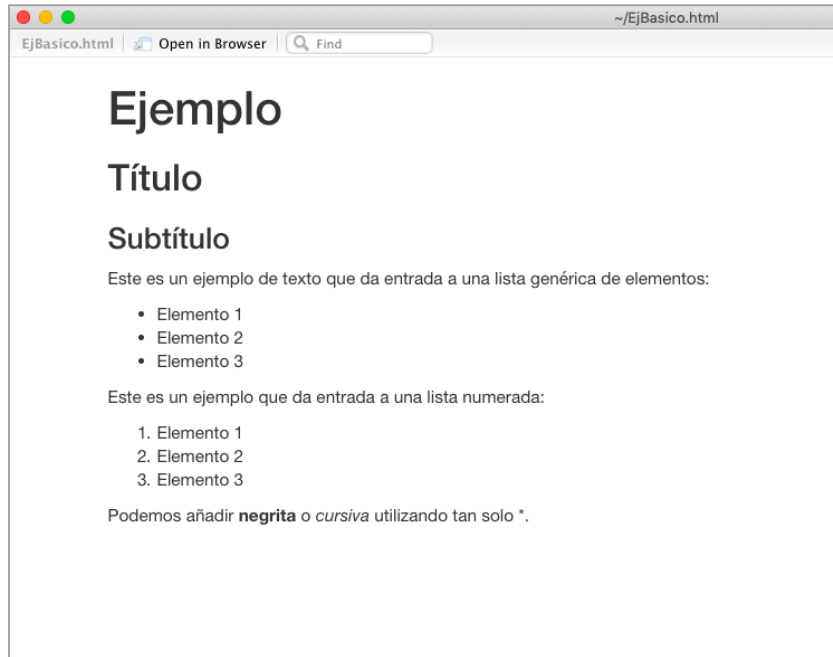


Figura 9.5 Salida del código anterior
Fuente: Elaboración propia

Así pues, para poder escribir utilizando el lenguaje Markdown necesitamos conocer su sintaxis y sus conceptos básicos.

9.3.2 Sintaxis y conceptos básicos de Markdown

Párrafos y saltos de línea

Para generar un párrafo nuevo en Markdown, se pulsa dos veces la tecla Intro. Si lo que queremos es introducir un salto de línea y empezar en la siguiente (dentro del mismo párrafo), hay que pulsar dos veces la barra espaciadora y seguidamente la tecla Intro.

Encabezados o títulos

Para crear títulos y subtítulos se insertan almohadillas ' #' , una para cada nivel, pudiéndose añadir hasta seis niveles. Es decir:

Encabezado 1 (Primer nivel).

Encabezado 2 (Segundo nivel).

Encabezado 3 (Tercer nivel).

.....



Encabezado 1

Encabezado 2

Encabezado 3

Énfasis

Para enfatizar palabras o frases, escribimos la palabra o frase entre dos asteriscos a cada lado ****negrita**** o dos guiones bajos `__negrita__`, y obtendremos en ambos casos: **negrita**. Si queremos resaltar palabras o frases con la cursiva, escribimos entre un asterisco a cada lado **cursiva** o entre un guion bajo `_cursiva_`, y obtendremos: *cursiva*.

Si pretendemos combinar ambos estilos, combinaremos también su sintaxis, o sea, escribiremos las palabras o frases con tres asteriscos a cada uno de los lados, o con tres guiones bajos, bien *****negrita y cursiva***** o bien `__negrita y cursiva__`, para así obtener: ***negrita y cursiva***.

Listas

Podemos ordenar la información mediante diferentes tipos de listas. Para crear una lista genérica, se permite escribir bien un asterisco *, el símbolo de la suma +, o un guion -, antes de cada elemento de la lista y separado por un espacio:

- Elemento 1	→	• Elemento 1
+ Elemento 2		• Elemento 2
* Elemento 3		• Elemento 3

No importa el símbolo que se utilice, incluso pueden combinarse.

Para crear una lista numerada, debemos utilizar la sintaxis ' número.':

1. Elemento 1	→	1. Elemento 1
2. Elemento 2		2. Elemento 2
3. Elemento 3		3. Elemento 3

Podemos combinar ambos tipos de lista, y también anidarlas introduciendo sangría en el nivel que queramos desplazar:

1. Elemento 1	→	1. Elemento 1
2. Elemento 2		2. Elemento 2
- Elemento 2.1		• Elemento 2.1
- Elemento 2.2		• Elemento 2.2
1. Elemento 2.2.1		1. Elemento 2.2.1
2. Elemento 2.2.2		2. Elemento 2.2.2
3. Elemento 3		3. Elemento 3

Línea horizontal

Si queremos separar partes del documento de forma visual mediante una línea horizontal, se introduce en una línea en blanco tres asteriscos, tres guiones o tres guiones bajos:

```
***   ---   _
```

y se obtendrá:

Citas

Para citar una frase o un fragmento de texto, se añade al principio de este el símbolo '>':

```
> Esto es una cita- Autor Desconocido.
```

Y obtenemos,

```
"Esto es una cita- Autor Desconocido.
```

Si la cita es un texto con más de un párrafo, se ha de añadir el símbolo '>' a cada uno de los párrafos.

Código

Si redactamos un documento de carácter científico, en algún momento podemos estar interesados en añadir código (en nuestro caso de R) bien en bloques o en línea. Si queremos hacerlo de la primera forma, debemos colocar el código entre tres acentos abiertos ``` , al principio y al final. Si tan solo queremos añadir una línea, lo haremos entre un acento abierto ` , al principio y al final.

Tablas

Para crear tablas en Markdown, seguiremos los pasos indicados en la web *Markdown.es* (Cristóbal, s.f.). En primer lugar, escribimos las cabeceras de cada columna, separadas por un guion vertical, es decir,

```
Título 1 | Titulo 2 | Título 3
```

Para especificar que los encabezados se terminan, ha de escribirse:

```
-- | -- | --
```

tantas veces como columnas se tengan.

Una vez hecho esto, se añade el contenido de la tabla. Se crea exactamente igual al encabezado, pero como introducimos una línea vacía, Markdown es capaz de interpretar eso correctamente. Así, si escribimos el siguiente código,

```
Título 1 | Titulo 2 | Título 3
-- | -- | --
Contenido 1-1 | Contenido 1-2 | Contenido 1-3
Contenido 2-1 | Contenido 2-2 | Contenido 2-3
Contenido 3-1 | Contenido 3-2 | Contenido 3-3
```

obtenemos la siguiente tabla:

TÍTULO 1	TÍTULO 2	TÍTULO 3
Contenido 1-1	Contenido 1-2	Contenido 1-3
Contenido 2-1	Contenido 2-2	Contenido 2-3
Contenido 3-1	Contenido 3-2	Contenido 3-3

Enlaces

Existen varios tipos de links, tal como se indica en la web *Markdown.es* (Cristóbal, s.f.). Por un lado, los enlaces automáticos, es decir, introducir el enlace de la página web directamente en el texto. Para ello, copiamos el link entre los símbolos < enlace web >. Por otro lado, los enlaces en línea, que sirve para colocar el enlace en línea con el texto. La palabra que queremos enlazar se escribe entre corchetes y seguidamente el enlace web entre paréntesis. Esto es:

[UJI](https://www.uji.es/)  [UJI](https://www.uji.es/)

Imágenes

Para añadir imágenes, lo haremos de forma similar a los enlaces (Cristóbal, s.f.). No obstante, entre paréntesis en lugar de escribir la url de la página web, escribiremos la ruta de la imagen, y además se añadirá un signo de exclamación ' ! ' :



! [Texto.alternativo](ruta/de/la/imagen/logo_google.jpg)  

Figura 9-1. Fuente: Wikipedia

Notas al pie de página

Para insertar notas al pie de página, siguiendo la web (Carles, 2019), justo después de la palabra donde queramos añadir la nota al pie de página, introduciremos '[^nombre de la nota]'. Seguidamente, en cualquier parte del documento escribimos:

[^nombre de la nota]: Contenido de la nota

De esta forma, se añadirá al final de la página la nota con el contenido indicado. Es recomendable definir las notas todas juntas al final del documento. Por ejemplo:

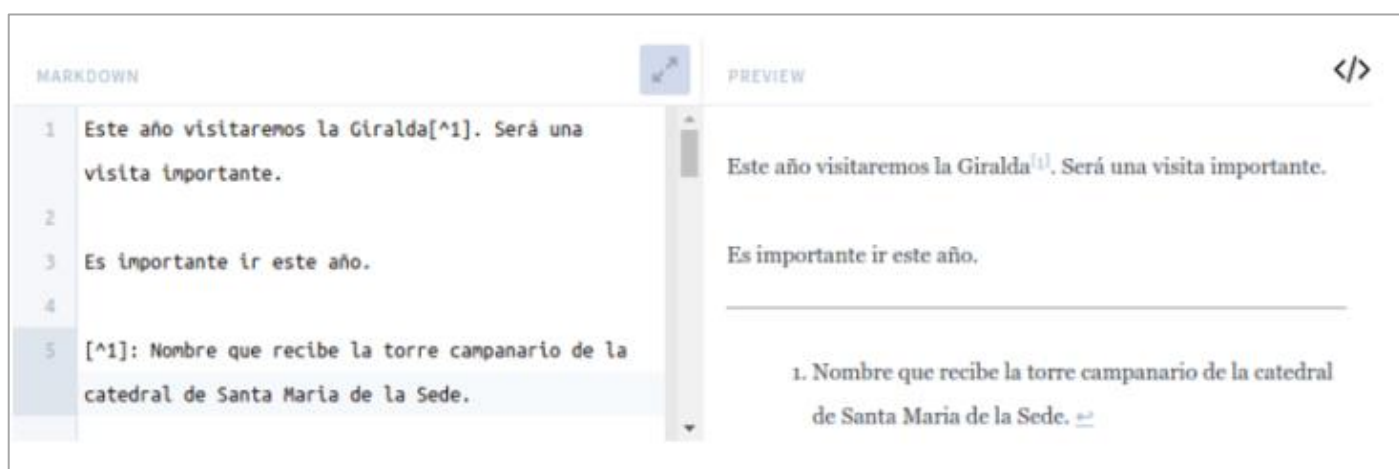


Figura 9.6 Nota al pie de página.

Fuente: <https://geekland.eu/aprender-markdown-en-minutos/>

Evitar Markdown

Para poder escribir símbolos como los asteriscos, los guiones, los corchetes, paréntesis, las almohadillas, etc; sin que Markdown los asocie a su función inicial, bastará con escribir una barra invertida '\' justo antes de cada símbolo. Si, además, queremos escribir texto que no se visualice en el documento final (HTML o PDF), se escribirá entre "<! ---Texto_oculto --->".

[\(Volver a la lectura\)](#)

9.4 R Markdown

R Markdown es un paquete de R que permite generar documentos que integran texto, imágenes y código de R, más los resultados de la ejecución del código. Es decir, R Markdown es el centro de este trabajo, ya que con este paquete se producirán los documentos de carácter científico que incluirán código R, cálculos, fórmulas, texto plano e imágenes. Además, uno de los aspectos más importantes es que "permite que se reproduzca el análisis realizado y si se incorporan nuevos datos, los resultados se actualizarán" (Zubcoff,2017). Los informes se componen por el

fichero que genera el informe, que es formato R Markdown, con extensión .Rmd; y el informe generado, en Word, PDF o HTML (Zubcoff,2017). La visualización en Word hace que el archivo sea fácilmente editable, el formato HTML permite visualizarlos a través de navegadores web y por último el PDF, que necesitaremos tener instalado LaTeX (sistema de edición de textos) (Santana y Hernández,2016).

Así, para poder generar un archivo .Rmd, deberemos conocer la sintaxis de R para el código (véase apartado 2.2), la de Markdown para la redacción del texto(véase apartado 2.4) y LaTeX para la escritura de fórmulas (véase apartado 2.6). No obstante, para poder integrar el código de forma correcta hay que recurrir a lo que se conoce como *Chunk*, que incluirá el código R. La sintaxis para introducir el *chunk* más simple es¹²:

```
```\r}
Líneas de código R
```
```

Dentro del corchete podemos incluir diferentes opciones (separadas mediante una coma) que nos permiten: mostrar o no el código, mostrar o no los resultados, cambiar la anchura o altura de los gráficos, organizar el código... En la siguiente tabla se muestran las diferentes opciones:

| Opción | Por defecto | Efecto |
|----------------|-------------|---|
| eval | TRUE | Indica si se va a evaluar el código |
| echo | TRUE | Indica si se muestra el código a la par que los resultados |
| warning | TRUE | Indica si se muestran advertencias |
| error | FALSE | Indica si se muestran errores |
| message | TRUE | Indica si se muestran mensajes |
| tidy | FALSE | Indica si se muestra el código de forma organizada |
| results | "markup" | Opciones: "markup" (marca los resultados con el formato de salida del documento), "asis" (escribe los resultados en el documento final sin reformatearlos), "hold" (escribe los resultados al final de la ejecución de todo el <i>chunk</i>), "hide" (oculta los resultados en el documento final) |
| cache | FALSE | Indica si se guardan los resultados en <i>cache</i> |

¹² Podemos insertar el Chunk de forma automática pulsando Ctrl+Alt+I

| | | |
|-------------------------|------|--|
| <code>comment</code> | "##" | Carácter de comentario para anteponer resultados |
| <code>fig.width</code> | 7 | Ancho en pulgadas para figuras generadas en el chunk |
| <code>fig.height</code> | 7 | Alto en pulgadas para figuras generadas en el chunk |

¹Fuente: <https://www.rstudio.com/wp-content/uploads/2015/03/rmarkdown-spanish.pdf>

Para generar un documento de R Markdown, tan solo tenemos que ir a 'File/New File / R Markdown'. Una vez seleccionado, aparece una ventana en la cual tendremos que poner título a nuestro documento y seleccionar el formato de salida (HTML, Word o PDF).

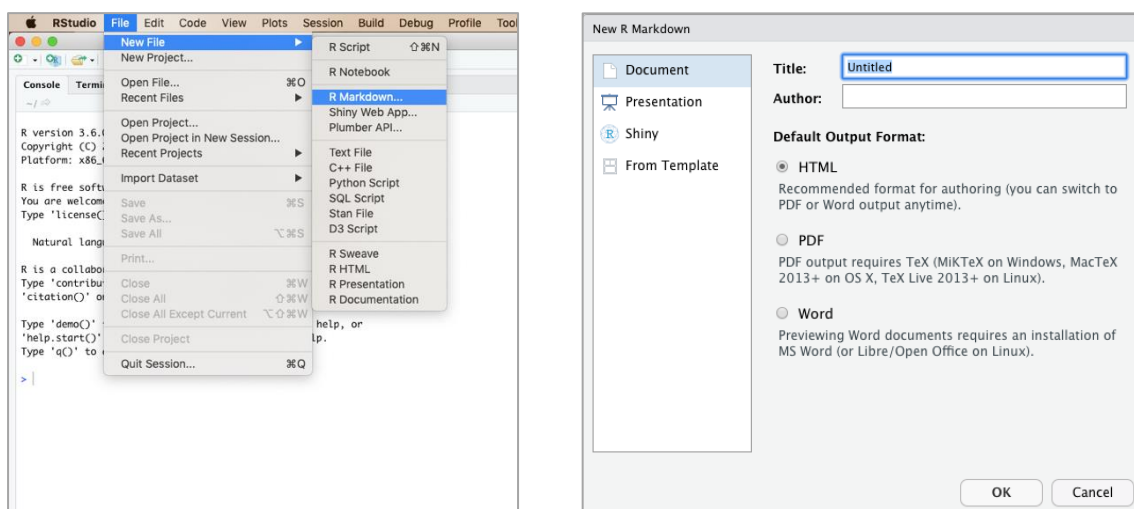



Figura 9.7 Crear documento R Markdown
Fuente: Elaboración propia

Así, de forma automática, RStudio genera una plantilla R Markdown. Para ejecutarlo, tan solo tenemos que hacer click en 'Knit'  .

En la web de R Markdown, (RStudio, 2016-2018) existe información más detallada a cerca del paquete R Markdown y sus múltiples usos.

9.4.1 Presentaciones con RMarkdown

Una ventaja que presenta este paquete es, que desde la consola de RStudio, también permite crear y elaborar presentaciones, bien en formato HTML o *ioslides*, en PDF o *beamer* e incluso en Power Point. Es decir, presentaciones que al igual que los documentos, podrán incluir fórmulas, código y sus salidas (*Chunk*), imágenes y texto plano.

Para generar el archivo donde se incluirá el texto y el código, se procede de forma similar a la anterior: nos dirigimos a 'File/New File / R Markdown', y una vez seleccionado, surge una ventana. En la parte izquierda aparece qué tipo de archivo buscamos elaborar, un documento, una presentación o un documento interactivo

con Shiny. En nuestro caso, señalamos "Presentation", y tan solo debemos escribir título y autor, y escoger el formato de salida.

