universidad de león

Primeros pasos con R y RStudio

Métodos Numéricos y Estadísticos
Grado en Ingeniería Informática / Mecánica

Curso 2020-2021

Eva María Mazcuñán Navarro

Contenidos

Página

1 Requisitos previos **2**

2 Escribir y ejecutar código de R **2**
2.1 Consola . . . 2
2.2 Scripts de R . . . 3
2.3 Documentos R Markdown . . . 4
2.3.1 Creación . . . 5
2.3.2 Compilación . . . 6
2.3.3 Encabezados . . . 8
2.3.4 Bloques de código . . . 8

3 Plantilla de R Markdown **10**
3.1 Primeros contenidos . . . 10
3.2 Algunas opciones . . . 11
3.3 Opciones globales . . . 11
3.4 Ejecución individual de un bloque de código . . . 12
3.5 Tabla de contenidos flotante . . . 14
3.6 Resumen . . . 14
3.7 Flujo de trabajo . . . 15

4 Paquetes **15**

5 Tipos de objetos en R **16**

6 Gráficos **16**

En este documento se da una breve introducción al lenguaje de programación R y a la interfaz gráfica RStudio.

Se trata de una práctica introductoria, en la que no hay que entregar ningún material, y no hay ninguna tarea evaluable asociada. Pero es necesario que estudies este material como previo a la realización de las siguientes prácticas y tareas.

1. Requisitos previos

Para realizar esta práctica tienes que instalar R y RStudio en tu equipo.

En el documento Instalación de R y RStudio encontrarás las instrucciones para hacerlo.

2. Escribir y ejecutar código de R

Para escribir y ejecutar código de R desde RStudio podremos utilizar bien la consola o bien escribir nuestro código en un archivo.

La **consola** se utiliza para ejecutar instrucciones sueltas, que no tenemos interés en conservar, por ejemplo para realizar cálculos auxiliares o para instalar paquetes.

Cuando el código que queremos escribir sea un conjunto de instrucciones que queramos conservar, de forma que podamos reutilizarlo posteriormente o compartirlo con otras personas, escribiremos nuestro código en un archivo. Entre los tipos de archivos que podemos crear desde RStudio para escribir y ejecutar código R están los **scripts** y los documentos **R Markdown**.

2.1. Consola

En RStudio encontrarás la consola en el panel de nombre Console en la ventana a la izquierda de la pantalla.

Para calcular $\sqrt{2}$ desde la consola, sitúa el cursor al lado del símbolo `>` en la consola, escribe la instrucción `sqrt(2)` y presiona `Enter`. Verás la salida

debajo:

```
> sqrt(2)

[1] 1.414214
```

2.2. Scripts de R

Los scripts de R son el tipo de archivo más simple para escribir y ejecutar código de R.

Para crear un script, utiliza el menú

File > New File > R Script

El script se abrirá en una pestaña de una nueva ventana sobre la ventana con el panel de la consola. Este script no es más que un archivo de texto, que se guardará con la extensión `.R`.

Escribe en la primera línea del script la instrucción

```
sqrt(3)
```

Para ejecutarla, sitúa el cursor sobre cualquier punto de la línea y presiona `Ctrl + Enter`. Verás la salida en la consola.

Si en un script queremos incluir varias instrucciones, cada nueva instrucción debe comenzar en una nueva línea.

Añade una nueva línea al script para calcular la raíz de 5, de forma que el contenido del script quede:

```
sqrt(3)
sqrt(5)
```

Para ejecutar las dos instrucciones al mismo tiempo, selecciona las dos líneas y presiona de nuevo `Ctrl + Enter`. En la consola, verás las dos instrucciones y su salida correspondiente.

Pueden dejarse tantas líneas en blanco como se quiera entre diferentes instrucciones, y también dividir el código de una misma instrucción en varías líneas. Por ejemplo:

```
sqrt(3)

sqrt(
```

```
    5
)
```

Notar que si situamos el cursor sobre cualquiera de las tres líneas que componen la segunda instrucción para calcular la raíz de 5 y presionamos `Ctrl + Enter`, RStudio reconoce que la línea en la que tenemos el cursor forma parte de una instrucción compuesta por varias líneas y ejecuta todas ellas.

