Contenido

[1. Autores del trabajo, planificación y entrega 3](#_Toc8072135)

[1.1 Autores 3](#_Toc8072136)

[1.2 Planificación 3](#_Toc8072137)

[1.3 Entrega 3](#_Toc8072138)

[2. Requisitos del prototipo a implementar 4](#_Toc8072139)

[2.1 Requisitos funcionales 4](#_Toc8072140)

[2.2 Otros requisitos 4](#_Toc8072141)

[3.1 Criterio 1:Facilidad de uso 4](#_Toc8072142)

[3.2 Criterio 2: Documentación disponible. 5](#_Toc8072143)

[3.3 Criterio 3: Tiempo invertido 5](#_Toc8072144)

[3.4 Criterio 4: Facilidad en la instalación 5](#_Toc8072145)

[3.5 Criterio 5: Rendimiento 5](#_Toc8072146)

[3.6 Criterio 6: Optimización 5](#_Toc8072147)

[4. Proyecto de implementación de un prototipo del sistema utilizando la tecnología Ggplot2 6](#_Toc8072148)

[4.1 Documentación de diseño 6](#_Toc8072149)

[4.2 Documentación de construcción 6](#_Toc8072150)

[**Gráfico de barras** 6](#_Toc8072151)

[**Gráfico de pastel** 8](#_Toc8072152)

[**Histograma** 10](#_Toc8072153)

[**Gráfico de densidad de Kernel** 11](#_Toc8072154)

[**Gráfico de línea** 12](#_Toc8072155)

[**Diagrama de cajas** 12](#_Toc8072156)

[Nube de palabras 13](#_Toc8072157)

[4.3 Documentación de pruebas 14](#_Toc8072158)

[4.4 Documentación de instalación 15](#_Toc8072159)

[4.5 Manual de usuario 16](#_Toc8072160)

[5. Proyecto de implementación de un prototipo del sistema utilizando la tecnología Googlevis 17](#_Toc8072161)

[5.1 Documentación de diseño 17](#_Toc8072162)

[5.2 Documentación de construcción 18](#_Toc8072163)

[**Gráfico de barras** 19](#_Toc8072164)

[**Gráfico de columnas** 19](#_Toc8072165)

[**Gráfico de pastel** 20](#_Toc8072166)

[**Histograma** 20](#_Toc8072167)

[**Gráfico de líneas** 21](#_Toc8072168)

[**Diagrama de cajas** 21](#_Toc8072169)

[5.3 Documentación de pruebas 22](#_Toc8072170)

[5.4 Documentación de instalación 23](#_Toc8072171)

[5.1.1. Paso 0 23](#_Toc8072172)

[4.1.2. Paso 1 24](#_Toc8072173)

[5.1.3. Paso 2 25](#_Toc8072174)

[5.1.4. Paso 3 26](#_Toc8072175)

[5.1.5. Paso 4 26](#_Toc8072176)

[5.1.6. Paso 5 (opcional) 26](#_Toc8072177)

[5.5 Manual de usuario 27](#_Toc8072178)

[5.1.7. Creación de una tabla de datos 28](#_Toc8072179)

[5.1.8. Gráficas. Ejemplo 1 28](#_Toc8072180)

[5.1.9. Gráficas. Ejemplo 2 28](#_Toc8072181)

[5.1.10. Funcionalidades 29](#_Toc8072182)

[5.1.10.1. Zoom 29](#_Toc8072183)

[5.1.10.2. Información adicional 29](#_Toc8072184)

[5.1.10.3. Seleccionar información por conjuntos 29](#_Toc8072185)

[5.1.10.4. Consultar cambios en las versiones de software 29](#_Toc8072186)

[6. Comparación de las dos implementaciones 31](#_Toc8072187)

[6.1 Evaluación de los criterios en la implementación usando el framework Ggplot2 31](#_Toc8072188)

[6.2 Evaluación de los criterios en la implementación usando la tecnología GoogleVis 31](#_Toc8072189)

[7. Comparación de la implementación de las tecnologías 32](#_Toc8072190)

[8. Conclusiones 32](#_Toc8072191)

# 1. Autores del trabajo, planificación y entrega

## 1.1 Autores

Grupo T6

Julia Martín Moracho (Coordinadora)

Javier Pascual Marzo

Marcos Vicente Díez

Robert Alejandro Hernández Andrade

Germán Dengra Rísquez

## 1.2 Planificación

En este apartado se debe incluir copias de pantalla de la planificación del trabajo con diagramas Gantt: o bien un enlace (URL) a la web donde esté disponible la planificación si se ha utilizado una herramienta online de diagramación Gantt (por ejemplo, [Teamweek](https://teamweek.com/free-online-gantt-chart.html), [GanttPro](https://ganttpro.com/), [tomsplanner](https://plan.tomsplanner.es/), [sinnaps](https://www.sinnaps.com/), u otra).