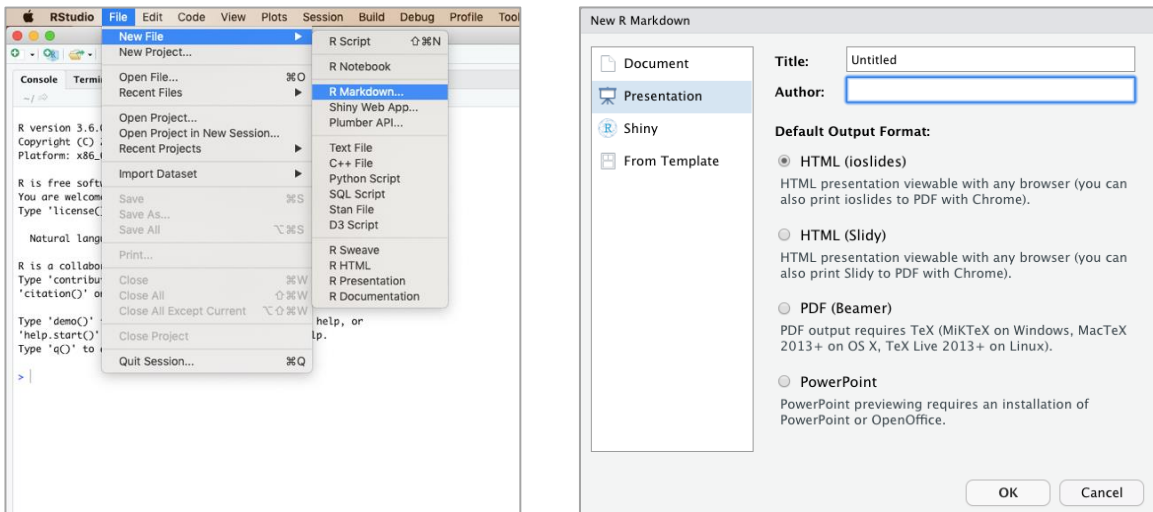



Figura 9.8 Crear presentación R Markdown
Fuente: Elaboración propia

De esto modo, se genera de forma automática, una plantilla a modo de ejemplo, que, para poder visualizar en el formato escogido, tan solo debemos hacer click en ' Knit'  .

La peculiaridad en este tipo de archivo es, que, para iniciar nueva diapositiva, podemos hacerlo con un título `## Título primera diapositiva`, o bien añadiendo tres asteriscos, `***`, si queremos la diapositiva sin título:

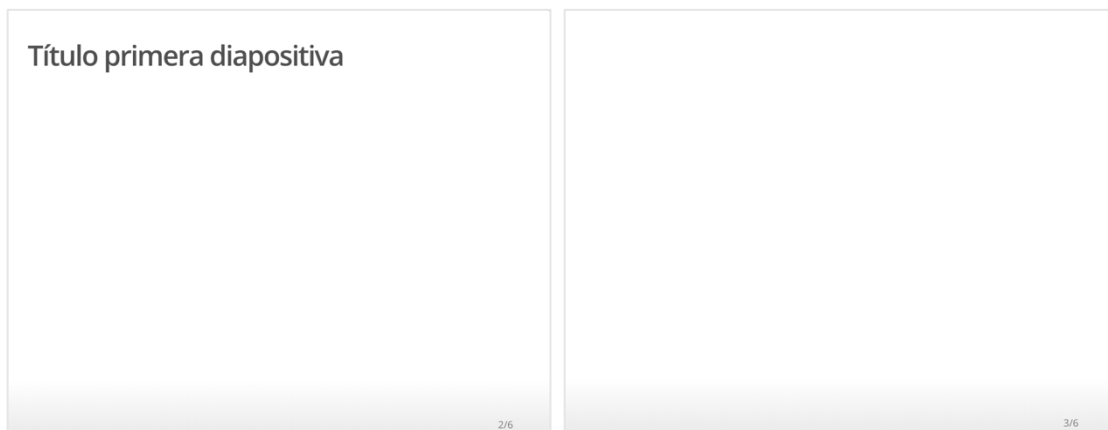


Figura 9.8 Nueva diapositiva-Con título y Sin título
Fuente: Elaboración propia

Si queremos que los elementos de una lista aparezcan de forma gradual, en primer lugar, se debe añadir a la cabecera la instrucción *incremental: true*, y en cada elemento, '>'. Si además se quiere reducir el tamaño de letra de algunas dispositivas,

en la cabecera se añadirá la instrucción *smaller: true*, y junto al título de la diapositiva que se pretenda reducir de tamaño, se colocará *{. smaller}*. Existen gran cantidad de instrucciones que permiten modificar otros aspectos relativos a la presentación, no obstante, tan solo se han comentado los esenciales.¹³

Si la presentación es en formato *beamer*, se puede cambiar el tema añadiendo en la cabecera *theme: "___"*, *colortheme: "___"*, *fonttheme: "___"*.

Es importante recalcar, que en algunos casos el formato *beamer* falla, principalmente en cuanto a salidas de código e imágenes se refiere, pues no es capaz de adaptarlo a la diapositiva. Esto no ocurre con *ioslides*, ya que realmente este formato es el pensado para incorporar de forma sencilla código de R y sus salidas.

[\[Volver a la lectura\]](#)

9.5 RStudio

Trabajar directamente con R no resulta cómodo, es por eso por lo que se creó la interfaz de RStudio, disponible para todas las plataformas (Windows, Linux y Mac).

RStudio es un entorno desarrollado integrado (IDE: Integrated Development Environment) para el lenguaje de programación R, y permite interactuar con él de forma muy simplificada. Es decir, “R será nuestro motor para el análisis de datos, y RStudio la carrocería que va a permitir manejar cómodamente toda la potencia del motor” (Santana y Hernández, 2016). Además, según sus creadores, su interfaz es muy intuitiva para el usuario con herramientas muy potentes que permiten sacar todo el potencial a R.

RStudio presenta muchas ventajas, así se indica en el sitio web (RStudio, s.f.): resaltado de la sintaxis, finalización del código, sangría inteligente. Permite ejecutar código R directamente desde el editor fuente, saltar rápidamente a definiciones de funciones; ayuda documentada integrada de R, gestiona fácilmente múltiples directorios de trabajo mediante proyectos, presenta amplias herramientas de desarrollo de paquetes, detecta errores de sintaxis e incluye la creación con paquetes como *Sweave* (para la escritura en LaTeX) o R Markdown.

9.5.1 Descarga e instalación de RStudio

Para la instalación de RStudio se requiere previamente haber instalado R (véase apartado 9.2.1). En la página web de RStudio (RStudio, s.f.) podemos

¹³ Para más detalles se recomienda consultar: <https://bookdown.org/yihui/rmarkdown/presentations.html>

encontrar el programa para descargarlo. En primer lugar, accedemos a la web y seleccionamos ' Download RStudio':

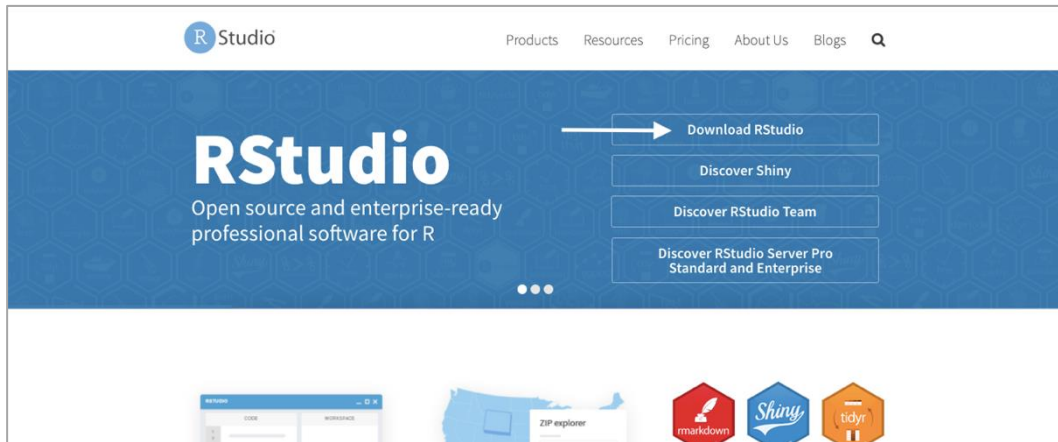


Figura 9.9 Página RStudio
Fuente: Elaboración propia

En la siguiente página, se selecciona la opción de RStudio Desktop y descargamos la versión Open Source Edition adecuada a nuestro sistema operativo:

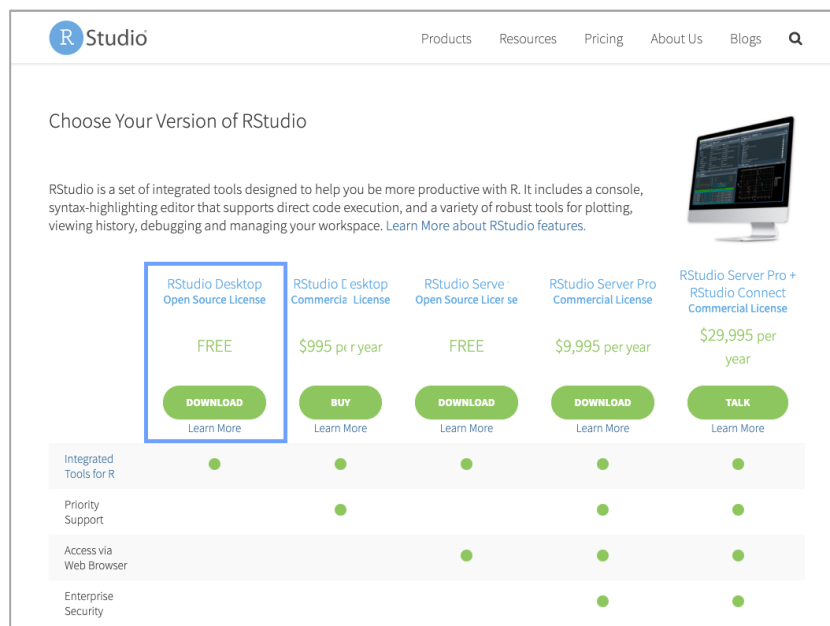


Figura 9.10 Descargar RStudio
Fuente: Elaboración propia

RStudio: The R IDE for Windows, Mac OS X, Linux, and Unix. [Products](#) [Resources](#) [Pricing](#) [About Us](#) [Blogs](#) [Q](#)

Installers for Supported Platforms

| Installers | Size | Date | MD5 |
|--|----------|------------|-----------------------------------|
| RStudio 1.2.1335 - Windows 7+ (64-bit) | 126.9 MB | 2019-04-08 | d0e2470f1f8ef4cd35a669aa323a2136 |
| RStudio 1.2.1335 - Mac OS X 10.12+ (64-bit) | 121.1 MB | 2019-04-08 | 6c570b0e2144583f7c48c284ce299eef |
| RStudio 1.2.1335 - Ubuntu 14/Debian 8 (64-bit) | 92.2 MB | 2019-04-08 | c1b07d0511469abfe582919b183eee83 |
| RStudio 1.2.1335 - Ubuntu 16 (64-bit) | 99.3 MB | 2019-04-08 | c142d69c210257fb10d18c045fff13c7 |
| RStudio 1.2.1335 - Ubuntu 18 (64-bit) | 100.4 MB | 2019-04-08 | 71a8d1990c0d97939804b46cfb0aea75 |
| RStudio 1.2.1335 - Fedora 19+/RedHat 7+ (64-bit) | 114.1 MB | 2019-04-08 | 296b6ef88969a91297fab6545f256a7a |
| RStudio 1.2.1335 - Debian 9+ (64-bit) | 100.6 MB | 2019-04-08 | 1e32d4d6f6e216f086a81ca82ef65a91 |
| RStudio 1.2.1335 - OpenSUSE 15+ (64-bit) | 101.6 MB | 2019-04-08 | 2795a63c7efd8e2aa2dae86ba09a81e5 |
| RStudio 1.2.1335 - SLES/OpenSUSE 12+ (64-bit) | 94.4 MB | 2019-04-08 | c65424b06ef6737279d982db9eeefcae1 |

Zip/Tarballs

| Zip/tar archives | Size | Date | MD5 |
|--|----------|------------|----------------------------------|
| RStudio 1.2.1335 - Windows 7+ (64-bit) | 186.6 MB | 2019-04-08 | f1e013ade0c241969400507cf258e0ad |
| RStudio 1.2.1335 - Ubuntu 14/Debian 8 (64-bit) | 137.6 MB | 2019-04-08 | e3e1ea2dd113fd9cfd40bc5035effdde |
| RStudio 1.2.1335 - Ubuntu 18 (64-bit) | 147.8 MB | 2019-04-08 | 5ee7dd7b501675f0a631c62d403ea1b6 |
| RStudio 1.2.1335 - Debian 9+ (64-bit) | 148.1 MB | 2019-04-08 | 8090451cb7d520633eba80fd335ad4c1 |
| RStudio 1.2.1335 - Fedora 19+/RedHat 7+ (64-bit) | 147.2 MB | 2019-04-08 | 34630cd7c66c3429879bd79982349380 |

Source Code

Figura 9.11 Versiones RStudio según el sistema operativo
Fuente: Elaboración propia

Una vez instalado, arrancamos el programa y se nos muestra una ventana de la siguiente forma:

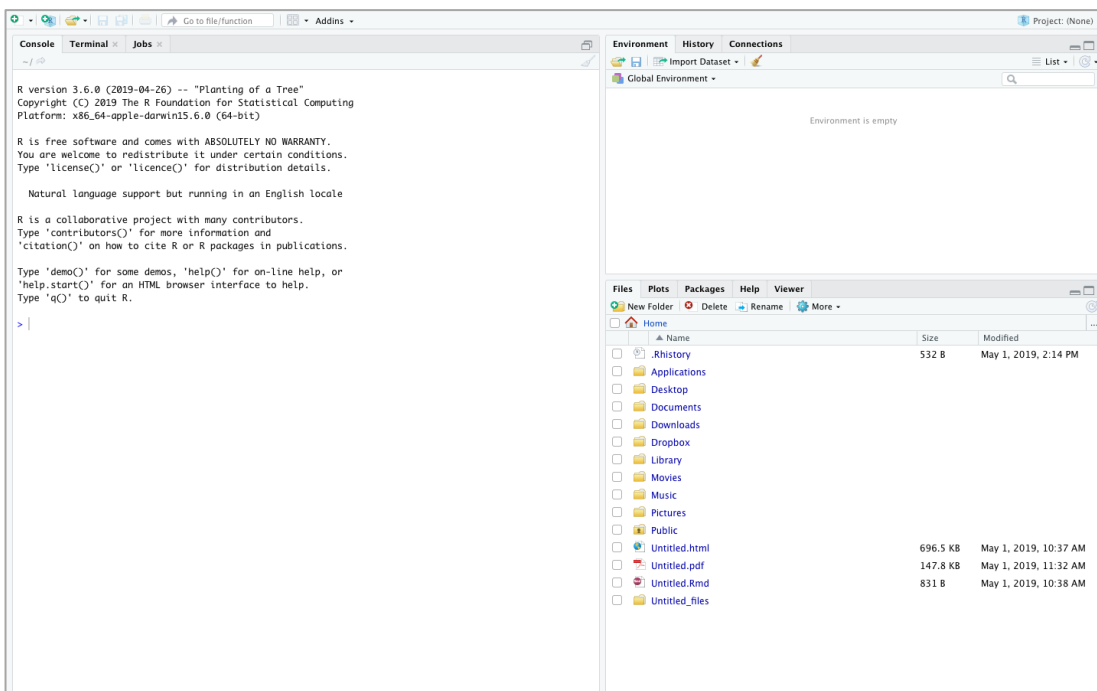



Figura 9.12 Ventana RStudio
Fuente: Elaboración propia

La pantalla queda dividida en tres secciones (la disposición de estas puede modificarse a gusto del usuario):

- En la parte superior derecha se encuentra el '**Global Environment**', es decir, el entorno de trabajo, donde aparecen los bancos de datos que se crean o se utilizan, las variables, funciones que se cargan, objetos, etc. Se observa también la pestaña 'History', que permite acceder al historial de órdenes y comandos que se han utilizado en las diferentes sesiones. Además, también se puede importar un *dataset* (conjunto de datos) o bien, si se selecciona el icono , limpiar todos los objetos del entorno.
- En la parte inferior derecha se observa el contenido del directorio 'Home', que es donde arranca R por defecto. Notar que existen cinco pestañas en esta ventana:
 1. **Files:** archivos del directorio de trabajo.
 2. **Plots:** se muestran los gráficos que se generan durante la sesión.
 3. **Packages:** lista de los paquetes/librerías que se tienen descargadas, así como también las disponibles para descargar y la que se está utilizando (marcada con un ✓)
 4. **Help:** a través de esta pestaña, accedemos directamente a la ayuda de R. Podemos introducir directamente en la barra de búsqueda la función sobre la cual se necesite más información.
 5. **Viewer:** accedemos al contenido web local.
- En la parte izquierda, aparece la **consola de R**, es decir, donde se introducen y ejecutan todos los comandos de R. Se escribirá detrás del *prompt* (símbolo >) y para realizar el cálculo pulsamos Intro (Al igual que con R). Aparecerá en la siguiente línea el resultado:


```

> 2+2
[1] 4
> sqrt(2)
[1] 1.414214
> x<-5; y<-6
> x
[1] 5
> y
[1] 6
> x+y
[1] 11
> x*y
[1] 30
> v<-c(1:10)
> v
[1] 1 2 3 4 5 6 7 8 9 10
> |

```

Figura 9.13 Consola RStudio
Fuente: Elaboración propia

Tal como se comentaba en el apartado 9.2.3, resulta más cómodo trabajar sobre *scripts* o ficheros separados de la consola, y directamente desde ellos

ejecutar las líneas de código. Para crear un *script* nuevo, bien seleccionamos en el menú File/New File/New Script o bien, sobre el icono  tal como se observa en la siguiente imagen:

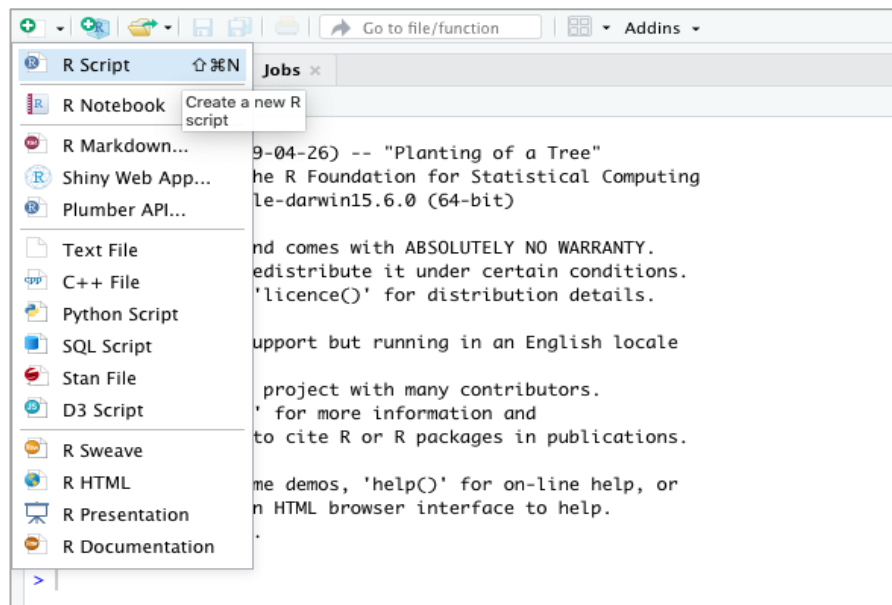


Figura 9.14 Crear Script con RStudio
Fuente: Elaboración propia

También se puede abrir un *script* ya creado en otra sesión (File/Open File). Una vez creado, las líneas de código pueden ejecutarse situando el cursor en la línea que queramos y pulsando Ctrl+Intro o Cmd+Intro.