Para añadir comentarios en un script, se utiliza el carácter `#`: Al ejecutar una línea de código, todo el texto escrito después del carácter `#` será ignorado. Puedes escribir por ejemplo

```
# calcular raíces
sqrt(3)
```

o

```
sqrt(3) # calcular la raíz de 3
sqrt(5) # calcular la raíz de 5
```

Crea ahora una carpeta, de nombre`IntroR`, para guardar el script que acabas de escribir y otros documentos que generaremos a lo largo de la práctica.

Para guardar el script que acabas de escribir, presiona `Ctrl + S` (o utiliza el correspondiente icono en la barra de herramientas del archivo).

Si aparece un cuadro de diálogo preguntando por la codificación del archivo, selecciona la codificación que aparezca listada en primer lugar como defecto para tu sistema operativo (verás el texto (System default) al lado de su nombre).

En el selector de archivos que se abrirá a continuación, navega hasta la carpeta `IntroR` que has creado antes e indica `script` como nombre del archivo. Verás entonces que la etiqueta de la pestaña del script en el panel de RStudio cambia de `Untitled1` a `script.R`.

2.3. Documentos R Markdown

Nuestra elección para realizar las prácticas y tareas de esta asignatura será utilizar documentos R Markdown. En un documento R Markdown podremos escribir tanto código R como texto. Y al compilarlo obtendremos un documento que incluirá el código, la salida resultante de ejecutar el código, y el texto explicativo.

- En el texto podremos utilizar la sintaxis propia del lenguaje de marcado

Markdown (independiente de R). Por ejemplo:

```
El resultado anterior permite extraer una conclusión muy **importante**.
```

- Y el código de R se incluirá en unos bloques especiales, que tendrán la estructura

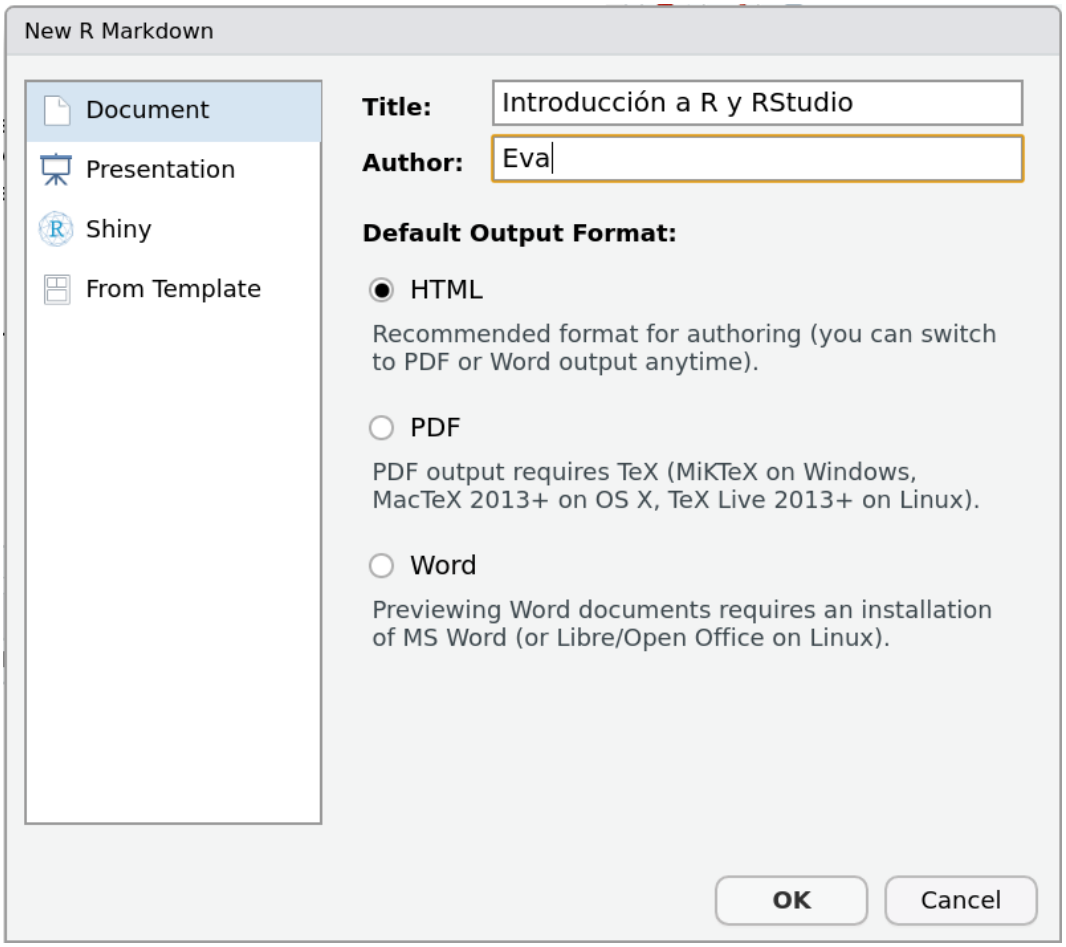
```
```{r [etiqueta], [opciones]}
<código R>
```
```