Hay que tener en cuenta que cada participante del grupo debe tener asignadas tareas que sumen al menos 45 horas. El peso de este trabajo en la calificación total de la asignatura es de un 30%, por tanto requiere de una dedicación de 45 horas del total de 150 horas de la asignatura.

## 1.3 Entrega

En este apartado debe incluirse un enlace (URL) a un repositorio en GitHub creado para el trabajo.

<https://github.com/juliamarmo/TG6>

En dicho repositorio debe encontrarse, al menos los siguientes archivos en la rama máster:

* Informe del trabajo: con el nombre TG3\_final.docx
* Presentación del trabajo: TG3\_final.pptx
* Prototipos obtenidos implementando cada una de las tecnologías (deben incluir el código fuente y todos los archivos necesarios para la instalación y uso de cada prototipo):
  + PrototipoTecnologiaA\_final.zip (o .rar)
  + PrototipoTecnologiaB\_final.zip (o .rar).

Dichos archivos serán los que se tendrán en cuenta para la calificación del trabajo.

# 2. Requisitos del prototipo a implementar

## 2.1 Requisitos funcionales

Los requisitos funcionales deben ser los mismos para las dos implementaciones.

En la siguiente tabla se indicará el catálogo de requisitos funcionales del sistema.

|  |  |
| --- | --- |
| **REQ.** | **DESCRIPCIÓN** |
| RF01 | Se deberán mostrar diferentes niveles de granularidad. |
| RF02 | Se deberá poder aumentar el nivel de abstracción. |
| RF03 | Se deberán poder mostrar dependencias y relaciones entre los datos de forma clara. |
| RF04 | Se permitirá hacer zoom. |
| RF05 | Se mostrará información adicional al pasar el ratón por encima de elementos. |
| RF06 | Implementar checkboxes para filtrar la información se desee mostrar. |
| RF07 | Se podrá analizar y mostrar los datos de diferentes maneras, entre ellas en *conjuntos.* |
| RF08 | Se podrá analizar y mostrar los datos de diferentes maneras, entre ellas en *clusters.* |
| RF09 | Se podrá analizar y mostrar los datos de diferentes maneras, entre ellas *ordenados.* |
| RF10 | Se podrán comparar cambios cada cierto tiempo y entre versiones diferentes del software. |

## 2.2 Otros requisitos

Se pueden incluir aquí otros requisitos para el prototipo que no puedan considerarse como funcionales. Por ejemplo, requisitos de datos, de seguridad, de interfaz de usuario, de rendimientos, etc.

En la siguiente tabla se indicará el catálogo de requisitos no funcionales del sistema.

|  |  |
| --- | --- |
| **REQ.** | **DESCRIPCIÓN** |
| RNF01 | Se podrá cambiar el color y la representación de los datos. |
| RNF02 | Se mostrarán los datos de una manera sencilla que facilite la comprensión. |
| RNF03 | El framework utilizara el lenguaje de programación R |
| RNF04 | El framework será open source |

**3. Criterios de comparación en la implementación**

## 3.1 Criterio 1:Facilidad de uso

Nombre del criterio: Facilidad de uso.

Descripción: Dificultad a la hora de cumplir los requisitos presentados.

Tipo valor: Numérico (Del 0 al 10)

## 3.2 Criterio 2: Documentación disponible.

Nombre del criterio: Documentación disponible.

Descripción: Documentación disponible del framework.

Tipo valor: Cualitativo.

## 3.3 Criterio 3: Tiempo invertido

Nombre del criterio: Tiempo invertido en la creación de los gráficos.

Descripción: Tiempo en minutos que se ha invertido en la realización de los grafos propuestos.

Tipo valor: Numérico (minutos).

## 3.4 Criterio 4: Facilidad en la instalación

Nombre del criterio: Facilidad en la instalación.

Descripción: Facilidad a la hora de instalar el framework.

Tipo valor: Numérico (0 al 10).

## 3.5 Criterio 5: Rendimiento

Nombre del criterio: Rendimiento

Descripción: Cómo de rápido se generan los grafos programados.

Tipo valor: Numerico (del 0 al 10)

## 3.6 Criterio 6: Optimización

Nombre del criterio: Optimización

Descripción: Se valora positivamente si el framework rinde igual de bien en ordenadores con diferentes especificaciones.