Así pues, para poder utilizar RStudio tan solo tenemos que instalarlo y utilizar las funciones y órdenes de R (Apartado 2.2.3). La principal diferencia radica en que RStudio resulta más fácil de manejar a la par que más amigable, ya que está pensado para facilitar al usuario el uso del lenguaje R.

[\(Volver a la lectura\)](#)

10 ANEXO 2: CONTENIDO CURRICULAR Y APLICACIÓN EN RSTUDIO

El currículo de las matemáticas regula y detalla los contenidos que se deberían impartir y adquirir a lo largo de cada uno de los cursos y de las diferentes modalidades de la asignatura de matemáticas: en 4º de la ESO, orientadas a las enseñanzas académicas y orientadas a las enseñanzas aplicadas, y en 1º y 2º de Bachillerato, científicas y aplicadas a las ciencias sociales.

En los documentos relativos a cada curso y modalidad, se definen cinco grandes bloques: Procesos, métodos y actitudes en matemáticas, Números y álgebra, Geometría, Análisis y, Estadística y probabilidad.

El primero de ellos, hace referencia a contenidos transversales y que conciernen al día a día en las aulas, como por ejemplo, la resolución de problemas, selección y ordenación de la información, estrategias de lectura comprensiva, imaginación y creatividad, sentido crítico, autoconcepto positivo, autocontrol de las emociones y automotivación, etc. En el segundo se trabajan conceptos relacionados principalmente con operaciones numéricas y algebraicas, como la explicación de los números reales y su representación en la recta real, logaritmos, resolución de ecuaciones, cálculo con fracciones algebraicas, etc. En el tercer bloque, se hace referencia a la parte geométrica, es decir, la más visual, introduciendo las figuras y cuerpos geométricos, así como también el estudio de rectas y planos en cursos más avanzados. En el bloque de Análisis, se tratan contenidos relativos al análisis de funciones, como puede ser el estudio (continuidad, derivabilidad, máximos, mínimos...) y su representación gráfica. Y por último, el bloque de Estadística y probabilidad, en el cual se van desglosando y viendo los principales parámetros estadísticos, junto con la representación e interpretación de los datos, y el cálculo de probabilidades asociadas a fenómenos aleatorios.

De este modo, los diferentes bloques se pueden trabajar con el software RStudio, aunque este sea principalmente un software estadístico. Por ello, el bloque de Estadística y probabilidad se puede trabajar con más profundidad, incluso en su totalidad, utilizando el programa. Aun así, RStudio permite trabajar también contenidos de otros bloques bien con funciones incorporadas o mediante el uso de paquetes externos que a continuación se indicarán.

10.1 Paquetes de R específicos

10.1.1 Paquete Ryacas

El paquete de R *Ryacas* fue creado para poder hacer uso de cálculo simbólico. Como se indica en el mismo manual del paquete (añadir nota pie de página con <https://cran.r-project.org/web/packages/Ryacas/Ryacas.pdf>), este permite al usuario utilizar el programa o lenguaje Yacas (Yet Another Computer Algebra System) desde la consola de R o RStudio. Yacas tiene su propia sintaxis, aunque en

general, es muy similar a la de R. Para información más detallada se puede visitar la web: <http://www.yacas.org/>

Para utilizar este paquete en R, en primer lugar, se deberá instalar y seguidamente ejecutar la librería. La función principal es `yacas`, a la cual se le pasa como argumento una expresión de R creada con el comando `expression()` que incluirá variables simbólicas junto con la función de Yacas que se pretenda ejecutar: cálculo de límites, de derivadas, simplificar fracciones, etc. Es recomendable definir las variables como simbólicas con `Sym("var")`.

Por ejemplo, si se pretende simplificar una expresión algebraica, primero definimos esta expresión a través de la función `expression()` y dentro de esta se aplica la función de Yacas para simplificar, 'Simplify':

```
exp=expression(Simplify((a^3*b)/(b*a^2)))
```

Una vez definida, para obtener el resultado, utilizamos la función principal `yacas()`:

```
yacas(exp)
```

Las funciones propias del lenguaje Yacas que se usarán:

| | Yacas | Argumentos |
|----------------------------|--|--|
| Operaciones con logaritmos | <code>LnExpand(expr)</code>
<code>LnCombine(expr)</code> | <i>expr</i> : expresión para operar |
| Simplificar | <code>Simplify(expr)</code> | - |
| Simplificar con radicales | <code>RadSimp(expr)</code> | - |
| Factorizar | <code>Factor(expr)</code> | - |
| Desarrollar expresiones | <code>Expand(expr, var)</code> | <i>expr</i> : expresión para operar
<i>var</i> : variable |
| Resolver ecuaciones | <code>Solve(expr, var)</code> | - |
| Límite | <code>Limit(expr, var, n)</code>
<code>Limit(expr, var, n, right/left)</code> | <i>expr</i> : expresión para operar
<i>var</i> : variable
<i>n</i> : número al que tiende la variable
<i>right/left</i> : límites laterales |
| Derivar | <code>D(var) (expr)</code> | <i>expr</i> : expresión para operar
<i>var</i> : variable |

| | | |
|----------------------------|---|--|
| Primitiva | <code>Integrate(expr, var)</code> | - |
| Integral definida | <code>Integrate(expr, var, a, b)</code> | <i>expr</i> : expresión para operar
<i>var</i> : variable
<i>a</i> : límite inferior
<i>b</i> : límite superior |
| Convertir a lenguaje LaTeX | <code>TeXForm(expr)</code> | <i>expr</i> : expresión para operar |

A través de la consola de R o de RStudio se puede consultar ayuda en lo relativo a yacas con `?yacas` o `?yacasTranslations`. Antes de utilizar las funciones del paquete, hay que instalarlo y ejecutar la línea de código: `library("nombre_del_paquete")`.

10.1.2 Paquete Polynom

Este paquete de R permite trabajar solo con polinomios y con funciones propias de estos. De este modo, se puede crear un polinomio de cualquier grado y realizar sobre él cualquier operación, como por ejemplo, resolver una ecuación polinómica, derivar un polinomio, evaluarlo en un punto o en un conjunto de puntos, dibujar la gráfica, etc.

La función principal es la que permite crear un polinomio, y es: '`polynomial(coef=c())`'. Es importante destacar que el polinomio se construye a partir de los coeficientes '`coef[1:n]`', de forma que el polinomio resultante es:

$$coef[1] + coef[2] \cdot x + coef[3] \cdot x^2 + \dots + coef[n] \cdot x^{n-1}$$

Las funciones que se utilizarán son principalmente:

| | Polynom | Argumentos |
|---------------------------------|--|--|
| Crear polinomio | <code>polynomial(coef=c())</code> | <i>coef</i> : coeficientes del polinomio |
| Resolver ecuación (igualar a 0) | <code>solve(polynomio)</code> | |
| Derivar | <code>deriv(polynomio)</code> | |
| Integrar | <code>integrate(polynomio, a, b)</code> | <i>a</i> : límite inferior
<i>b</i> : límite superior |
| Mínimo Común Múltiplo (m.c.m) | <code>LCM(polynomio1, polynomio2)</code> | |
| Máximo Común Divisor (M.C.D) | <code>GCD(polynomio1, polynomio2)</code> | |
| Dibujar | <code>plot(polynomio)</code> | |

| | | |
|--------------------------------|---|-------------------------------|
| Crear a partir de raíces dadas | <code>poly.from.roots(raíces=c())</code> | raíces: vector con las raíces |
| Evaluar en un punto/s | <code>predict(polynomio, punto/s)</code> | |

Existen muchas más funciones que se pueden consultar bien en el manual del paquete (<https://cran.r-project.org/web/packages/polynom/polynom.pdf>) o bien en la ayuda de R o RStudio. Antes de utilizar las funciones del paquete, hay que instalarlo y ejecutar la línea de código : `library("nombre_del_paquete")`.

10.1.3 Paquete Matlib

Tal como se describe en el manual del paquete *Matlib*, se usa para trabajar conceptos relacionados con el álgebra lineal utilizando de base el álgebra matricial. Principalmente se usará para trabajar con sistemas y con matrices.

Las funciones que se utilizarán:

| | Matlib | Argumentos |
|--------------------------------|---|---|
| Adjunto de un elemento | <code>cofactor(A, i, j)</code> | A: matriz
i: índice de fila
j: índice de columna |
| Menor complementario | <code>minor(A, i, j)</code> | A: matriz
i: índice de fila
j: índice de columna |
| Determinante | <code>Det(A, method=c("elimination", "cofactors"), verbose=TRUE/FALSE, fractions=TRUE/FALSE)</code> | A: matriz
method: método de cálculo
verbose: opción que permite mostrar los pasos (TRUE) o no (FALSE) |
| Resolver sistema | <code>Solve(A, b, fractions=TRUE)</code> | A: matriz de coeficientes del sistema
b: vector de términos independientes |
| Mostrar ecuaciones | <code>showEqn(A, b)</code> | A: matriz de coeficientes del sistema
b: vector de términos independientes |
| Dibujar ecuaciones del sistema | <code>plotEqn(A, b)</code>
<code>plotEqn3d(A, b)</code> | A: matriz de coeficientes del sistema
b: vector de términos independientes |

| | | |
|----------------------|---|--|
| Escalonar una matriz | <code>echelon(A, verbose=TRUE/FALSE)</code> | A: matriz
verbose: opción que permite mostrar los pasos (TRUE) o no (FALSE) |
| Método de Gauss | <code>gaussianElimination(A, b)</code> | A: matriz de coeficientes del sistema
b: vector de términos independientes |

Existen muchas más funciones que se pueden consultar bien en el manual del paquete (<https://cran.r-project.org/web/packages/matlib/matlib.pdf>) o bien en la ayuda de R o RStudio. Antes de utilizar las funciones del paquete, hay que instalarlo y ejecutar la línea de código: `library("nombre_del_paquete")`.

10.2 4º ESO: Matemáticas orientadas a las enseñanzas académicas y a las enseñanzas aplicadas

Revisando los contenidos curriculares de ambas modalidades se ha considerado oportuno unir en el mismo apartado las matemáticas académicas y las aplicadas, pues las diferencias son mínimas.

10.2.1 Bloque: Números y álgebra

Contenido curricular y aplicación en RStudio

1. Logaritmos y representación

Para trabajar las operaciones con logaritmos utilizando las propiedades, bien de forma numérica o de forma simbólica se utilizará el paquete *Ryacas*.

Si se quiere calcular el desarrollo del logaritmo de un número o de una expresión, siempre que sea posible, se utilizará el comando `LnExpand(expr)`, que utiliza las propiedades de los logaritmos para el cálculo:

```
exp1=expression(LnExpand(log(18)))
exp2=expression(LnExpand(log(x^2)))
yacac(exp1)
yacac(exp2)
```

Y se obtiene,

```
expression(log(2) + 2 * log(3))
expression(2 * log(x))
```

Es decir, internamente se han aplicado las propiedades de los logaritmos $\log(18) = \log(2 \cdot 9) = \log(2 \cdot 3^2) = \log(2) + \log(2^3) = \log(2) + 2\log(3)$ y $\log(x^2) = 2\log(x)$

Si el cálculo es con una combinación de logaritmos, se utilizará *LnCombine(expr)*, que descompone y agrupa los logaritmos y por último, realiza las operaciones indicadas:

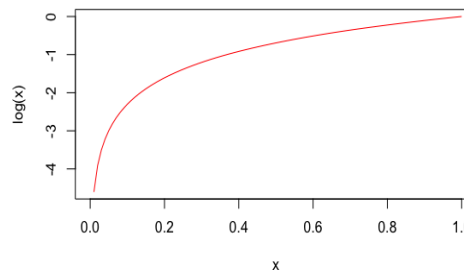
```
exp3=expression(LnCombine(log(3)-log(12)+log(18)))
yacas(exp3)
```

Y se obtiene el resultado del cálculo,

```
expression(log(9/2))
```

Para la representación gráfica, se utilizará la función *curve(función_representar, 'desde', 'hasta', type="", col="",...)*¹⁴:

```
curve(log(x), 0, 1, col="red")
```



2. Cálculo con potencias y radicales

El programa RStudio, no solo nos permite utilizar funciones o comandos, sino que también es una potente calculadora. Por ello, se pueden realizar cálculo de todo tipo. En este caso particular, estamos interesados en potencias y radicales.

Recordar que, para elevar un número o una expresión, es conveniente poner entre paréntesis la base de la potencia y también el exponente, que se introduce mediante el símbolo ' ^ '. Para los radicales, simplemente hay que utilizar la función *sqrt()*, o bien utilizar la forma de exponente fraccionario. De este modo, podemos realizar operaciones sencillas y más complejas, siempre y cuando utilicemos de forma correcta los paréntesis:

```
( (-6)^2 * (5/2)^3 * (2)^(-3) ) / ( (-6)^4 * (5/2)^(2) * (2)^(2) )
[1] 0.002170139
```

Si estamos interesados en la fracción, se puede utilizar la función *Simplify(expr)* del paquete *Ryacas*:

¹⁴ Por defecto, la función *curve* dibuja *type="l"*.

```
exp=expression(Simplify(( (-6)^2 * (5/2)^3* (2)^(-3) ) /
                      ((-6)^4 * (5/2)^(2) * (2)^(2) )))
yacas (exp)
```

Y obtenemos,

```
expression(5/2304)
```

Es decir, $\frac{(-6)^2(5/2)^3 2^{-3}}{(-6)^4(5/2)^2 2^2} = \frac{5}{2304}$

No obstante, si lo que se pretende es llegar a la forma simplificada operando con potencias sin llegar al cálculo del número exacto, conviene hacerlo a mano.

En referencia a los radicales, podemos hacer uso de lo anterior intercambiando las potencias bien por un exponente fraccionario o bien por una raíz con el comando *sqrt()*. Asimismo, dentro del paquete *Ryacas*, existe una función específica si trabajamos directamente con raíces, que nos permite simplificar expresiones, *RadSimp(expr)*:

```
exp_rad_simp=expression(RadSimp(Sqrt(5+2*Sqrt(6)) /
                               Sqrt(5- 2*Sqrt(6))))
yacas (exp_rad_simp)
```

Y se obtiene,

```
expression(root(24, 2) + 5)15
```

Es decir, $\frac{\sqrt{5+2\sqrt{6}}}{\sqrt{5-2\sqrt{6}}} = \sqrt{24} + 5$.

3. Expresiones algebraicas

Para poder resolver ecuaciones, primero necesitamos aprender a operar con polinomios y a desarrollar o simplificar expresiones algebraicas. Utilizaremos los mismos comandos explicados anteriormente, pero en lugar de introducir números, introduciremos variables (x,y,a,b...):

```
exp=expression(Simplify((a^3*b) / (b*a^2)))
yacas (exp)
```

Obtenemos,

```
expression(a)
```

¹⁵ En *root(n,m)* el primer valor hace referencia al radicando y el segundo al índice

Es decir, $\frac{a^3b}{ba^2} = a$.

También utilizando *RadSimp(expr)* podemos simplificar expresiones algebraicas con radicales. Por último, el comando *Expand(expr,var)*, desarrolla el polinomio que se le pasa como expresión, en función de la variable que se le indica, en nuestro caso siempre será 'x'. De este modo, podemos desarrollar las igualdades notables o cualquier otra operación:

```
p=expression(Expand((a-x)*(b-x),x))
p1=expression(Expand((x-2)^4),x)
yacas(p)
yacas(p1)
```

Y el resultado es,

```
expression(x^2 - (b + a) * x + a * b)
expression(x^4 - 8 * x^3 + 24 * x^2 - 32 * x + 16)
```

O sea, $(a - x)(b - x) = x^2 - (b + a)x + ab$ y
 $(x - 2)^4 = x^4 - 8x^3 + 24x^2 - 32x + 16$

4. Polinomios y raíces

Para trabajar con polinomios, existe el paquete externo *Polynom* que permite realizar cálculos básicos con polinomios, por tanto, será de utilidad para tratar este bloque. En primer lugar, creamos el polinomio con la función *polynomial(coef=c(,))* de la siguiente forma:

```
p=polynomial(coef=c(-2,1,1)) #-2+x+x^2
p
-2 + x + x^2
```

Notar que la notación es al revés que a la que acostumbramos, es el último término del vector de coeficientes el que se asigna a la x de mayor grado, hasta llegar al término independiente. Es decir, si queremos introducir el polinomio $p(x) = 2x^3 - x^2 - 5$, el vector de coeficientes será:

```
coef=c(-5,3,0,2)16
```

Una vez tenemos el polinomio definido, podemos realizar las operaciones de adición, sustracción y multiplicación, bien con otro u otros polinomios o con números. Por ejemplo,

¹⁶ Si hay algún grado de la x que no aparece en el polinomio, incluimos un 0 en el vector de coeficientes.

```

p1=polynomial(coef=c(-3,2,-5))
p2=polynomial(coef=c(2,5,-1))
p1+p2
1 + 7*x - 6*x^2
p1*p2
6 - 11*x + 3*x^2 - 27*x^3 + 5*x^4

```

La división de polinomios es un asunto más complicado para el paquete, ya que tan solo muestra el cociente de la división.