2.3.1. Creación

Para crear tu primer documento R Markdown utiliza el menú

File > New File > R Markdown ...

Aparecerá un cuadro de diálogo de nombre **New R Markdown**.



Rellena 'Introducción a R y RStudio' en el campo **Title** y tu nombre en el campo **Author**. Para el campo **Default output format**, conserva la elección 'HTML' que aparece por defecto.

Al presionar el botón **OK** se abrirá una nueva pestaña en el panel de RStudio con el nuevo documento R Markdown. Presiona `Ctrl + S` para guardarlo, en la carpeta `IntroR` que creaste antes para la práctica, con el nombre `intro-r`. Verifica que la etiqueta de la pestaña del documentocambia de `Untitled1` a `intro-r.Rmd`.

Las primeras líneas del archivo (1 a 6), delimitadas por tres guiones (`---`)

```
---
title: "Introducción a R y RStudio"
author: "Eva"
date: "19/4/2021"
output: html_document
---
```

conforman la llamada cabecera YAML del documento. Incluye metadatos como el título, el autor y la fecha y el formato de salida del documento que se generará al compilar.

Las líneas siguientes (8 a 10)

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

las incluiremos en todos nuestros documentos R Markdown, y establecen la opción de mostrar el código que incluyamos en el cuerpo de nuestro documento `.Rmd` en el formato de salida.

El resto de líneas (12 en adelante), son los contenidos propiamente dichos del documento. Se trata de unos contenidos de muestra, que enseguida reemplazaremos por nuestros contenidos propios. Pero antes de borrar estos contenidos de muestra, compilaremos el documento para ver el resultado inicial.

