

Chunks en R

Tony

Chunks

Chunk: Bloque de código.

Los bloques de código de R dentro de un documento R Markdown se indican de la manera siguiente: Se abren tres comillas ‘, luego, entre corchetes {} se escribe el lenguaje; después el código y se finaliza con otras tres comillas ‘, que resulta en

```
sqrt(2)-exp(-2)
```

```
## [1] 1.278878
```

Vamos a calcular $\sqrt{2} - e^{-2}$:

```
sqrt(2)-exp(-2)
```

```
## [1] 1.278878
```

Etiquetas en chunks

Podemos etiquetar el chunk con algún nombre después del lenguaje indicado en los corchetes como {r MiPrimerChunk}. NO se pueden repetir las etiquetas o labels ya que R marcará error.

```
sqrt(2)-exp(-2)
```

```
## [1] 1.278878
```

Parámetros de los chunks

echo:

Con TRUE se muestra el código fuente, con FALSE no. {r MiPrimerChunk, echo=TRUE}:

- TRUE:

```
sqrt(2)-exp(-2)
```

```
## [1] 1.278878
```

- FALSE:

```
## [1] 1.278878
```

eval:

Con TRUE se evalúa el código fuente, con FALSE no. `{r MiPrimerChunk, echo=TRUE, eval=TRUE}`:

- TRUE:

```
sqrt(2)-exp(-2)
```

```
## [1] 1.278878
```

- FALSE:

`{r MiPrimerChunk, eval=TRUE}`:

- TRUE:

```
sqrt(2)-exp(-2)
```

```
## [1] 1.278878
```

- FALSE:

```
sqrt(2)-exp(-2)
```

message:

Con TRUE se muestran los mensajes que R produce al ejecutar el código fuente, con FALSE no. `{r MiPrimerChunk, message=TRUE}`:

- TRUE:

```
library(magic)
```

```
## Loading required package: abind
```

```
magic(6)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]    7    6   35   34   15   14
## [2,]    8    5   33   36   16   13
## [3,]   27   26   19   18   11   10
## [4,]   25   28   20   17    9   12
## [5,]   23   22    3    2   31   30
## [6,]   21   24    1    4   29   32
```

- FALSE:

```
library(magic)
magic(6)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]    7    6   35   34   15   14
## [2,]    8    5   33   36   16   13
## [3,]   27   26   19   18   11   10
## [4,]   25   28   20   17    9   12
## [5,]   23   22    3    2   31   30
## [6,]   21   24    1    4   29   32
```

warning:

Con TRUE se muestran los mensajes de advertencia que R produce al ejecutar el código fuente, con FALSE no. `{r MiPrimerChunk, warning=TRUE}`.

comment:

Con NA se omiten los ## en los mensajes que R produce al ejecutar el código fuente, con FALSE no. `{r MiPrimerChunk, comment=NA}`:

```
library(magic)
magic(6)
```

```
      [,1] [,2] [,3] [,4] [,5] [,6]
[1,]    7    6   35   34   15   14
[2,]    8    5   33   36   16   13
[3,]   27   26   19   18   11   10
[4,]   25   28   20   17    9   12
[5,]   23   22    3    2   31   30
[6,]   21   24    1    4   29   32
```

result:

Se encarga de darle formato al mensaje del código evaluado por R de acuerdo a la opción escogida. `{r MiPrimerChunk, echo=TRUE, results='option'}`:

- **markup**: Es el valor por defecto, muestra los resultados en el documento final línea a línea, encabezados por ##. `{r MiPrimerChunk, echo=TRUE, results='markup'}`.
- **hide**: No se nos muestra el resultado en el documento final, pero sí se evalúa el código, a diferencia de **eval**, el cual no necesariamente ejecuta el código. `{r MiPrimerChunk, echo=TRUE, results='hide'}`.
- **asis**: Devuelve de manera literal los resultados línea a línea en el documento final y el programa con el que se abre el documento final los interpreta como texto y formatea adecuadamente. `{r MiPrimerChunk, echo=TRUE, results='asis'}`.
- **hold**: Muestra todos los resultados al final del bloque de código. `{r MiPrimerChunk, echo=TRUE, results='hold'}`.

Ejemplos:

- **markup:**

```
sec=10:20
sec
```

```
## [1] 10 11 12 13 14 15 16 17 18 19 20
```

```
cumsum(sec)
```

```
## [1] 10 21 33 46 60 75 91 108 126 145 165
```

- **hide:**

```
sec=10:20
sec
cumsum(sec)
```

- **asis:**

```
sec=10:20
sec
```

```
[1] 10 11 12 13 14 15 16 17 18 19 20
```

```
cumsum(sec)
```

```
[1] 10 21 33 46 60 75 91 108 126 145 165
```

- **hold:**

```
sec=10:20
sec
cumsum(sec)
```

```
## [1] 10 11 12 13 14 15 16 17 18 19 20
```

```
## [1] 10 21 33 46 60 75 91 108 126 145 165
```

En cualquier chunk podemos usar en conjunto los distintos parámetros vistos anteriormente.

Chunk en linea

Para introducir una parte de código dentro de un párrafo y que se ejecute al comilarse el documento mostrando así el resultado final, hay que hacerlo utilizando abriendo una comilla, ‘, al inicio y al final del chunk; seguida de la primera comilla se indica el lenguaje: ‘r codigo‘

Ejemplos:

- En L^AT_EX: $\sqrt{2}$
- En R haciendo 1.4142136

- La frase completa: $\sqrt{2} = 1.4142136$.
- El número π empieza por 3.14159.
- La raíz cuadrada de 64 es 8 o, lo que viene siendo lo mismo, $\sqrt{64} = 8$.
- La raíz quinta de 32 es 2 o, lo que viene siendo lo mismo, $\sqrt[5]{64} = 32^{(1/5)}$.