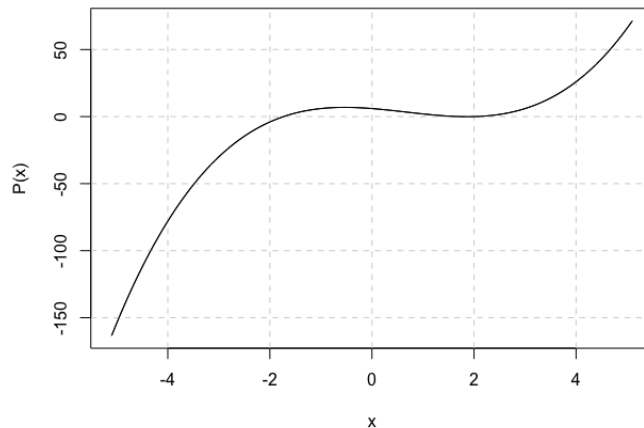
Otro aspecto importante relacionado con los polinomios es la factorización, y por tanto el cálculo de raíces de un polinomio, que no es más que la solución de la ecuación $p(x) = 0$, siendo $p(x)$ un polinomio. Para ello, utilizaremos la instrucción `solve(polynomio)`, y así obtendremos las raíces de este. El único inconveniente que presenta, es que las raíces se obtienen en forma decimal:

```

p3=polynomial(coef=c(6,
-3,-2,1))

raices=solve(p3)
raices
[1]-1.732051 1.732051
2.000000

```



Es decir, estamos calculando: $p(x) = x^3 - 2x^2 - 3x + 6 = 0$, y al tratarse de una ecuación de tercer grado, obtenemos tres soluciones. En el caso de ser un polinomio con soluciones imaginarias, también se proporcionan.

Igualmente resulta de utilidad evaluar el polinomio en algún punto para saber si se anula o no en tal punto, o tan solo para saber el valor. Se hará mediante la instrucción `predict(polynomio, c(valores))`:

```

predict(p3,c(0,2,1))
[1] 6 0 2

```

Es decir, evaluamos el polinomio 'p3' en $x = 0$, $x = 2$, $x = 1$, y efectivamente se comprueba como $x = 2$ es raíz del polinomio, pues el valor asociado es 0.

Por otro lado, aunque no se hace especial énfasis, podemos dibujar el polinomio, con la función genérica `plot(polynomio)`:

```

plot(p3,xlim=c(-5,5)) #xlim indica el intervalo de la x en el
cual dibujar

```

Para trabajar con expresiones algebraicas que incluyan sumas o restas con denominadores algebraicos polinómicos, es necesario calcular el mínimo

común múltiplo de estos. A través de este paquete se puede llevar a término de la siguiente manera: definimos los polinomios de los que queremos saber el mínimo común múltiplo,

```
p1=polynomial(c(-1,0,1))
p2=polynomial(c(1,1))
```

y seguidamente, aplicamos la función *LCM* (*pol1,pol2...*):

```
LCM(p1,p2)
```

Y se obtiene:

```
-1 + x^2
```

Es decir, hemos calculado el *m. c. m.* $m(x^2 - 1, x + 1) = x^2 - 1$

Por último, hay que destacar un comando útil sobretodo para profesores, pues permite crear un polinomio a partir de las raíces que se le indican. Se lleva a cabo mediante la instrucción *poly.from.roots*(*c(raíces)*):

```
poly.from.roots(c(1,-2,3))
6 - 5*x - 2*x^2 + x^3
```

Es decir, indicamos que queremos un polinomio $p(x)$, que tenga por raíces $x = 1, x = -2, x = 3$. Y así, obtenemos que dicho polinomio es $p(x) = x^3 - 2x^2 - 5x + 6$.

5. Ecuaciones y sistemas

Como se ha visto, es posible resolver ecuaciones polinómicas en RStudio, así pues, también lo es resolver sistemas de ecuaciones lineales. No obstante, para poder usar el paquete de R se necesita conocer el concepto de matriz, por tanto, esto quedará recogido en los cursos de bachillerato.

Cabe indicar también, que los sistemas de ecuaciones no lineales no pueden solucionarse con el software que utilizamos.

10.2.2 Bloque: Geometría

Contenido curricular y aplicación en RStudio

1. Trigonometría

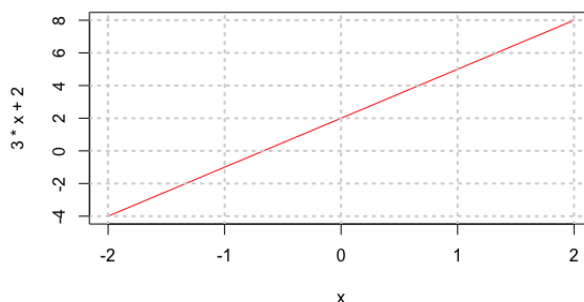
Para trabajar este contenido, RStudio se utilizaría como mera calculadora de razones trigonométricas de un ángulo α , a través de las funciones: $\sin(\alpha)$, $\cos(\alpha)$, $\tan(\alpha)$, $\text{asin}(\alpha)$, $\text{acos}(\alpha)$ y $\text{atan}(\alpha)$.

2. Ecuaciones de la recta. Paralelismo y perpendicularidad

En cuanto a la representación de rectas para poder visualizar la posición relativa de estas, podemos utilizar la función `curve()`. Tan solo debemos indicar la función a dibujar, y el dominio de definición, es decir, los valores de la x , siguiendo `curve(función_representar, 'desde', 'hasta', type="", col="",...)`

En la función `curve()` podemos añadir opciones del tipo de línea (`type`, que por defecto es `"l"`), color (`col`), grosor (`lwd`)... :

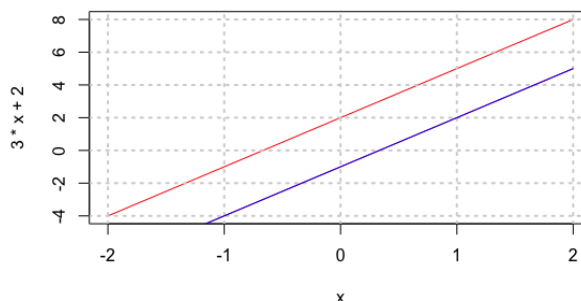
```
curve(3*x+2, -2, 2, col="red")
grid(lwd=2)
```



El comando `grid()` con la opción `lwd` lo utilizamos para dibujar la cuadrícula en nuestro gráfico con el grosor (`lwd`) que determinamos.

Para añadir otra recta, utilizamos la misma instrucción, pero debemos añadir al final de esta `"add=TRUE"`, de esta forma, la nueva recta o curva dibujada se añadirá al dibujo. Es recomendable cambiar el color para poder distinguirlas:

```
curve(3*x-1,
-
2, 2, col="blue", add=TRUE)
```



Y así sucesivamente con todas las rectas que queramos añadir a nuestro gráfico. De este modo, podemos observar si las rectas se cortan o son paralelas.

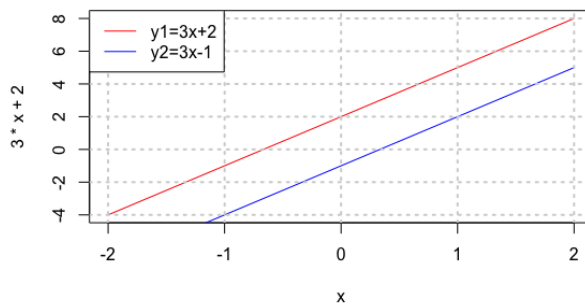
Quizás pueda resultar útil añadir una leyenda al gráfico para identificar las rectas. Se utiliza la instrucción `legend()`, y en ella debemos indicar:

- Posición de la leyenda: bien por coordenadas (x,y), o indicando `"topright"`, `"topleft"` para las esquinas superiores o `"bottomright"`, `"bottomleft"` para las esquinas inferiores; `"bottom"` o `"top"` para centrarlas superiormente o inferiormente y `"center"` para añadirla al centro del gráfico.
- Texto de la leyenda: se añade escribiendo `legend=c("función1", "función2",...)`.

- Color asociado a cada recta: ha de corresponderse con el color que hemos indicado para dibujarla, `col=c("red","blue")`.
- Estilo de la línea: para indicar el dibujo de línea en la cuadrícula, escribimos `lty=c()`; incluyendo dentro un 1 que indica una línea normal, un 2 una línea con rayas separados o un 3 una línea con puntos separados por cada recta que se dibuje.
- Muchas más opciones estilísticas, para obtener más se puede consultar la ayuda de RStudio, escribiendo en la consola `?legend`.

Así, la leyenda en nuestro gráfico será:

```
legend("topleft",
legend=c("y1=3x+2",
"y2=3x-1"),
col=c("red","blue"),
lty=c(1,1))
```



10.2.3 Bloque: Análisis

Contenido curricular y aplicación en RStudio

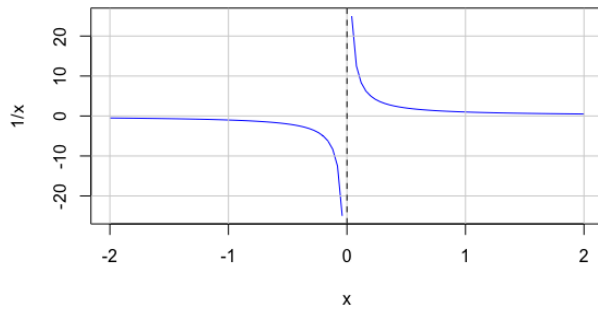
1. Funciones de proporcionalidad inversa, exponenciales, logarítmicas

En relación con el bloque de análisis, lo más importante es la representación e identificación de los diferentes tipos de funciones. En este curso se trabajan principalmente las de proporcionalidad inversa, las exponenciales y las logarítmicas. Estas últimas ya se explican en un apartado anterior.

Para la representación, se procede igual que en los casos anteriores, primero se define la variable x , y seguidamente la función en sí con el nombre de variable y . Es importante tener en cuenta en todo momento el dominio de definición de las funciones, pues si estamos ante una función con logaritmos el dominio no incluye números negativos.

- Función de proporcionalidad inversa:

```
curve(1/x, -2, 2,
      col="blue")
grid(lwd=1, lty=1)
```



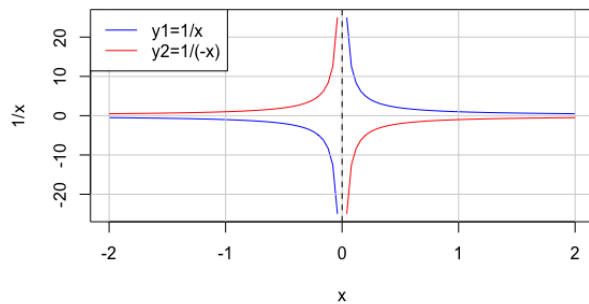
Además, es importante destacar que estas funciones tienen una asíntota vertical en $x = 0$, que conviene dibujar mediante el comando `abline (v= x,lty=2)`, donde x en este caso es 0, y con `lty=2` indicamos que la línea sea discontinua:

```
abline(v=0, lty=2)
```

También podemos añadir más funciones para compararlas, junto con una leyenda:

```
curve(1/(-x),
      -2, 2, col="red", add=TRUE)

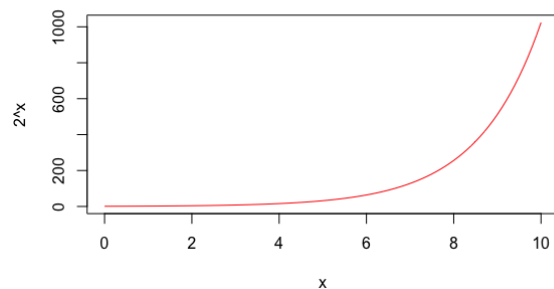
legend("topleft",
      legend=c("y1=1/x",
              "y2=1/(-x)"),
      col=c("blue", "red"),
      lty=c(1, 1))
```



- Función exponencial

De igual forma, para la función exponencial procedemos de la misma forma. Si la base es un número natural, tan solo escribimos la función como una potencia:

```
curve(2^x, 0, 10, col="red")
```

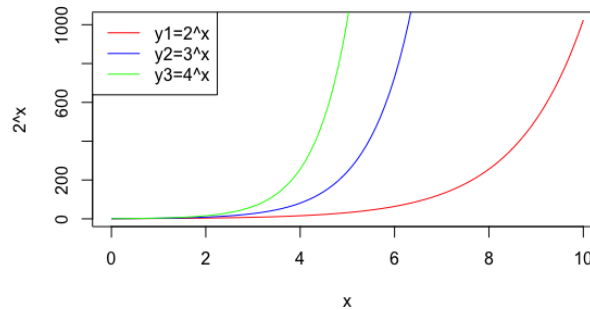


De igual forma, si queremos comparar diferentes exponenciales añadimos a las instrucciones siguientes `add=TRUE`:

```

curve(3^x, 0,10,col="blue",lty=1,add=TRUE)
curve(4^x, 0,10,col="green",lty=1,add=TRUE)
legend("topleft",legend=c("y1=2^x","y2=3^x","y3=4^x"),
col=c("red","blue","green"),lty=c(1,1))

```



En el caso específico de la función exponencial con base e , escribiremos " $exp(x)$ "

- Rectas y parábolas

En el apartado de polinomios se indica como representarlas, pues una recta es un polinomio de primer grado y una parábola de segundo grado.

10.2.4 Bloque: Estadística y probabilidad

Contenido curricular y aplicación en RStudio

1. Tabla de frecuencias

En primer lugar, se deben introducir los datos o nuestra variable. Si son datos procedentes de un ejercicio se pueden introducir como un vector ¹⁷ :

```

variable_años=c(13,15,14,16,13,15,14,16,15,14,13,13,13,15,14,16,
14, 14)
N=length(variable_años) #nos indica el número de datos que tenemos

```

Una vez introducidos, con el comando `table(nombre_variable)`, obtenemos las frecuencias absolutas y con `prop.table(table(nombre_variable))` las frecuencias relativas:

```

table(variable_años)
datos
13 14 15 16
5 6 4 3

prop.table(table(variable_años))
datos
13 14 15 16
0.2777778 0.3333333 0.2222222 0.1666667

```

¹⁷ Ver apartado del anexo 9.2.3-Vectores para ver como definir datos.

Es decir, obtenemos que la frecuencia absoluta de 14 años es 6, y la relativa 0.333, es decir, un 33% de la muestra tiene 14 años.

Si la variable es de carácter cualitativo se procede de igual manera¹⁸ :

```
variable_sexo=c(0,1,0,0,1,0,0,1,1,0)
variable_sexo=factor(variable_sexo, levels=c(0,1),
                      labels=c("Hombre", "Mujer"))
#recodificamos las categorías de la variable

table(variable_sexo)
variable_sexo
Hombre  Mujer
      8     10
prop.table(table(variable_sexo))
variable_sexo
      Hombre      Mujer
0.4444444 0.5555556
```

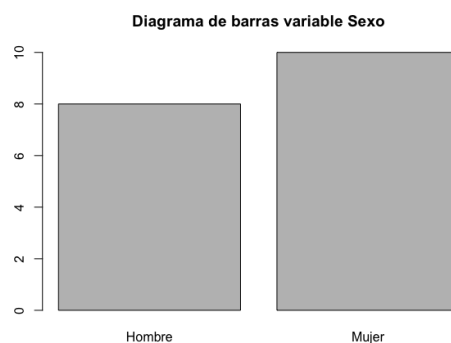
Observamos que el 44% de la muestra son hombres y el 55% restante mujeres.

2. Gráficos estadísticos

- Diagrama de barras

Este tipo de gráfico se asocia a variables cuantitativas discretas o cualitativas. Para dibujar este gráfico, se utiliza la función `barplot(table(nombre_variable))`, es decir, se pasa como argumento de la función las frecuencias absolutas. En este caso utilizaremos la variable 'sexo' por ser cualitativa:

```
barplot(table(variable_sexo), main="Diagrama de barras variable
Sexo")
```



Notar que con el argumento `main=" Título "`, se escribe el título del gráfico. En el eje Y aparece la frecuencia absoluta.

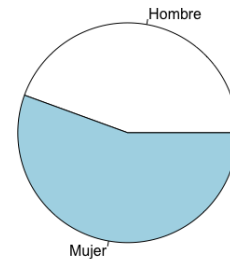
¹⁸ Ver apartado del anexo 9.2.3-Vectores para ver como definir una variable de tipo factor o cualitativa.

- Diagrama de sectores

El diagrama de sectores se suele utilizar en los mismos casos que el gráfico anterior. Para dibujarlo debemos introducir el comando `pie(table(nombre_variable))`:

```
pie(table(variable_sexo), main="Diagrama de sectores variable Sexo")
```

Diagrama de sectores variable Sexo



Podemos cambiar los colores, la forma y otros aspectos. Para ello se puede consultar la ayuda de R para dicha función.