2.3.2. Compilación

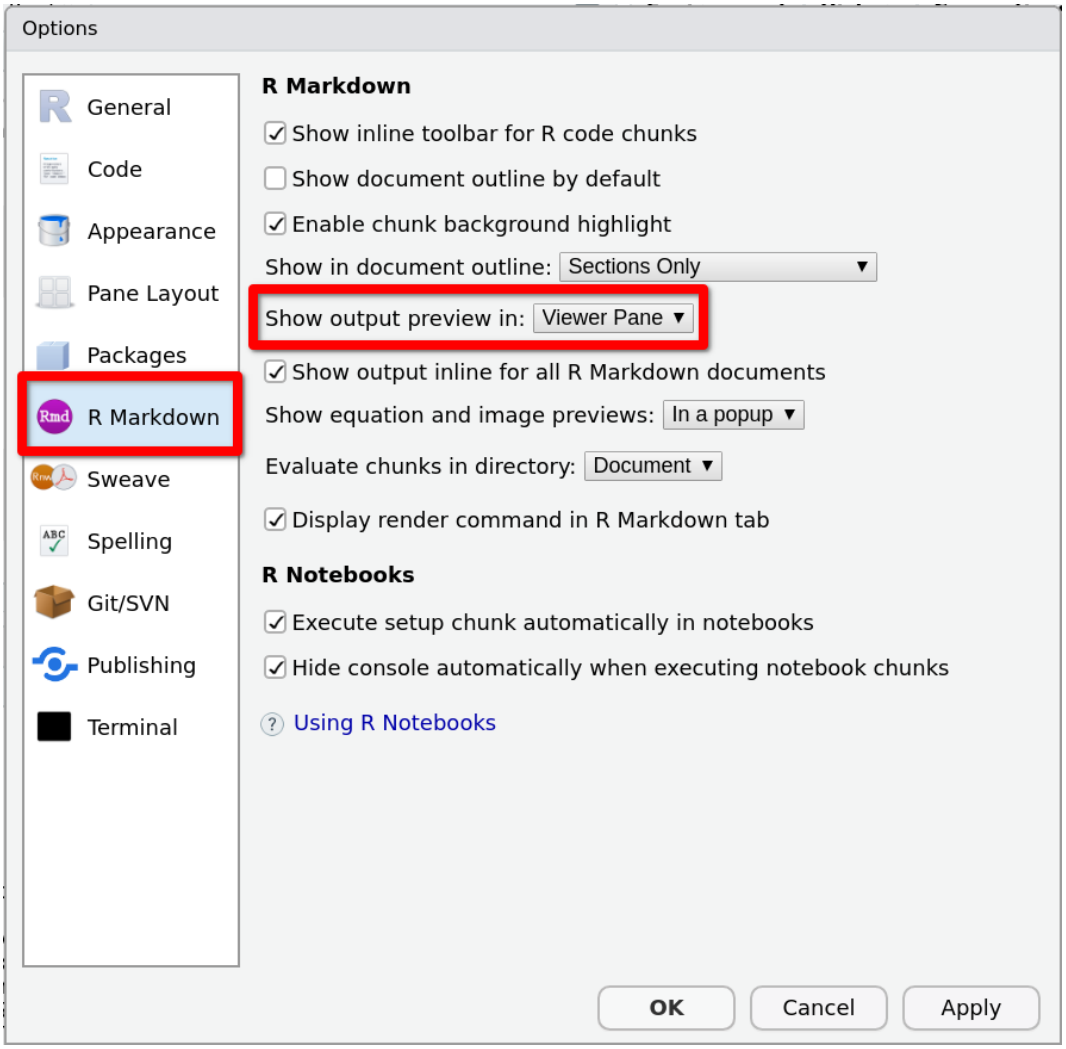
Para compilar el documento presiona el botón **Knit** en la barra de herramientas de la pestaña del archivo.

El documento compilado aparecerá en el panel **Viewer**, en la ventana de la zona derecha inferior. Si el documento no se abre en este panel, sino en una ventana emergente, cierra esa ventana y modifica este comportamiento siguiendo los siguientes pasos:

1. Selecciona el menú ::: {.menu data-latex=} Tools > Global Options :::
2. Se abrirá una ventana de nombre **Options**. Selecciona la sección **R Markdown** en el menú lateral e indica

Show output preview in: Viewer Pane

como indica la imagen siguiente:



Si miras los contenidos de la carpeta `IntroR` (puedes hacerlo desde el panel

Files de RStudio) verás que, como resultado de la compilación del archivo fuente `intro-r.Rmd`, se ha creado el archivo de salida `intro-r.html`. Éste es el archivo que estamos visualizando en el visor de documentos. También podríamos abrirlo en el navegador web, pudiendolo hacer desde el propio visor, presionando el icono resaltado en la siguiente imagen:

A continuación compararemos el documento fuente `intro-r.Rmd` con el documento `intro-r.html` en el visor, para entender cómo se traducen los contenidos que escribimos en un archivo R Markdown en el formato de salida HTML. Nos fijaremos en particular en los siguientes elementos: encabezados y bloques de código.

2.3.3. Encabezados

Las líneas 21

```
## R Markdown
```

y 31

```
## Including Plots
```

se traducen en la salida como encabezados de secciones. Si inspeccionas el código del archivo `intro-r.html` verás que se crean elementos de tipo `<h2>`. En general,

```
# Título
```

produce un encabezado de nivel 1,

```
## Título
```

un encabezado de nivel 2, y así sucesivamente.

Hay que indicar que es una casualidad la coincidencia del símbolo `#` para encabezados en el lenguaje Markdown (independiente de R) y para comentarios en código R.

2.3.4. Bloques de código

Los elementos más importantes del documento son los bloques de código R (*code chunks*). En el documento de muestra encontramos dos bloques de código:

El primero en las líneas 27-29

```
```{r cars}
summary(cars)
```
```

y el segundo en las líneas 35-37

```
```{r pressure, echo=FALSE}
plot(pressure)
```
```

Si miras el documento compilado, verás que, para el primer bloque se muestra el código y a continuación la salida o resultado de su ejecución; mientras que para el segundo, se muestra solo la salida, y no el código debido a la opción `echo=FALSE`.