- Histograma

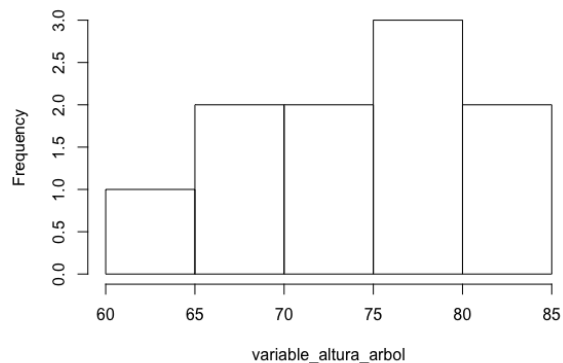
Este gráfico se usa para representar variables continuas, pues los valores se agrupan por intervalos y se levantan barras según la frecuencia absoluta que tenga. Para dibujarlo utilizamos `hist(nombre_variable)`:

```
variable_altura_arbol=c(70.4, 65.2, 63.4, 72.5, 81, 83, 66.3, 75.1, 80, 75.7, 79.9, 76.7, 76.9, 69.4, 75.1, 74, 85.8, 86.9, 71.5, 64.6, 78.8, 80.4, 74.3)

hist(variable_altura_arbol, main="Histograma variable Altura")
```

Dentro de la función `hist()` hay gran cantidad de opciones que se pueden modificar, como el número de intervalos, los colores, el título de los ejes, etc. Con `?hist` (ayuda de R para la función) podemos consultar todos estos aspectos.

Histograma variable Altura

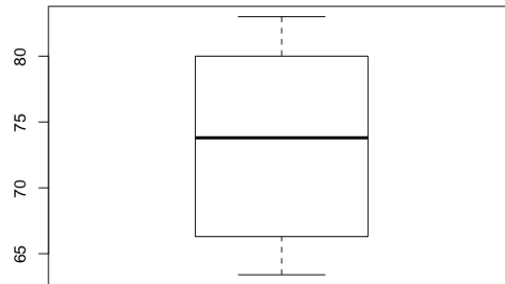


- Diagrama de caja o boxplot

Este diagrama suele utilizarse en variables numéricas, principalmente para comparar varias variables o una misma variable en distintos grupos. Además, muestra una simplificación de la distribución de la variable, pues solo se representan las medidas de posición. Para dibujar este gráfico, utilizaremos `boxplot(nombre_variable)`:

Diagrama de caja variable Altura

```
boxplot(variable_altura_arbol,
main="Diagrama de caja
variable Altura")
```



3. Medidas de centralización, posición y dispersión

- Centralización: por un lado tenemos la media, que para calcularla lo haremos escribiendo `mean(nombre_variable)`; la mediana que utilizaremos `median(nombre_variable)` y por último la moda, que aunque no existe una función específica, podemos obtenerla a través de la tabla de frecuencias absolutas:

```
media=mean(variable_altura_arbol)
media
[1] 73.26
mediana=median(variable_altura_arbol)
mediana
[1] 73.8
```

- Posición: entre estas medidas encontramos los cuartiles y los percentiles. Se utiliza la función `quantile(nombre_variable)` para los cuartiles o también `quantile(nombre_variable, prob=seq(0,1,0.25))` y para los percentiles `quantile(nombre_variable, prob=seq(0,1,0.01))`. Con `prob=seq(0,1,0.25)` indicamos que dividimos en 4 partes la muestra (cuartil), y con `prob=seq(0,1,0.01)` en 100 partes (percentil):

```
quantile (variable_altura_arbol) #Cuartiles
0% 25% 50% 75% 100%
63 72 76 80 87

quantile (variable_altura_arbol, prob=seq(0,1,0.01))
#Percentiles
0% 1% 2% 3% 4% 5% 6% 7% 8% 9% 10% 11% 12%
63.0 63.3 63.6 63.9 64.2 64.5 64.8 65.1 65.4 65.7 66.0 66.9 67.8
.....
89% 90% 91% 92% 93% 94% 95% 96% 97% 98% 99%
100%
82.7 83.0 83.6 84.2 84.8 85.2 85.5 85.8 86.1 86.4 86.7 87.0
```

Un comando realmente útil es `summary(nombre_variable)` o `summary(conjunto_datos)` en caso de tener un conjunto de datos con más de una variable, pues nos indica el valor máximo y mínimo, los cuartiles, y la media de la variable que se le indica o de cada una de las variables que forman el conjunto de datos.

- **Dispersión:** como medidas de dispersión existe el rango, $range(nombre_variable)$ que indica el valor mínimo y máximo; la varianza $var(nombre_variable)$, la desviación típica $sd(nombre_variable)$. Para el coeficiente de variación, aunque no existe fórmula directa, se calcula dividiendo la desviación típica entre la media, $sd()/mean()*100$, para expresarlo en porcentaje:

```
rango=range(variable_altura_arbol)
rango
[1] 63.4 80

varianza=var(variable_altura_arbol)
varianza
[1] 47.65822

desvtipica=sd(variable_altura_arbol)
desvtipica
[1] 6.903493

coef.var=desvtipica/media*100
coef.var
[1] 9.423278
```

4. Probabilidad

En este nivel, aun no se trabaja con distribuciones de probabilidad. Por tanto, RStudio se puede utilizar como calculadora para las probabilidades. Incluso se pueden escribir números combinatorios con $choose(n,m)$ que indica el número combinatorio n sobre m.

10.3 1º Bachillerato: Matemáticas científicas y aplicadas a las CCSS

10.3.1 Bloque: Números y álgebra

Contenido curricular y aplicación en RStudio

Algunos contenidos como los relativos a operaciones combinadas con potencias y radicales, polinomios y sus raíces, o resolución de ecuaciones polinómicas los comandos a utilizar son los explicados en el mismo bloque de números y álgebra del curso 4.º ESO. Por tanto, solo se incluirán los contenidos que requieran una explicación nueva.

1. Ecuaciones exponenciales y logarítmicas de base 'e'

Para la resolución de este tipo de ecuaciones, se hará uso del paquete *Ryacas* y del comando $Solve(expr, var)$. Es importante tener en cuenta, que la ecuación que resuelve es ' $expr = 0$ ', es decir, que la expresión que debemos introducir es la resultante de dejar a uno de los lados de la igualdad un 0; y a continuación la

variable respecto de la cual se quiere resolver la ecuación. Por ejemplo, para resolver la ecuación $e^x = 1$, despejamos $e^x - 1 = 0$ y esta expresión es la que introducimos:

```
eq=expression(Solve(exp(x)-1,x))
yacas(eq)
```

Y obtenemos la solución:

```
Yacas vector:
[1] x == 0
```

Por ejemplo, la ecuación $\ln(x) + \ln(20) = 3$,

```
eq=expression(Solve(log(x)+log(20)-3,x))
yacas(eq)
Yacas vector:
[1] x == exp(3 - log(20))
```

tiene por solución: $x = e^{3-\ln 20}$.

2. Sistemas de ecuaciones lineales con dos y tres incógnitas

Aunque en el currículo de las matemáticas no se establezca para el primer curso de bachillerato la noción de matriz, es este caso resulta necesaria para poder resolver sistemas de ecuaciones, pues la función que se va a necesitar utiliza como argumento la matriz de coeficientes del sistema ¹⁹. Para ello, se hará uso del paquete 'Matlib' de R.

En el caso de los sistemas con dos ecuaciones lineales y dos incógnitas, la matriz es de dimensión 2x2. La función específica del paquete es $Solve(A, b)$ siendo A la matriz de coeficientes, y b el vector de términos independientes.

De este modo, para resolver el sistema:

$$\begin{cases} x - 2y = -4 \\ 2x - y = 1 \end{cases}$$

en primer lugar, separamos el sistema en la forma $Ax = b$:

$$\begin{pmatrix} 1 & -2 \\ 2 & -1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -4 \\ 1 \end{pmatrix}$$

Seguidamente, ya introducimos la matriz en RStudio y aplicamos la función $Solve(A,b)$. Para escribir la matriz, escribimos los números por filas, añadimos el número de columnas y filas (con $nrow$ y $ncol$) y para poder rellenarla por filas es necesario añadir $byrow=TRUE$:

¹⁹ Ver apartado 9.2.3-Matrices del anexo para recordar como definir una matriz

```
library(matlib)
A=matrix(c(1,-2,2,-1),nrow=2,ncol=2,byrow=TRUE)
b=c(-4,1)
```

Podemos ver si el sistema está bien escrito con el comando *showEqn(A,b)*:

```
showEqn(A, b)
1*x1 - 2*x2 = -4
2*x1 - 1*x2 = 1
```

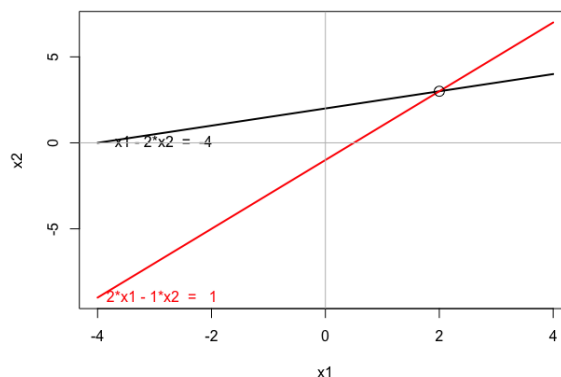
Notar que internamente el programa asocia a la primera variable, en nuestro caso la x el $x1$, y a la y el $x2$.

Resolvemos el sistema:

```
Solve(A, b, fractions = TRUE) #con fractions=TRUE indicamos que
si el resultado es una fracción aparezca como tal
x1 = 2
x2 = 3
```

Además, podemos obtener la representación gráfica del sistema y ver que efectivamente como la solución es un punto, es el punto donde se cruzan ambas rectas. Utilizamos el comando *plotEqn(A,b)*:

```
plotEqn(A, b)
x1 - 2*x2 = -4
2*x1 - 1*x2 = 1
```



Si el sistema es de 3 ecuaciones, la matriz será de dimensión 3×3 , y se procederá de la misma forma:

Por ejemplo, el sistema:

$$\begin{cases} x - 3y + 4z = 21 \\ 3x + y - z = -18 \\ 2x - y + 3z = 12 \end{cases}$$

lo separamos en la forma $Ax=b$,

$$\begin{pmatrix} 1 & -3 & 4 \\ 3 & 1 & -1 \\ 2 & -1 & 3 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 21 \\ -18 \\ 12 \end{pmatrix}$$

y se introduce en RStudio:

```

A=matrix(c(1,-3,4,3,1,-1,2,-1,3),nrow=3,ncol=3,byrow=TRUE)
b=c(21,-18,12)
showEqn(A,b)
1*x1 - 3*x2 + 4*x3 = 21
3*x1 + 1*x2 - 1*x3 = -18
2*x1 - 1*x2 + 3*x3 = 12

Solve(A,b,fractions = TRUE)
x1      = -4
  x2     =  1
  x3     =  7

```

La representación de los sistemas lineales de tres ecuaciones y tres incógnitas se deja para 2º Bachillerato.

Una función interesante de este paquete es la función `echelon(A,b,verbose=TRUE,fractions=TRUE)` que muestra paso a paso como convierte el sistema en un sistema escalonado por el método de Gauss:

```

echelon(A,b,verbose=TRUE,fractions=TRUE)

Initial matrix:
      [,1] [,2] [,3] [,4]
[1,]  1  -3   4  21
[2,]  3   1  -1 -18
[3,]  2  -1   3  12

row: 1

      exchange rows 1 and 2
      [,1] [,2] [,3] [,4]
[1,]  3   1  -1 -18
[2,]  1  -3   4  21
[3,]  2  -1   3  12

      multiply row 1 by 1/3
      [,1] [,2] [,3] [,4]
[1,]  1  1/3 -1/3 -6
[2,]  1  -3   4  21
[3,]  2  -1   3  12

      .....
      multiply row 3 by 13/10 and add to row 2
      [,1] [,2] [,3] [,4]
[1,]  1   0   0  -4
[2,]  0   1   0   1
[3,]  0   0   1   7

```

3. Números complejos

Uno de los tipos de números en el lenguaje R son los números complejos. Estos, que son de la forma $a + bi$, siendo a la parte real y b la parte imaginaria, se escriben igual en lenguaje R. De este modo, mediante los símbolos habituales de

las operaciones podemos realizar sumar, restas, multiplicaciones y divisiones de números complejos:

```
z1=3+2i #Definimos un numero real y lo guardamos como z1
z2=5-3i #Definimos un numero real y lo guardamos como z2
z1+z2
[1] 8-1i
z1*z2
[1] 21+1i
```

Por otro lado, existen funciones específicas que calculan el módulo, la parte real, la parte imaginaria, el argumento y el conjugado; y son $Mod(z)$, $Re(z)$, $Im(z)$, $Arg(z)$ y $Conj(z)$ respectivamente:

```
Mod(z1)
[1] 3.605551
Re(z1);Im(z1)
[1] 3
[1] 2
Arg(z1) #En radianes
[1] 0.5880026
Conj(z1)
[1] 3-2i
```

10.3.2 Bloque: Geometría

Contenido curricular y aplicación en RStudio

En la modalidad de matemáticas aplicadas a las ciencias sociales no existe bloque de geometría. En las matemáticas científicas sí, pero no se han encontrado contenidos que puedan trabajarse con RStudio.

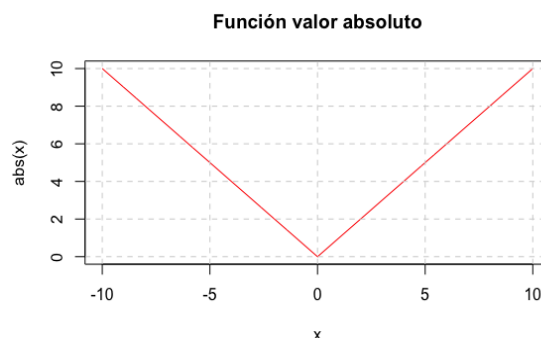
10.3.3 Bloque: Análisis

Contenido curricular y aplicación en RStudio

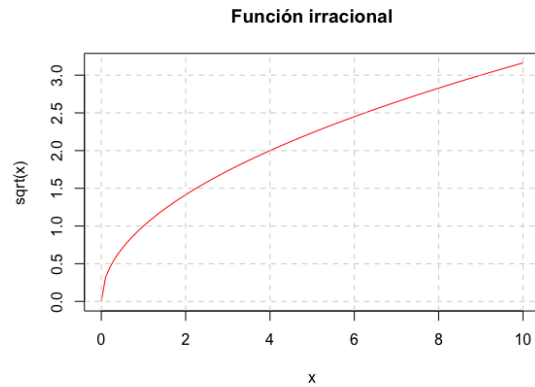
1. Funciones elementales

Las funciones que se incluyen de nuevo en el temario de Bachillerato son las de valor absoluto, las trigonométricas y las irracionales. Para la representación se procede de igual forma a la explicada en apartados anteriores, haciendo uso de la función $curve()$:

```
#Valor absoluto
curve(abs(x), -10, 10, col="red",
main="Función valor absoluto")
grid(lty=2)
```

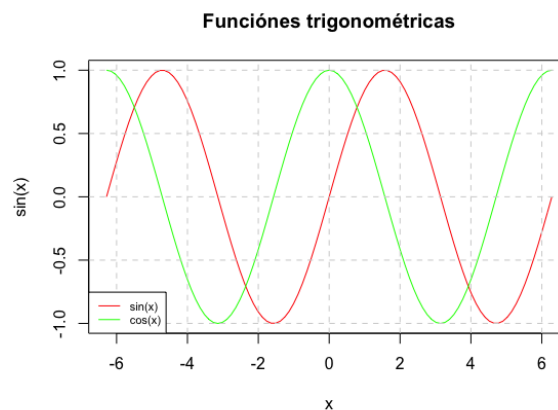


```
#Irrracional
curve(sqrt(x),0,10,col="red",
main="Función irracional")
grid(lty=2)
```



```
#Sin/Cos
curve(sin(x),-2*pi,2*pi,col="red",
main="Funciones trigonométricas")
curve(cos(x),-2*pi,2*pi,col="green",add=TRUE)
grid(lty=2)

legend("bottomleft",legend=c("sin(x)", "cos(x)"),
col=c("red", "green"),lty=c(1,1),cex=0.7)
```



2. Límites

Cuando tratamos con variables simbólicas, es decir, a las que no le asignamos un valor, trabajamos con el paquete *Ryacas*. Así pues, para el cálculo de algunos límites puede ser de utilidad. Para ello, en primer lugar es necesario en este caso indicar que la variable x es simbólica, y seguidamente se hace uso de la función $Limit(expr, var, valor)$ o bien si interesa calcular los límites laterales $Limit(expr, var, valor, right/left)$. Indicar, que el signo cuando se calcula el límite lateral, la función no indica el signo del valor que se obtiene del límite. Para calcular el $\lim_{x \rightarrow 0}(x + 2)$:

```
library("Ryacas")
x=Sym("x")
lim1=yacas(Limit(x+2,x,0))
lim1
expression(2)
```

Evidentemente, obtenemos que el valor de dicho límite es 2.

En el caso de los laterales:

```
lim2=yacas(Limit(1/x,x,0,"Right"))
lim2 #El signo deberá incluirlo el alumno
expression(NaN)
```

Es decir, NaN (*Not A Number*) significa que el valor de dicho límite es ∞ , pero el signo deberemos añadirlo nosotros, en este caso $+\infty$.

3. Derivadas

Cuando tratamos con derivadas también estamos hablando de cálculo simbólico. Así, continuaremos con el uso del paquete *Ryacas* con la función $D(var)$ (*Expr_derivar*):

```
library("Ryacas")
derivada=expression( D(x) (sin (x)) )
yacas(derivada) #obtenemos el resultado
expression(cos(x))
```

Efectivamente $[sin(x)]' = cos(x)$.

10.3.4 Bloque: Estadística y probabilidad

Contenido curricular y aplicación en RStudio

1. Estadística descriptiva bidimensional

En este curso ya se empieza a trabajar de forma más realista la estadística, pues al introducir la bidimensionalidad, es decir, el estudio de un conjunto de datos con dos variables, nos acercamos más a la realidad por lo que a tratamiento de datos respecta; ya que en una base de datos suelen aparecer un gran número de variables. De este modo, estaremos interesados en estudiar la relación entre las dos variables que se analizarán, tanto conjuntamente como individualmente, realizar estimaciones e incluso predicciones. Además, RStudio carga por defecto el paquete 'datasets' que incluye bases de datos reales con diversas variables para poder estudiar las relaciones entre estas.

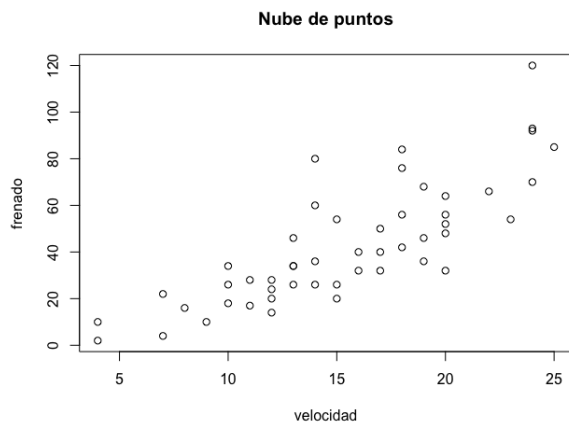
En primer lugar, es necesario escoger las variables y representarlas a través de la nube de puntos, $plot(x, y)$, para poder tratar de ver si existe o no relación entre ellas:

```
#base de datos ' cars'
del paquete datasets
datos=cars

#accedemos a la base de datos
attach(datos)
#View(datos)

#Nombramos a las variables
velocidad=datos$speed #x
frenado=datos$dist #y

#Dibujamos la NUBE DE PUNTOS
plot(velocidad,frenado,
main="Nube de puntos")
```



En este caso, se observa una correlación alta y positiva, pues a más velocidad, más distancia de frenado se necesita. Podemos calcular el coeficiente de correlación para ver que efectivamente así es, mediante $cor(x, y)$:

```
cor(velocidad,frenado) #Coeficiente de correlación
[1] 0.8068949
```

Como la correlación indica la relación entre ambas variables, es decir, marca la tendencia, podemos aproximar la nube de puntos a través de la recta de regresión. Así pues, a través de RStudio podemos calcular el modelo de regresión, estimar la recta y dibujarla sobre la nube de puntos inicial.

En primer lugar, para el modelo de regresión se utiliza la función $lm(y \sim x, data=nuestros_datos)$:

```
regresion=lm(frenado ~ velocidad)
```

Notar que la variable y , en nuestro caso la variable *frenado*, se coloca primero, seguido de la variable x , que en este caso es la variable *velocidad*.

El resultado que se obtiene al realizar la regresión lineal se puede visualizar mediante $summary(regresión)$, no obstante, en nuestro caso tan solo estamos interesados en saber la estimación de los coeficientes de la recta de regresión. Para ello, basta con escribir el nombre de la variable:

```
regresion
Call:
lm(formula = frenado ~ velocidad)

Coefficients:
(Intercept)      velocidad
   -17.579         3.932
```

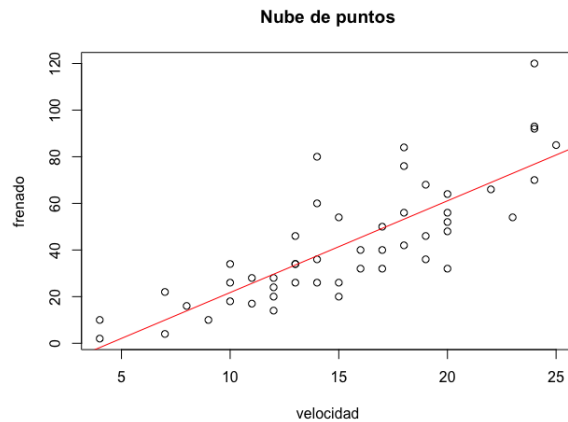
De aquí obtenemos que la recta de regresión es:

$$\text{frenado}(y) = 3,932 * \text{velocidad}(x) - 17,579$$

siendo 3,932 el coeficiente estimado para la variable x , y -17,579 el término independiente de la recta.

Para dibujar la recta, utilizamos `abline(regresión, col="red")`:

```
abline(regresión,  
col="red")
```



Ver que la recta se ajusta de forma adecuada a la nube de puntos que generan los datos, y que por tanto el coeficiente de correlación es coherente.

Otra medida que puede ser de utilidad para determinar si el modelo de regresión explica de forma correcta los datos es el coeficiente de determinación. Para calcularlo, bastará elevar al cuadrado el coeficiente de correlación de Pearson:

```
R2=cor(velocidad,frenado)^2  
R2  
[1] 0.6510794
```

Este coeficiente indica que el modelo de regresión explica el 65% de la variabilidad total de los datos.

Por último, para poder realizar predicciones del valor de la variable y que se obtendría para valores de la variable x , se crea un *dataframe* con los valores de la variable x para los que se desee calcular la predicción, y se utiliza la función `predict.lm(regresión, nuevos_datos)`:

```
datos_pred=data.frame(velocidad=c(36, 50))  
prediccion=predict(regresion, datos_pred)  
prediccion  
      1      2  
123.9876 179.0413
```

Es decir, para una velocidad de 36 mph se estima una distancia de frenado de 123,99 ft; y para una velocidad de 50 mph, una distancia de frenado de 179,04 ft. Por tanto, a más velocidad, más distancia de frenado.

En referencia a las tablas de contingencia para dos variables, se crean mediante `table(variable1,variable2)`, y si queremos la proporción o frecuencias relativas `prop.table(table(variable1,variable2))`.

También podemos calcular las frecuencias marginales, tanto absolutas como relativas, con la función `addmargins(table(var1,var2))` para las absolutas, o `addmargins(prop.table(table(var1,var2)))` para las relativas:

```
addmargins(table(variable_años , variable_sexo))
      variable_sexo
variable_años Hombre Mujer Sum
      13          1     4    5
      14          4     2    6
      15          1     3    4
      16          2     1    3
      Sum          8    10   18

addmargins(prop.table(table(variable_años , variable_sexo)))
      variable_sexo
variable_años  Hombre      Mujer      Sum
      13  0.05555556 0.22222222 0.27777778
      14  0.22222222 0.11111111 0.33333333
      15  0.05555556 0.16666667 0.22222222
      16  0.11111111 0.05555556 0.16666667
      Sum 0.44444444 0.55555556 1.00000000
```

Asimismo, para poder calcular las distribuciones condicionadas utilizamos la función `prop.table(table(),1)` o `prop.table(table(),2)` o también podemos hacerlo con las marginales²⁰. La primera de ellas nos calcula la distribución de la segunda variable condicionada a valores de la primera variable. La segunda, al contrario. Por ejemplo, en la siguiente instrucción,

```
prop.table(table(variable_años , variable_sexo),1)
      variable_sexo
variable_años  Hombre      Mujer
      13          0.2000000 0.8000000
      14          0.6666667 0.3333333
      15          0.2500000 0.7500000
      16          0.6666667 0.3333333
```

indicamos que condicionamos la variable sexo a valores de la variable años. Por último, para calcular los parámetros estadísticos como la media, la varianza o la desviación típica de cada una de las variables, tan solo se ha de aplicar las funciones específicas explicadas con anterioridad a cada una de las variables.

2. Distribuciones de probabilidad

Las principales distribuciones de variables aleatorias que se trabajan son la binomial, para variables discretas, y la normal para variables continuas. En lenguaje R, cuando se hace referencia a la distribución binomial se escribe '*binom*' y cuando se trata de la normal '*norm*'. Además, si se trabaja con la función

²⁰ `addmargins(prop.table(table(),1))` o `addmargins(prop.table(table(),2))`

de densidad²¹, se añade una 'd' delante del tipo de distribución que se utilice, y si se hace con la función de probabilidad²², se añade una 'p'. En el caso de la binomial la sintaxis es la siguiente: $dbinom(k,n,p)$ o $pbinom(k,n,p)$, siendo k el valor que toma la X , n el número de veces que se repite el experimento, y p la probabilidad de éxito para cada una de las experiencias. En el caso de la normal, la sintaxis es: $dnorm(k, \mu, \sigma)$ o $pnorm(k, \mu, \sigma)$, siendo k el valor que toma la X , μ la media y σ la desviación típica.

Así, si tenemos una variable $X \sim \text{Binomial}(10;0,5)$, la probabilidad $P(X=4)$:

```
prob_4=dbinom(4,10,0.5) #P(X=4)
prob_4
[1] 0.2050781
```

Y la probabilidad $P(X \leq 2)$, $P(X > 2)$ o la de $P(3 \leq X \leq 6)$:

```
prob_menor_2=pbinom(2,10,0.5) #P(X<=2)
prob_menor_2
[1] 0.0546875

prob_mayor_2=1-pbinom(2,10,0.5) #P(X>2)=1-P(X<=2)
prob_mayor_2
[1] 0.9453125

prob_3_6=pbinom(6,10,0.5)-pbinom(3,10,0.5) #P(3<X<=6)=P(X<=6)-P(X<=3)
prob_3_6
[1] 0.65625
```

En el caso de la normal se procede de igual forma, pero utilizando ' $dnorm$ ' y ' $pnorm$ '.

Por otro lado, si queremos calcular (sin fórmula) la media de la población y la desviación típica basta con generar una muestra aleatoria grande de la distribución que trabajemos y calcularlas. La muestra aleatoria se genera añadiendo una 'r' a la distribución: $rbinom(\text{tamaño_muestra},n,p)$ o $rnorm(\text{tamaño_muestra}, \mu, \sigma)$. Por tanto, si queremos calcular dichos parámetros para la distribución binomial anterior:

```
var_binom=rbinom(1000,10,0.5) #Muestra de tamaño 1000
mean(var_binom)
[1] 4.999
sd(var_binom)
[1] 1.561229
```

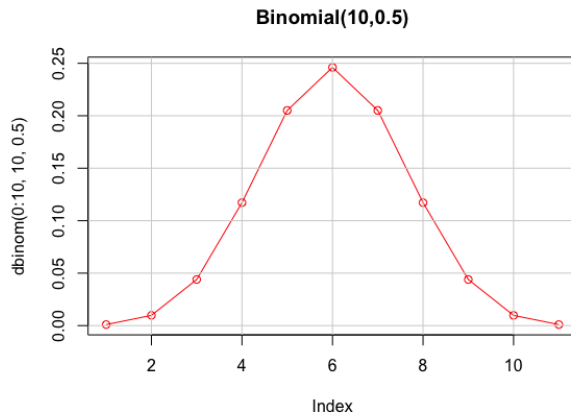
²¹ $P(X=k)$, $X \sim \text{Binomial}(n,p)$ ó $X \sim \text{Normal}(\mu, \sigma)$

²² $P(X \leq k)$, $X \sim \text{Binomial}(n,p)$ ó $X \sim \text{Normal}(\mu, \sigma)$

Los resultados coinciden aproximadamente con los valores que se obtienen utilizando las fórmulas.

Para representar la función de densidad, lo hacemos a través de la función *plot*, pues la función *curve* es para variables continuas:

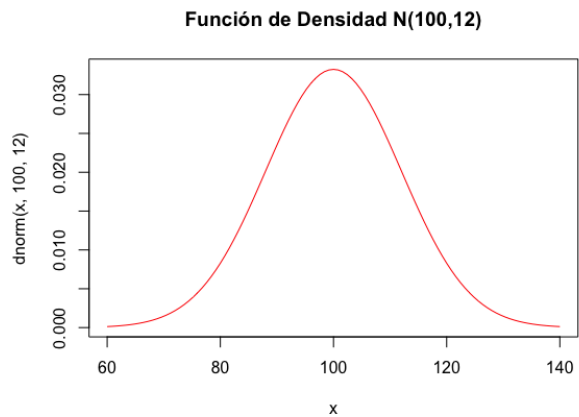
```
plot(dbinom(0:10,10,0.5),
     type="o",col="red",
     main="Binomial(10,0.5)")
grid(lty=1)
```



Para la de probabilidad cambiaríamos 'dbinom' por 'pbinom'.

Para representar la función de densidad de una variable continua, en este caso una variable normal, hay que hacerlo mediante la función *curve(dnorm(), opciones_plot)* de la siguiente forma²³:

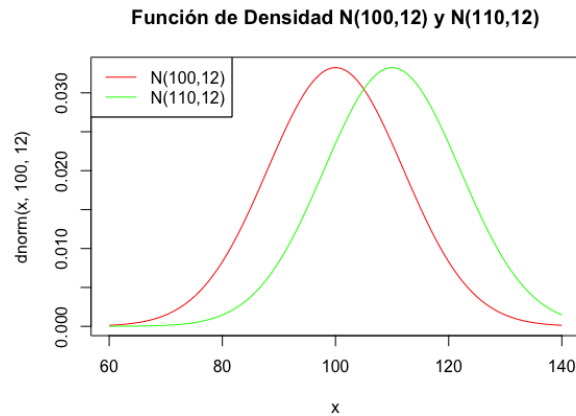
```
curve(dnorm(x,100,12),
      xlim=c(60,140),col="red",
      main="Función de Densidad
      N(100,12)")
```



Para comparar varias distribuciones, debemos añadir en las siguientes que se dibujen la opción 'add=TRUE' y es recomendable cambiar el color, así como también añadir una leyenda:

```
curve(dnorm(x,110,12),col="green",add=TRUE)
legend("topleft",legend=c("N(100,12)", "N(110,12)"),lty=c(1,1),
      col=c("red","green"))
```

²³ Resulta conveniente añadir 'xlim' para centrar el gráfico en la media, en este caso el intervalo es $[\mu-4\sigma, \mu+4\sigma]$



10.4 2º Bachillerato: Matemáticas científicas y aplicadas a las CCSS

10.4.1 Bloque: Números y álgebra

Contenido curricular y aplicación en RStudio

1. Matrices y sistemas

Los contenidos de este bloque se reducen al cálculo matricial, determinantes y a la resolución y discusión de sistemas de ecuaciones.

Para realizar la adición y sustracción de matrices, se definen las matrices y se operan utilizando los operadores comunes. En cambio, para el producto matricial se utiliza una sintaxis diferente a ' $*$ ', pues este multiplica término a término los elementos de las matrices. Así pues, utilizaremos ' $A \% * \% B$ '.

```
A=matrix(c(1,-2,1,3,0,4,0,4,1),nrow=3,ncol=3,byrow = TRUE)
B=matrix(c(0,4,1,1,-2,1,3,0,4),nrow=3,ncol=3,byrow=TRUE)
```

| A | | | | B | | | |
|------|------|------|------|------|------|------|------|
| | [,1] | [,2] | [,3] | | [,1] | [,2] | [,3] |
| [1,] | 1 | -2 | 1 | [1,] | 0 | 4 | 1 |
| [2,] | 3 | 0 | 4 | [2,] | 1 | -2 | 1 |
| [3,] | 0 | 4 | 1 | [3,] | 3 | 0 | 4 |

| A+B | | | | A-B | | | |
|------|------|------|------|------|------|------|------|
| | [,1] | [,2] | [,3] | | [,1] | [,2] | [,3] |
| [1,] | 1 | 2 | 2 | [1,] | 1 | -6 | 0 |
| [2,] | 4 | -2 | 5 | [2,] | 2 | 2 | 3 |
| [3,] | 3 | 4 | 5 | [3,] | -3 | 4 | -3 |

| A%*%B | | | |
|-------|------|------|------|
| | [,1] | [,2] | [,3] |
| [1,] | 1 | 8 | 3 |
| [2,] | 12 | 12 | 19 |
| [3,] | 7 | -8 | 8 |

Para calcular la potencia de una matriz escribiremos ' $A \% ^ \% n$ ', siendo n el número al que elevamos la matriz.

Por otro lado, el cálculo de determinantes se hace relativamente fácil, pues tan solo debemos escribir la función $\det(A)$:

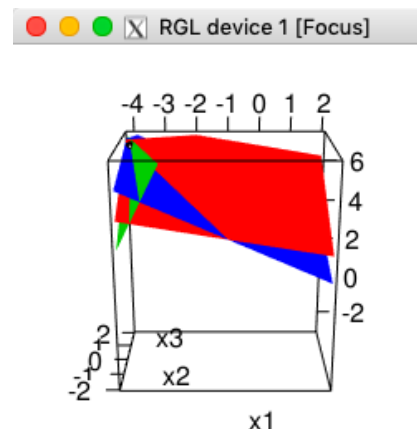
```
det(A)
[1] 2
```

También existe una función en el paquete *Matlib* que permite calcular el determinante según el método que le indiquemos e incluso visualizar los pasos: $\text{Det}(A, \text{method}=\text{c}(\text{"elimination"}, \text{"cofactor"}), \text{verbose}=\text{TRUE})$. Además con $\text{cofactor}(A, i, j)$ y $\text{minor}(A, i, j)$ obtenemos el adjunto y el menor complementario del elemento α_{ij} .

Una vez se tiene clara la notación matricial, se pueden resolver sistemas de ecuaciones de tres ecuaciones y tres incógnitas utilizando el método de Gauss en la matriz de coeficientes, y dibujar los planos resultantes (con la función plotEqn3d del paquete *Matlib*). En el apartado del curso 1º Bachillerato, se encuentra explicado de forma más detallada:

```
library(matlib)
A=matrix(c(1,-3,4,3,1,-1,2,-1,3),
nrow=3,ncol=3,byrow=TRUE)
b=c(21,-18,12)
showEqn(A,b)
1*x1 - 3*x2 + 4*x3 = 21
3*x1 + 1*x2 - 1*x3 = -18
2*x1 - 1*x2 + 3*x3 = 12
Solve(A,b,fractions = TRUE)
x1 = -4
x2 = 1
x3 = 7

plotEqn3d(A,b)
#Representación de los planos
```



Al tratarse de un sistema compatible determinado (S.C.D) los tres planos se cortan en un punto.

Dentro del mismo paquete, existe una función que permite saber a priori si el sistema será S.C.D, S.C.I o S.I, pues calcula el rango de la matriz de coeficientes y el de la matriz ampliada. Para ello, con la función $R(A)$ obtenemos el rango de la matriz de coeficientes A , y con $R(\text{cbind}(A,b))$ el rango de la matriz ampliada denotada habitualmente por A' :

```
c( R(A), R(cbind(A,b)) )
[1] 3 3
```

Como $\text{Rango}(A)=\text{Rango}(A')=3$, el sistema es compatible determinado.