Como se indicó al principio de la sección, la sintaxis general para incluir un bloque de código R en un documento R Markdown es la siguiente:

```
```{r [etiqueta], [lista de opciones]}
<código R>
```
```

Las etiquetas de los dos bloques de código del documento de muestra son `cars` y `presure`. La etiqueta de un bloque de código sirve para identificarlo, podemos interpretarlo como su nombre, pero es opcional y puede omitirse.

El primer bloque de código no tiene ninguna opción. Y el segundo tiene la opción `echo=TRUE`, que como hemos dicho antes inhibe la impresión del código en el documento compilado.

Puesto que la etiqueta y la lista de opciones son opcionales, el esqueleto básico de un bloque de código R incluido en un documento R Markdown es

```
```{r}
<código R>
```
```

En el cuerpo del bloque podemos escribir instrucciones de R igual que si estuvieramos escribiendo en un script (incluidos comentarios precedidos por el carácter `#`).

Notar el cambio de enfoque al escribir en un documento R Markdown respecto a escribir en un script:

- En un script, se espera que escribamos código R, y para escribir texto

ordinario hemos de usar comentario utilizando el carácter `#`

- Por el contrario, en un documento R Markdown, se espera que escribamos texto (con posibilidad de incluir la sintaxis propia del lenguaje Markdown), y para escribir código R hemos de incluirlo en un bloque especial delimitado por ```` ```{r} ```` y ```` ``` ````.

3. Plantilla de R Markdown

Ahora vamos a reemplazar los contenidos de muestra del archivo de R Markdown que hemos creado por nuestros propios contenidos. Crearemos un primer capítulo con un primer bloque código y dejaremos el documento preparado para practicar el código de R que se presenta en los siguientes capítulos.

3.1. Primeros contenidos

Borra los contenidos de muestra (línea 15 en adelante) y añade la siguiente línea para crear un encabezado de nivel 1:

```
# Bloques de código
```

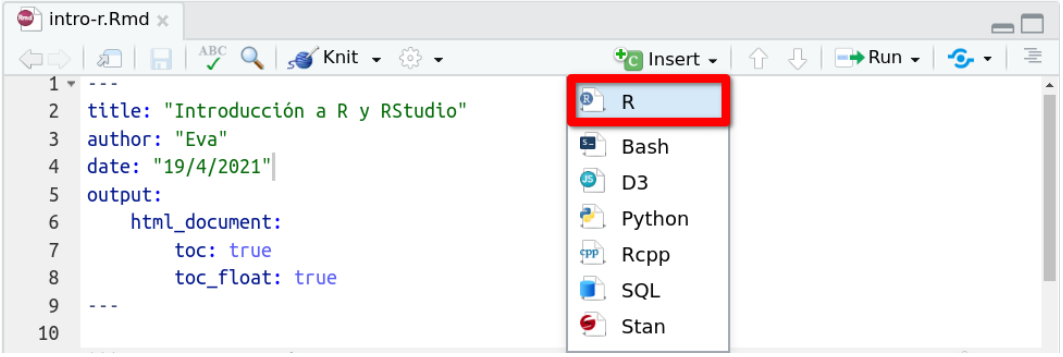
Ahora vamos a crear un primer bloque de código. Para escribir su esqueleto usa el atajo

`Ctrl + Alt + I`

(`I` de *Insert*) o, alternativamente, el menú

Insert > R

en la barra de herramientas de la pestaña del documento, que se muestra en la siguiente imagen:



En el bloque de código que acabas de crear añade las instrucciones

```
sqrt(8)
sqrt(10)
```

de forma que los contenidos añadidos queden:

```
# Bloques de código

```{r}
sqrt(8)
sqrt(10)
```
```

Compila para ver el resultado.

3.2. Algunas opciones

Tras compilar el documento, verás que en el documento HTML generado aparecen:

- El código de la primera instrucción `sqrt(8)`
- Su salida 2.8284271
- El código de la segunda instrucción `sqrt(10)`
- Su salida 3.1622777

Si prefieres que se muestre primero el código para las dos instrucciones y a continuación las dos salidas, añade la opción `results='hold'`:

```
```{r, results='hold'}
sqrt(8)
sqrt(10)
```
```

y vuelve a compilar para ver el resultado.