Por último, para calcular la inversa se ha de resolver el sistema $AX=I$, siendo A la matriz de la cual queremos calcular la inversa, X la matriz que queremos

obtener, e I la matriz identidad. Para ello se utiliza la función $solve(A)$ que viene por defecto en RStudio²⁴:

```
solve(A)
      [,1] [,2] [,3]
[1,] -8.0  3.0 -4.0
[2,] -1.5  0.5 -0.5
[3,]  6.0 -2.0  3.0
```

2. Programación Lineal

Existe un paquete de R para resolver problemas de programación lineal mediante un algoritmo conocido como *Método Simplex*, no obstante, se ha considerado no añadirlo por su complejidad.

10.4.2 Bloque: Geometría

Contenido curricular y aplicación en RStudio

En la modalidad de matemáticas aplicadas a las ciencias sociales no existe bloque de geometría. En las matemáticas científicas sí, pero no se han encontrado contenidos que puedan trabajarse con RStudio, tan solo podría servir como calculadora.

10.4.3 Bloque: Análisis

Contenido curricular y aplicación en RStudio

Los contenidos relacionados con conceptos de funciones, representación de estas y límites ya queda recogido en el apartado del Bloque de Análisis del curso anterior. Hay que destacar que en el curso de 2º de Bachillerato se introduce la regla de l' Hôpital, no obstante, la función que se explicó para calcular límites calcula el valor del límite independientemente de las reglas específicas que se utilicen para el cálculo manual. También el cálculo de derivadas aparece en el mismo apartado. Por tanto, tan solo se introducirá el cálculo de primitivas e integrales definidas.

1. Primitivas e integral definida

Al tratarse de cálculo simbólico, se hace uso del paquete *Ryacas*. Para calcular la primitiva de una función respecto una variable utilizamos la función $Integrate(expr,var)$:

²⁴ También se puede utilizar la función 'Inverse(A)' del paquete *Matlib*.

```
library(Ryacas)
int1=expression(Integrate(x+2,x))
yacas(int1)
```

Obtenemos:

```
expression(x^2/2 + 2 * x)
```

Efectivamente, $\int (x + 2)dx = \frac{x^2}{2} + 2x + C$

Para el calcular integrales definidas mediante la Regla de Barrow, a la función anterior añadiremos los límites de integración $\text{Integrate}(expr,var,a,b)$:

```
int2=expression(Integrate(x+2,x,0,1))
yacas(int2)
```

y se obtiene:

```
expression(5/2)
```

Es decir, $\int_0^1 (x + 2)dx = \frac{5}{2}$.

10.4.4 Bloque: Estadística y probabilidad Contenido curricular y aplicación en RStudio

1. Estadística bidimensional

Se explica en el bloque de Estadística y probabilidad relativo a 1º. de Bachillerato.

2. Muestreo y simulación

Desde RStudio podemos generar muestras aleatorias de poblaciones que siguen cualquier distribución de probabilidad, y también podemos seleccionar muestras aleatorias del tamaño deseado de un conjunto de datos, bien con reemplazamiento o bien sin reemplazamiento. Para esto último se utiliza la función $\text{sample}(x, n, \text{replace}=\text{TRUE}/\text{FALSE})$ siendo x el objeto o conjunto de datos del que se seleccionará la muestra, n el tamaño de la muestra, y replace indica si es con reemplazamiento (TRUE) o sin reemplazamiento (FALSE):

```
x=1:100
sample(x,20,replace=FALSE)
[1] 64 58 47 66 77 78 25 44 75 72 24 14 2 29 28 16 15 37 71 46
```

Al ser un procedimiento aleatorio, cada vez que ejecutemos la orden obtendremos unos datos diferentes. Si se pretenden realizar cálculos posteriormente con esos datos, es recomendable establecer una *semilla*, de este modo cada vez que se ejecute el comando aleatorio, se obtendrá el mismo resultado. Esto se realiza a través de la función `set.seed(número)`. Por tanto, si escogemos para esta sesión '`set.seed(2)`', siempre se ejecutará primero esta línea y seguidamente el resto de código y así, podremos obtener siempre los mismos resultados.

Si lo que se pretende es generar una muestra aleatoria de una población, en nuestro caso binomial o normal, se utiliza `rbinom(N,n,p)` o `rnorm(N,μ,σ)` siendo N el tamaño de la muestra que se desee crear. Como se ha indicado anteriormente, al tratarse de un muestreo aleatorio, si se quieren obtener los mismos resultados, se debe añadir la línea de código `set.seed(número)`.

Podemos realizar estimaciones puntuales de la proporción y también de la media a través de las muestras generadas. En el caso de la proporción²⁵ podemos generar una binomial, $Bi(1,0.6)$, que puede indicar, por ejemplo, la probabilidad de acertar una pregunta :

```
set.seed(2)
N=100
x=rbinom(N,1,0.6)
estimacion_p=sum(x)/N #sum suma los valores de x
estimacion_p
[1] 0.62
```

Es decir, la proporción de acertar (es decir, $x=1$) es $\hat{p} = 0.62$.

En el caso de la normal, podemos estimar la media muestral de igual modo, en este caso se utiliza la media:

```
set.seed(2)
N=100
y=rnorm(N,12,12)
estimacion_mu=mean(y)
estimacion_mu
[1] 11.63162
```

Es decir, $\hat{\mu} = 11.63162$.

2. Distribución de la media y la proporción muestral

Dada una distribución cualquiera de una variable X , con media μ y desviación típica σ , $X \sim \text{Distribución}(\mu, \sigma)$, la distribución de las medias muestrales tiene por media la misma que la población, μ , y por desviación típica $\frac{\sigma}{\sqrt{n}}$. Además si el

²⁵ La proporción es el número de éxitos entre el número de pruebas que se realizan.

tamaño de las muestras es mayor que 30, $n \geq 30$, se puede aproximar a través de una distribución normal, $N(\mu, \frac{\sigma}{\sqrt{n}})$.

Para la proporción ocurre de forma similar. Si el tamaño de las muestras mayor que 30, entonces la distribución de las proporciones muestrales se aproxima a través de una distribución normal, $N(p, \sqrt{\frac{pq}{n}})$ siendo p la proporción poblacional y $q=1-p$.

De este modo, podemos generar muestras aleatorias para las distribuciones de las medias y las proporciones de igual modo que se explica en el apartado anterior, tan solo definiendo los nuevos parámetros necesarios.

3. Intervalos de confianza

Obtener la estimación puntual de un parámetro no nos da la información necesaria para saber si efectivamente este valor es representativo. Para ello es importante construir intervalos de confianza, pues es una especie de medida del error de estimación. Se construyen según un determinado nivel de confianza $(1-\alpha)\%$. Los más habituales suelen ser los intervalos a un nivel de confianza del 95%, es decir, $\alpha=0.05$; y los del 90% con $\alpha=0.1$.

- Intervalo de confianza para la media con varianza conocida de una población normal o con tamaño muestral grande

El intervalo para μ es : $(\bar{x} - z_{1-\alpha/2} \frac{\sigma}{\sqrt{n}}, \bar{x} + z_{1-\alpha/2} \frac{\sigma}{\sqrt{n}})$

Para introducirlo en RStudio, se definen las variables *alpha*, *n*, *varianza o desviación típica*, *media* y $z_{1-\alpha/2}$, y se construyen los límites del intervalo aplicando la fórmula anterior. Para el ejemplo elegimos un nivel de confianza del 95%, y una variable $X \sim \text{Distribucion}(\mu = 12, \sigma = 12)$ y una $n=100$:

```
alpha=0.05
n=100
desv.tipica=12
media=12
z=qnorm(1-(alpha/2)) #es el valor de z_{alpha/2}
extr.inf=media-z*desv.tipica/sqrt(n) #extremo inferior del intervalo
extr.sup=media+z*desv.tipica/sqrt(n) #extremo superior del intervalo
IC.95=c(extr.inf,extr.sup)
IC.95
```

[1] 9.648043 14.351957

Es decir, el intervalo de confianza para la media poblacional μ es:

$$IC_{95\%} = (9.648, 14.352)$$

- Intervalo de confianza para la proporción de una población

De igual forma se procede en el caso de las proporciones. El intervalo para la proporción poblacional es p es: $\left(\hat{p} - z_{1-\alpha/2} \sqrt{\frac{\hat{p}(1-\hat{p})}{n}}, \hat{p} + z_{1-\alpha/2} \sqrt{\frac{\hat{p}(1-\hat{p})}{n}} \right)$

Por tanto, tan solo hemos de definir las variables necesarias y aplicar las fórmulas. En este caso, escogemos la proporción muestral, $\hat{p}=0.9$ y la $n=100$:

```
alpha=0.05
n=120
 exitos=90
 p= exitos/n
 z= qnorm(1-(alpha/2)) #es el valor de z_{alpha/2}
 extr.inf= p-z*sqrt(p*(1-p)/n) #extremo inferior del intervalo
 extr.sup= p+z*sqrt(p*(1-p)/n) #extremo superior del intervalo
 IC.95=c(extr.inf,extr.sup)
 IC.95
 [1] 0.6725256 0.8274744
```

Es decir, el intervalo de confianza para la proporción es: $IC_{95\%} = (0.672, 0.827)$

[\(Volver a la lectura\)](#)

11 ANEXO 3: CÓDIGOS DE LOS ARCHIVOS .RMD DEL CAPÍTULO 6

11.1 Código del Ejemplo Ejercicios

```
---
title: "Ejercicios 4.º ESO- Matemáticas orientadas a las enseñanzas académicas"
author: "Alumno"
date: "6/26/2019"
output:
  pdf_document: default
  html_document: default
  word_document: default
---

``{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
``

#### Ejercicio-Álgebra

***1. Encuentra las raíces de los siguientes polinomios y factoriza***:

+  $p_1(x) = x^3 + 2x^2 - x - 2$ ,
+  $p_2(x) = 2x^3 + x^2 - 18x - 9$ ,
+  $p_3(x) = x^2 - \frac{13}{4}x - 3$ .

**Solución:**
Para encontrar las raíces de un polinomio  $p(x)$ , se iguala a cero, es decir  $p(x)=0$  y se resuelve la ecuación.
Por tanto, una vez obtenidas las raíces, podemos escribir el polinomio con factores del tipo  $(x-a)$ , siendo  $a$  una raíz.

+  $p_1(x) = x^3 + 2x^2 - x - 2$ 

``{r }
library(polynom)
p1=polynomial(coef=c(-2,-1,2,1))
raices_p1=solve(p1)
``

Por tanto, las raíces de  $p_1(x)$  son : `r raices_p1`.

Y la factorización será:
 $p_1(x) = (x+2)(x+1)(x-1)$ 

***2. Comprueba gráficamente que las raíces encontradas, lo son. ***

**Solución:**
Para comprobar gráficamente, dibujamos el polinomio, y donde corte con el eje X, debe de coincidir con el valor de las raíces:

``{r }
plot(p1)
```

```
abline(h=0,lty=2,col="red") #Marcar el eje X
```

```
```
```

Vemos como el polinomio corta al eje en los puntos  $x=-2$ ,  $x=-1$  y  $x=1$ . Por tanto, queda comprobado.

\*\*\*3. Los valores  $x=3$ ,  $x=-1$  y  $x=12$ , ¿son raíces del polinomio  $p(x)=3x^4-2x^3+12x-100$ ? \*\*\*

**\*\*Solución:\*\***

Para saber si un valor es raíz de un polinomio, sustituimos dicho valor en el polinomio, y si el resultado es igual 0, es raíz:

```
```{r}
p=polynomial(coef=c(-100,12,0,-2,3))
predict(p, c(3,-1,12))
```
```

Ningún valor es 0, por tanto no son raíces del polinomio.

### ### Ejercicio - Estadística y probabilidad

\*\*\*La profesora de lengua castellana ha contabilizado las faltas de sus alumnos en un examen, y ha obtenido los siguientes resultados:\*\*\*

```
```{r ,echo=FALSE}
faltas_ortografia=c(3, 4, 5, 1, 0, 2, 4, 3, 6, 3, 4, 5, 2, 6, 4, 3, 5, 4, 5, 2, 1, 0, 1, 1, 5, 6, 4)
```
`r faltas_ortografia`
```

\*\*\*1. Representálos con el gráfico adecuado\*\*\* .

**\*\*Solución:\*\***

Como se trata de una variable cuantitativa discreta, podemos representarla con un diagrama de barras:

```
```{r}
barplot(table(faltas_ortografia),main="Diagrama de barras")
```
```

\*\*\*2. ¿Qué porcentaje de alumnos ha hecho 4 faltas de ortografía? \*\*\*

**\*\*Solución:\*\***

Para saberlo, se necesita la tabla de frecuencias relativas y multiplicarla por 100 para obtener el porcentaje:

```
```{r}
prop.table(table(faltas_ortografia))
```
```

Si miramos la columna que indica que el número de faltas es 4, deducimos que el porcentaje es del 22,2\%.

\*\*\*3. ¿Cuántos alumnos han hecho 5 faltas o más? ¿Cuál es el número de faltas más frecuente?\*\*\*

**\*\*Solución:\*\***

Para saberlo, se necesita la tabla de frecuencias absolutas:

```
```{r}
table(faltas_ortografia)
```
```

Por lo tanto, 5 faltas o más son los alumnos que han hecho 5 faltas y 6 faltas. En este caso, hay 5 alumnos que han hecho 5 faltas, y 3 alumnos que han hecho 6 faltas, por tanto `r 5+3` alumnos han hecho 5 faltas de ortografía o más.

Para saber el número de faltas más frecuente, tan solo tenemos que buscar la frecuencia absoluta más grande, es decir, la que se corresponde con 4 faltas.

\*\*\*4. Calcula las medidas de centralización y dispersión, escribiendo sus fórmulas.\*\*\*

\*\*Solución:\*\*

\*\*\*Medidas de centralización\*\*\*

+ \*\*Media\*\*, que es igual a  $\bar{x} = \frac{\sum x_i f_i}{N}$ .

La calculamos y obtenemos que la media,  $\bar{x}$  es `r mean(faltas\_ortografia)` faltas.

+ \*\*Mediana\*\*, obtenemos que  $Me = \text{r median}(faltas_ortografia)$  faltas.

+ \*\*Moda\*\*, ya la hemos calculado en el ejercicio anterior y es 6.

\*\*\*Medidas de dispersión\*\*\*

+ \*\*Rango\*\*, que es igual  $R = \text{Valor}_{\text{màx}} - \text{Valor}_{\text{mín}}$

Lo calculamos,

```
``{r}
range(faltas_ortografia)
``
```

Por tanto, el rango es  $R = 6 - 0 = 6$ .

+ \*\*Varianza y desviación típica\*\*, cuyas fórmulas son:

$$\text{Var} = \sigma^2 = \frac{\sum (x_i)^2 f_i}{N} - \{\bar{x}\}^2$$

$$\sigma = \sqrt{\text{Var}} = \sqrt{\frac{\sum (x_i)^2 f_i}{N} - \{\bar{x}\}^2}$$

Para calcularlas,

```
``{r}
varianza=var(faltas_ortografia)
desv.tipica=sd(faltas_ortografia)
``
```

Y obtenemos, que la varianza,  $\sigma^2 = \text{r varianza}$ , y que la desviación típica,  $\sigma = \text{r desv.tipica}$ .

[\(Volver a la lectura\)](#)



## 11.2 Código del Ejemplo Presentación Ejercicios

```

title: "Corrección de ejercicios"
author: "Alumno"
date: "6/27/2019"
output:
 ioslides_presentation: default
 beamer_presentation: default
 slidy_presentation: default
 smaller: true

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = FALSE)
```

Ejercicio de Álgebra

1.Encuentra las raíces del siguiente polinomio y factoriza:

+ $p_1(x) = x^3 + 2x^2 - x - 2$

Para encontrar las raíces de un polinomio $p(x)$, se iguala a cero, es decir, $p(x)=0$ y se resuelve la ecuación.
Por tanto, una vez obtenidas las raíces, podemos escribir el polinomio con factores del tipo $(x-a)$, siendo a una raíz.

```{r echo=TRUE }
library(polynom)
p1=polynomial(coef=c(-2,-1,2,1))
raices_p1=solve(p1)
raices_p1
```

Por tanto, las raíces de $p_1(x)$ son : `r raices_p1`.

Y la factorización será:

$$p_1(x) = (x+2)(x+1)(x-1)$$

***2. Comprueba gráficamente que las raíces encontradas, lo son. ***

Para comprobar gráficamente, dibujamos el polinomio, y donde corte con el eje X, debe de coincidir con el valor de las raíces:

```{r echo=FALSE}
plot(p1)
abline(h=0,lty=2,col="red") #Marcar el eje X
```

```

\*\*\*

Vemos como el polinomio corta al eje en los puntos  $x=-2$ ,  $x=-1$  y  $x=1$ . Por tanto, queda comprobado.

\*\*\*

\*\*\*3. Los valores  $x=3$ ,  $x=-1$  y  $x=12$ , ¿son raíces del polinomio  $p(x)=3x^4-2x^3+12x-100$ ? \*\*\*

Para saber si un valor es raíz de un polinomio, sustituimos dicho valor en el polinomio, y si el resultado es igual 0, es raíz:

```
```{r echo=TRUE}
p=polynomial(coef=c(-100,12,0,-2,3))
predict(p, c(3,-1,12))
```
```

Ningún valor es 0, por tanto no son raíces del polinomio.