Si quieres omitir los caracteres `##` al comienzo de las líneas de la salida, añade la opción `comment = ''`.

3.3. Opciones globales

Al comienzo de nuestro documento, hemos conservado el bloque de código

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

que estaba incluido en el documento de muestra.

Las opciones especificadas en el argumento de `knitr::opts_chunk$set` aplicarán a todos los bloques de código que se incluyan en el documento.

Si se desea aplicar las opciones `results='hold'` y `comment = ''` que se explicaron antes, a todos los bloques de código del documento, las añadiremos al argumento de la función `knitr::opts_chunk$set`, de forma que quede:

```
knitr::opts_chunk$set(
    echo = TRUE,
    results='hold',
    comment = ''
)
```

En tal caso, estas dos nuevas opciones aplicarán a todos los bloques, sin necesidad de repetirlas de forma individual en cada uno de ellos, y puedes borrarlas del bloque de código que creaste antes.

3.4. Ejecución individual de un bloque de código

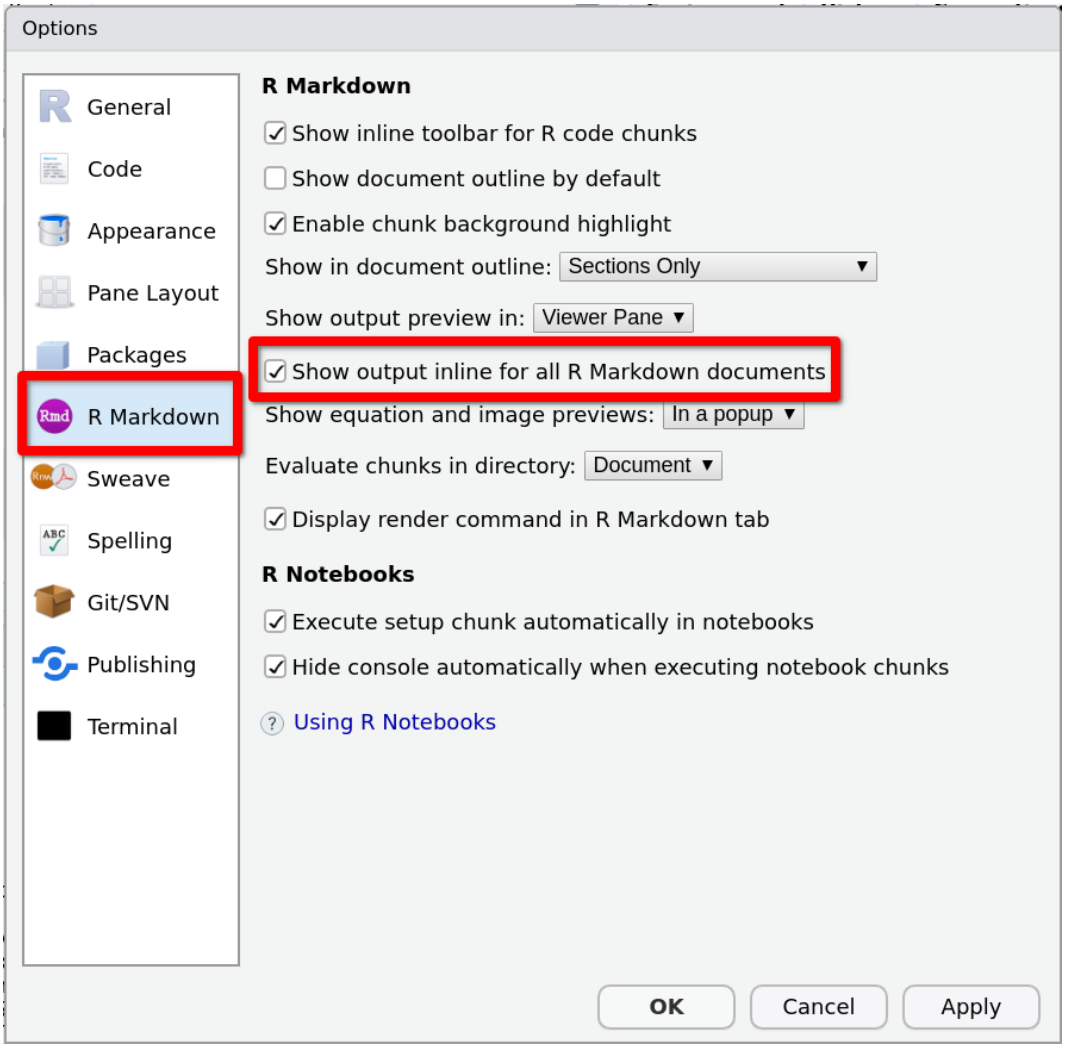
Cuando compilamos un documento R Markdown, se ejecutan todos los bloques de código que contenga, y en el documento compilado podemos visualizar, para cada bloque, el código y la salida (si no hay opciones como `echo=FALSE` que inhiban la impresión del código y/o de la salida).

Pero también podemos ejecutar determinadas instrucciones en un bloque de código de forma individual, sin necesidad de compilar el documento completo. Para ello, podemos proceder exactamente igual que en el caso de los scripts, es decir:

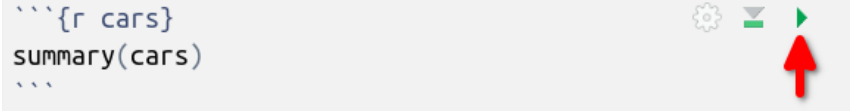
- Para ejecutar una sola instrucción, situamos el cursor en cualquiera de las líneas que compongan la instrucción y presionamos `Ctrl + Enter`.
- Para ejecutar varias instrucciones, seleccionamos las correspondientes líneas y presionamos `Ctrl + Enter`.

La salida se mostrará en la consola, y también incrustada en el propio documento, justo debajo del bloque de código. Para esto último ha de estar marcada la

opción **Show output inline for all R Markdown documents** en las opciones para R Markdown, como muestra la siguiente imagen:



Además, podemos ejecutar todas las instrucciones que componen un bloque de código utilizando el botón a la derecha del comienzo del bloque que se resalta en la siguiente imagen:



3.5. Tabla de contenidos flotante

Ahora vamos a personalizar el formato de salida para que nuestro documento incluya una tabla de contenidos flotante y para numerar los capítulos.

Para ello modificamos la línea

```
output: html_document
```

en la cabecera YAML por

```
output:
    html_document:
        toc: true
        toc_float: true
        number_sections: true
```

Asegúrate de indentar las líneas conforme se indica, porque el indentado es fundamental para que los campos anidados se lean correctamente en el proceso de compilación.

Para que nuestra tabla de contenidos tenga más de una entrada, añade al final del documento un segundo capítulo, con el título del siguiente capítulo de esta práctica:

```
# Paquetes
```

Compila de nuevo y abre el resultado en el navegador. Verás la tabla de contenidos con los títulos de los capítulos numerados.

Hasta que el documento no tenga la extensión suficiente para ocupar la pantalla completa y que aparezca la barra de *scroll* para recorrerlo, no se apreciará la funcionalidad completa de la tabla de contenidos flotante.

3.6. Resumen

Después de los cambios que hemos ido haciendo en el documento de muestra, ha debido quedarte así:

```
---
title: "Introducción a R y RStudio"
author: "Eva"
date: "19/4/2021"
output:
    html_document:
```

```
        toc: true
        toc_float: true
        number_sections: true
---

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

Bloques de código

```{r}
sqrt(8)
sqrt(10)
```

Paquetes
```

3.7. Flujo de trabajo

La idea es que, conforme vayas estudiando el resto de capítulos de esta práctica, continúes escribiendo en el documento R Markdown que tienes ahora.

La idea es que crees un nuevo capítulo por cada capítulo de la práctica, e incluyas bloques de código para practicar el código de R que vayas encontrando en la práctica, así como el texto que creas oportuno para documentar el código y entenderlo cuando releas el documento. Experimenta, escribiendo el código que te apetezca para probar las ideas que te vayan surgiendo y escribiendo el texto que consideres para explicarlas.

Puedes ir compilando cada bloque o instrucción de forma individual, para ver su salida *inline* y compilar el documento cada cierto tiempo.

4. Paquetes

TODO

5. Tipos de objetos en R

TODO

6. Gráficos

TODO