\*\*\*

### ## Ejercicio de Estadística

\*\*\*La profesora de lengua castellana ha contabilizado las faltas de sus alumnos en un examen, y ha obtenido los siguientes resultados:\*\*\*

```
```{r ,echo=FALSE}
faltas_ortografia=c(3, 4, 5, 1, 0, 2, 4, 3, 6, 3, 4, 5, 2, 6, 4, 3, 5, 4, 5, 2, 1, 0, 1, 1, 5, 6, 4)
```
`r faltas_ortografia`
```

\*\*\*

\*\*\*1. Représentalos con el gráfico adecuado\*\*\* .

Como se trata de una variable cuantitativa discreta, podemos representarla con un diagrama de barras:

\*\*\*

```
```{r echo=FALSE, }
faltas_ortografia=c(3, 4, 5, 1, 0, 2, 4, 3, 6,
3, 4, 5, 2, 6, 4, 3, 5, 4, 5, 2, 1, 0, 1, 1, 5, 6, 4)
barplot(table(faltas_ortografia),main="Diagrama de barras")
```
```

\*\*\*

\*\*\*2. ¿Qué porcentaje de alumnos ha hecho 4 faltas de ortografía? \*\*\*

Para saberlo, se necesita la tabla de frecuencias relativas y multiplicarla por 100 para obtener el porcentaje:

```
```{r }
prop.table(table(faltas_ortografia))
```
```

Si miramos la columna que indica que el número de faltas es 4, deducimos que el porcentaje es del 22,2\%.

\*\*\*

\*\*\*3. ¿Cuántos alumnos han hecho 5 faltas o más? ¿Cuál es el número de faltas más frecuente?\*\*\*

Para saberlo, se necesita la tabla de frecuencias absolutas:

```
```{r}
table(faltas_ortografia)
```
```

Por lo tanto, 5 faltas o más son los alumnos que han hecho 5 faltas y 6 faltas. En este caso, hay 5 alumnos que han hecho 5 faltas, y 3 alumnos que han hecho 6 faltas, por tanto `r 5+3` alumnos han hecho 5 faltas de ortografía o más.

Para saber el número de faltas más frecuente, tan solo tenemos que buscar la frecuencia absoluta más grande, es decir, la que se corresponde con 4 faltas.

\*\*\*

\*\*\*4. Calcula las medidas de centralización y dispersión, escribiendo sus fórmulas.\*\*\*

+ \*\*\*Medidas de centralización\*\*\*

- \*\*\*Media\*\*\*, que es igual a

$$\bar{x} = \sum \frac{x_i f_i}{N}$$

La calculamos:

```
```{r}
mean(faltas_ortografia)
```
```

y obtenemos que la media,  $\bar{x}$  es `r mean(faltas\_ortografia)` faltas.

\*\*\*

- \*\*\*Mediana\*\*\*,  
para calcularla:

```
```{r}
median(faltas_ortografia)
```
```

y obtenemos que  $Me =$  `r median(faltas\_ortografia)`

- \*\*\*Moda\*\*\*, ya la hemos calculado en el ejercicio anterior y es 6.

+ \*\*\*Medidas de dispersión\*\*\*

- \*\*\*Rango\*\*\*,

$$R = \text{Valor}_{\{m\grave{a}x\}} - \text{Valor}_{\{m\grave{i}n\}}$$

Lo calculamos,

```
```{r}
range(faltas_ortografia)
```
```

Por tanto, el rango es  $R = 6 - 0 = 6$ .

\*\*\*

-\*\*Varianza y desviación típica\*\*, cuyas fórmulas son:

$$\begin{aligned} \text{Var} &= \sigma^2 = \frac{\sum (x_i)^2 f_i}{N} - \{\bar{x}\}^2 \\ \sigma &= \sqrt{\text{Var}} = \sqrt{\frac{\sum (x_i)^2 f_i}{N} - \{\bar{x}\}^2} \end{aligned}$$

Para calcularlas,

```
``{r}
```

```
varianza=var(faltas_ortografia)
```

```
desv.tipica=sd(faltas_ortografia)
```

```
``
```

Y obtenemos, que la varianza,  $\sigma^2 = \text{r varianza}$ , y que la desviación típica,  $\sigma = \text{r desv.tipica}$ .

[\(Volver a la lectura\)](#)

### 11.3 Código del Ejemplo Apuntes

```

```

```
title: "Apuntes 2.º Bachillerato"
```

```
author: "Profesor"
```

```
date: "6/27/2019"
```

```
output:
```

```
 html_document: default
```

```
 pdf_document: default
```

```

```

```
``{r setup, include=FALSE}
```

```
knitr::opts_chunk$set(echo = TRUE)
```

```
``
```

```
Álgebra - Interpretación geométrica de un sistema de ecuaciones
```

```
Sistemas de dos ecuaciones con dos incógnitas
```

Una vez explicada la forma matricial de un sistema, es importante recalcar la **interpretación geométrica de las ecuaciones** que forman nuestro sistema. Recordar, que en un sistema con dos ecuaciones y dos incógnitas, no son más que dos rectas, que pueden:

+ Ser **secantes**, es decir, cortarse en un punto. En este caso el sistema es Compatible Determinado (S.C.D)

+ Ser **coincidentes**. En este caso el sistema es Compatible Indeterminado (S.C.I), pues existen infinitas soluciones.

+ Ser **paralelas**, es decir, no cortarse en ningún punto. En este caso el sistema es Incompatible (S.I).

**\*\*\*Ejemplo\*\*\***. Sea el sistema:

```
$$\begin{pmatrix}
```

```
1&-2 \ \
```

```
2&-1
```

```
\end{pmatrix}
```

```

\begin{pmatrix}
x \\
y
\end{pmatrix}
=
\begin{pmatrix}
-4 \\
1
\end{pmatrix}

```

Si lo resolvemos, obtenemos por solución:

```

``{r echo=FALSE}
library(matlib)
A=matrix(c(1,-2,2,-1),nrow=2,ncol=2,byrow = TRUE)
b = c(-4,1)
solucion_sistema=Solve(A, b, fractions = TRUE)
``

```

Si además, dibujamos las ecuaciones, observamos que son dos rectas secantes que se cortan en un punto:

```

``{r echo=FALSE, results='hide'}
plotEqn(A,b)
``

```

y por tanto el sistema es Compatible Determinado.

### ### Sistemas de tres ecuaciones con tres incógnitas

Si estamos ante un sistema de **tres** ecuaciones con **tres** incógnitas, se incorpora una nueva incógnita, y por tanto, ya no hablamos de rectas, sino de **planos**.

En primer lugar, sabemos que a partir de los rangos de la matriz original y la matriz ampliada, podemos saber si el sistema tiene una única solución (S.C.D), tiene infinitas (S.C.I) o no tiene solución (S.I).

Sea  $A$ , la matriz de coeficientes,  $A'$  la matriz ampliada,  $n$  el número de incógnitas y,  $r$  y  $r'$  los rangos de la matriz  $A$  y la matriz  $A'$  respectivamente. Entonces,

- + Si  $r=r'$  y  $r=n$ , entonces el sistema es **Compatible Determinado**.
- + Si  $r=r'$  y  $r<n$ , entonces el sistema es **Compatible Indeterminado**.
- + Si  $r \neq n$ , entonces el sistema es **Incompatible**.

La interpretación geométrica de tres ecuaciones con tres incógnitas, presenta más variantes que la anterior; pues al tratarse de tres planos, las combinaciones posibles aumentan. Veamos un par de ejemplos:

**Ejemplo 1**. Sea el sistema:

```

\begin{pmatrix}
1&-3&4 \\
3&1&-1 \\
2&-1&3
\end{pmatrix}

```

```

\begin{pmatrix}
x \\
y \\
z
\end{pmatrix}
=
\begin{pmatrix}
21 \\
-18 \\
12
\end{pmatrix}

```

En primer lugar, calculamos el rango de la matriz de coeficientes y el de la ampliada, y se obtiene:

```

```{r echo=FALSE}
library(matlib)
A=matrix(c(1,-3,4,3,1,-1,2,-1,3),nrow=3,ncol=3,byrow=TRUE)
b=c(21,-18,12)
rangoA=R(A)
rangoAmp=R(cbind(A,b))
```

```

que el  $R(A) = r \text{ rangoA}$ , y  $R(A') = r \text{ rangoAmp}$ . Como son iguales, y además es igual al número de incógnitas, el sistema es Compatible Determinado, y por tanto los tres planos se cortarán en un punto, que es la solución del sistema:

```

```{r echo=FALSE}
library(matlib)
Solve(A,b)
plotEqn3d(A,b)
```

```

![Tres planos que se cortan en un punto](/Users/mariaespinosa/Desktop/Capturas TFM/Captura de pantalla 2019-06-27 a las 18.48.25.png)

\*\*\*Ejemplo 2\*\*\*. Sea el sistema:

```


$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 21 \\ -18 \\ 12 \end{pmatrix}$$


```

En primer lugar, calculamos el rango de la matriz de coeficientes y el de la ampliada, y se obtiene:

```
``{r echo=FALSE}
library(matlib)
A=matrix(c(1,1,1,1,1,1,1,1,1),nrow=3,ncol=3,byrow=TRUE)
b=c(-2,4,-2)
rangoA=R(A)
rangoAmp=R(cbind(A,b))
``
```

que el  $\text{Rango}(A) = \text{rangoA}$ , y  $\text{Rango}(A') = \text{rangoAmp}$ . Como son diferentes, el sistema es Incompatible, y por tanto los tres planos pueden ser los tres paralelos entre sí, o puede haber dos planos coincidentes paralelos al restante. Veámoslo:

```
``{r echo=FALSE}
library(matlib)
Solve(A,b)
plotEqn3d(A,b)
``
```

![Dos planos coincidentes paralelos al tercero](/Users/mariaespinosa/Desktop/Capturas TFM/Captura de pantalla 2019-06-27 a las 19.06.39.png)

Por tanto, esto nos indica que dos de los planos son coincidentes, y son paralelos al tercero.

## Análisis - Primitivas, integral definida. Regla de Barrow.

### Primitivas e integral indefinida.

\*\*\*Definición\*\*\*: Sea  $f(x)$  una función real de variable real. Llamamos **primitiva** de  $f(x)$  a la función  $F(x)$  tal que su derivada es igual a la función  $f(x)$ . Es decir, a la función  $F(x)$  que cumple:  $F'(x) = f(x)$

**Ejemplo**: La función  $F(x) = x^3 + 2x - 1$  es una primitiva de  $f(x) = 3x^2 + 2$ , ya que  $F'(x) = f(x)$ .

\*\*\*Definición\*\*\*: La **integral indefinida** de una función  $f(x)$  es el conjunto de todas sus primitivas. Se denota por:  $\int f(x) dx$ . Por tanto, si  $F(x)$  es una primitiva de  $f(x)$ , entonces:

$\int f(x) dx = F(x) + C$   
siendo  $C \in \mathbb{R}$ , la constante de integración.

### Integral definida. Regla de Barrow.

\*\*\*Definición\*\*\*: La **integral definida** de una función  $f(x)$  continua en un intervalo  $[a, b]$ , es igual al área entre la curva  $f(x)$ , las rectas  $x=a$  y  $x=b$ , y el eje de abscisas,  $y=0$ . Lo denotamos por:

$\int_a^b f(x) dx$

Los valores  $a$  y  $b$  son los **límites de integración**.

Sabemos que para calcular el área bajo una curva, debemos calcular la integral, y en este caso se hará entre los valores  $a$  y  $b$ , aplicando la regla de Barrow.

**\*\*\*Regla de Barrow:\*\*\*** Sea  $f(x)$  una función continua, definida en un intervalo  $[a,b]$ , y  $F(x)$  una primitiva de  $f(x)$ , entonces:

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

**\*\*Ejemplo\*\*:** Sea  $f(x) = x^2 + 3$ .

Calcula el área entre la curva  $f(x)$ , el eje de abscisas, y la rectas  $x = -2$  y  $x = 2$ .

En primer lugar, la función  $f(x)$  es continua en el intervalo  $[-2,2]$  por ser una función polinómica.

Entonces queremos calcular el área que encierra la parábola junto con dos rectas verticales y el eje de abscisas, es decir:

```
``{r echo=FALSE}
library(polynom)
p=polynomial(coef=c(3,0,1))
plot(p,xlim = c(-4,4),ylim = c(0,10))
abline(v=-2,lty=2,lwd=2,col="red")
abline(v=2,lty=2,lwd=2,col="red")
abline(h=0,lty=2,lwd=2,col="blue")
``
```

Para ello, aplicamos la regla de Barrow, y se obtiene el valor:

```
``{r echo=FALSE}
library(Ryacas)
x=Sym("x")
integral_def=(Integrate(x^2+3,x,-2,2))
yacac(integral_def)
``
```

Es decir,

$$\int_{-2}^2 (x^2+3)dx = \frac{52}{3} \text{ u}^2$$

que será el valor del área buscada.

[\[Volver a la lectura\]](#)

## 11.4 Código del Ejemplo Presentación Apuntes

```

title: "Apuntes 2º Bachillerato"
author: "Profesor"
date: "6/27/2019"
output:
 beamer_presentation: default
 ioslides_presentation: default
 smaller: true

``{r setup, include=FALSE}
knitr::opts_chunk$set(echo = FALSE)
``

Álgebra - Interpretación geométrica de un sistema de ecuaciones
```



### ### **\*\*Sistemas de dos ecuaciones con dos incógnitas\*\***

Una vez explicada la forma matricial de un sistema, es importante recalcar la **\*\*interpretación geométrica de las ecuaciones\*\*** que forman nuestro sistema. Recordar, que en un sistema con dos ecuaciones y dos incógnitas, no son más que dos rectas, que pueden:

\*\*\*

+ Ser **\*\*secantes\*\***, es decir, cortarse en un punto. Por tanto, el sistema es Compatible Determinado (S.C.D)

+ Ser **\*\*coincidentes\*\***. Por tanto, el sistema es Compatible Indeterminado (S.C.I), pues existen infinitas soluciones.

+ Ser **\*\*paralelas\*\***, es decir, no cortarse en ningún punto. Por tanto, el sistema es Incompatible (S.I).

\*\*\*

\*\*\*Ejemplo\*\*\*. Sea el sistema:

$$\begin{pmatrix} 1 & -2 \\ 2 & -1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -4 \\ 1 \end{pmatrix}$$

Si lo resolvemos, obtenemos por solución:

```
``{r echo=FALSE}
library(matlib)
A=matrix(c(1,-2,2,-1),nrow=2,ncol=2,byrow=TRUE)
b=c(-4,1)
solucion_sistema=Solve(A,b,fractions=TRUE)
``
```

Si además, dibujamos las ecuaciones, observamos que son dos rectas secantes que se cortan en un punto:

\*\*\*

```
``{r echo=FALSE, results='hide'}
plotEqn(A,b)
``
```

y por tanto el sistema es Compatible Determinado.

\*\*\*

### ### **\*\*Sistemas de tres ecuaciones con tres incógnitas\*\***

Si estamos ante un sistema de **\*\*tres\*\*** ecuaciones con **\*\*tres\*\*** incógnitas, se incorpora una nueva incógnita, y por tanto, ya no hablamos de rectas, sino de **\*\*planos\*\***.

En primer lugar, sabemos que a partir de los rangos de la matriz original y la matriz ampliada, podemos saber si el sistema tiene una única solución (S.C.D), tiene infinitas (S.C.I) o no tiene solución (S.I).

\*\*\*

Sea  $A$ , la matriz de coeficientes,  $A'$  la matriz ampliada,  $n$  el número de incógnitas y,  $r$  y  $r'$  los rangos de la matriz  $A$  y la matriz  $A'$  respectivamente. Entonces,

- + Si  $r=r'$  y  $r=n$ , entonces el sistema es **Compatible Determinado**.
- + Si  $r=r'$  y  $r<n$ , entonces el sistema es **Compatible Indeterminado**.
- + Si  $r \neq n$ , entonces el sistema es **Incompatible**.

\*\*\*

La interpretación geométrica de tres ecuaciones con tres incógnitas, presenta más variantes que la anterior; pues al tratarse de tres planos, las combinaciones posibles aumentan. Veamos un par de ejemplos:

\*\*\*Ejemplo 1\*\*\*. Sea el sistema:

$$\begin{pmatrix} 1 & -3 & 4 \\ 3 & 1 & -1 \\ 2 & -1 & 3 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 21 \\ -18 \\ 12 \end{pmatrix}$$

En primer lugar, calculamos el rango de la matriz de coeficientes y el de la ampliada, y se obtiene,

```

>{r echo=FALSE}
library(matlib)
A=matrix(c(1,-3,4,3,1,-1,2,-1,3),nrow=3,ncol=3,byrow=TRUE)
b=c(21,-18,12)
rangoA=R(A)
rangoAmp=R(cbind(A,b))

```

...

que el  $Rango(A) = r \text{ rangoA}$ , y  $Rango(A') = r \text{ rangoAmp}$ .

\*\*\*

Como son iguales, y además es igual al número de incógnitas, el sistema es **Compatible Determinado**, y por tanto los tres planos se cortarán en un punto, que es la solución del sistema:

![Tres planos que se cortan en un punto](/Users/mariaespinosa/Desktop/Capturas TFM/Captura de pantalla 2019-06-28 a las 0.51.52.png)

[\(Volver a la lectura\)](#)



+ Frecuencias relativas

```
``{r}
prop.table(table(faltas_ortografia))
``
```

2. Para saber el porcentaje de `falt1` faltas, debemos mirar la tabla de frecuencias relativas y multiplicar por 100:

```
``{r}
prop.table(table(faltas_ortografia))[falt1+1] *100
``
```

3. Para calcular el número de alumnos que ha hecho `falt2` faltas o más, vamos a la tabla de frecuencias absolutas, y sumamos todos los valores que queden por encima de `falt2`, este inclusive:

```
``{r}
table(faltas_ortografia)[falt2+1:max(faltas_ortografia)]
``
```

4. La representación es un diagrama de barras:

```
``{r}
barplot(table(faltas_ortografia),main="Diag.de barras")
``
```

---

```
`r c('-->', "[1+solucion]`
```

[\(Volver a la lectura\)](#)

---