Tipo valor: Númerico (del 0 al 10)

# 4. Proyecto de implementación de un prototipo del sistema utilizando la tecnología Ggplot2

## 4.1 Documentación de diseño

## 4.2 Documentación de construcción

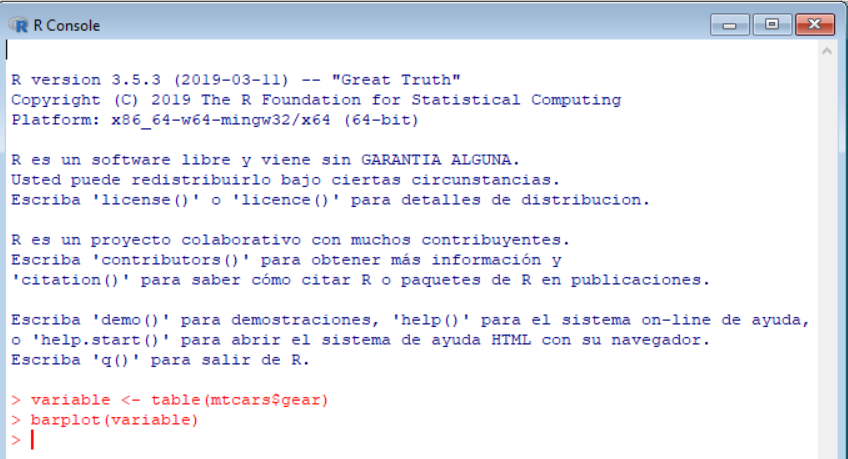
En este apartado trataremos de implementar la visualización de datos (Data Visualization). Concretamente trataremos de implementar todos los gráficos básicos que existen, los cuales son los siguientes:

* Gráfico de barras (Bar chart)
* Gráfico de pastel (Pie chart)
* Historgrama (Histogram)
* Gráfico de densidad del núcleo (kernel density plot)
* Gráfico de línea (Line chart)
* Diagrama de caja (Box plot)
* Nube de palabras (Word cloud)

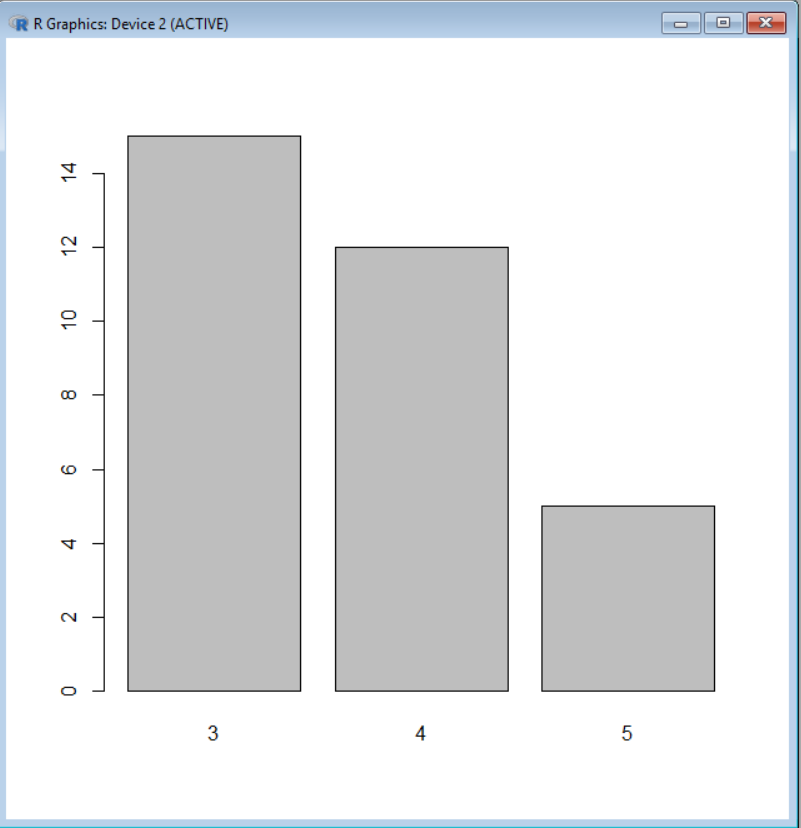
En cada uno de estos gráficos incluiremos tanto el resultado que hemos obtenido como el código que no lo ha generado, tratando de explicar cada una de las variables que hemos definido.

### **Gráfico de barras**

Éste grafico de visualización de datos es de los más simples y de los que más solemos utilizar, para crearlo únicamente tenemos que asignarle a una variable la tabla “mtcars$gear”, la cual lleva unos números por defecto que nos crea el gráfico que queremos. La implementación sería la siguiente:



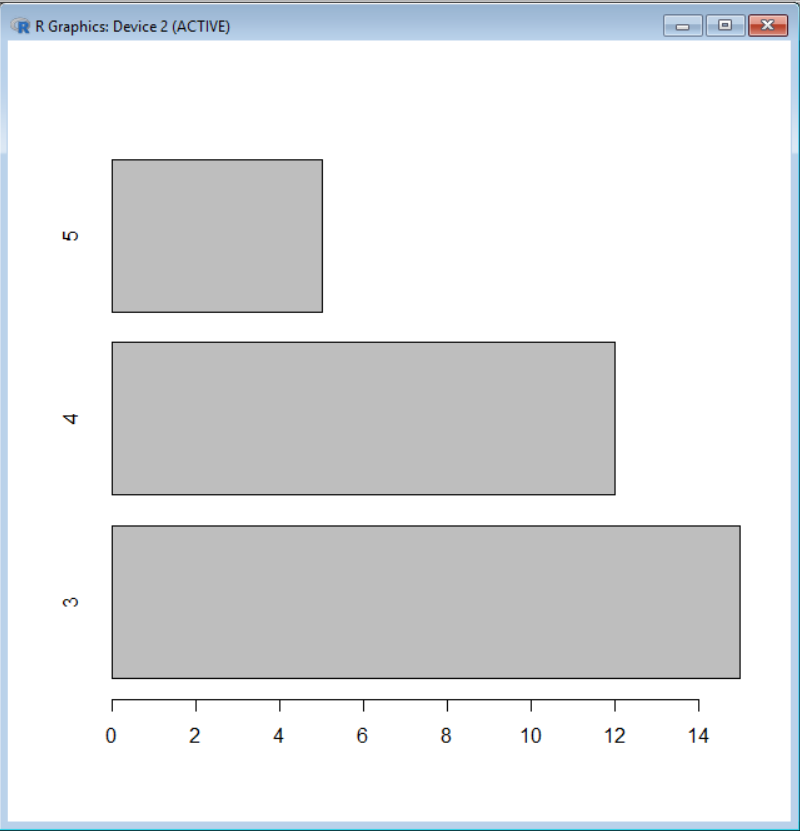
Una vez introducimos el barplot de la variable se nos crea el gráfico de barras:



Este tipo de gráfico lo podemos poner en horizontal añadiéndole al barplot “horiz= TRUE”, quedaría, por tanto, de la siguiente manera:

* Barplot(variable,horiz = TRUE)

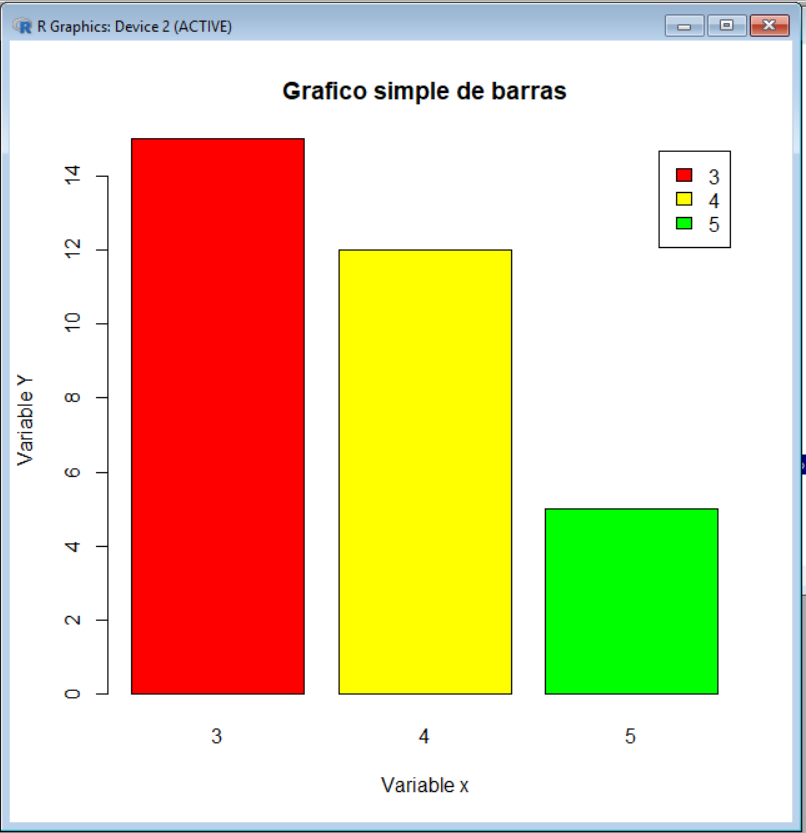
Éste sería el resultado.



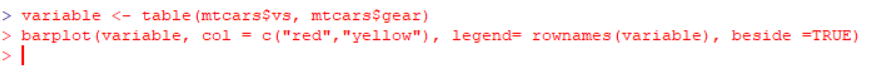
A este tipo de gráficos podemos asignarle colores, títulos, leyendas, etc. Un ejemplo más completo sería el siguiente:

* barplot(variable, main= "Grafico simple de barras", xlab = "Variable x", ylab ="Variable Y", legend = rownames(variable), col = c("red","yellow","green"))

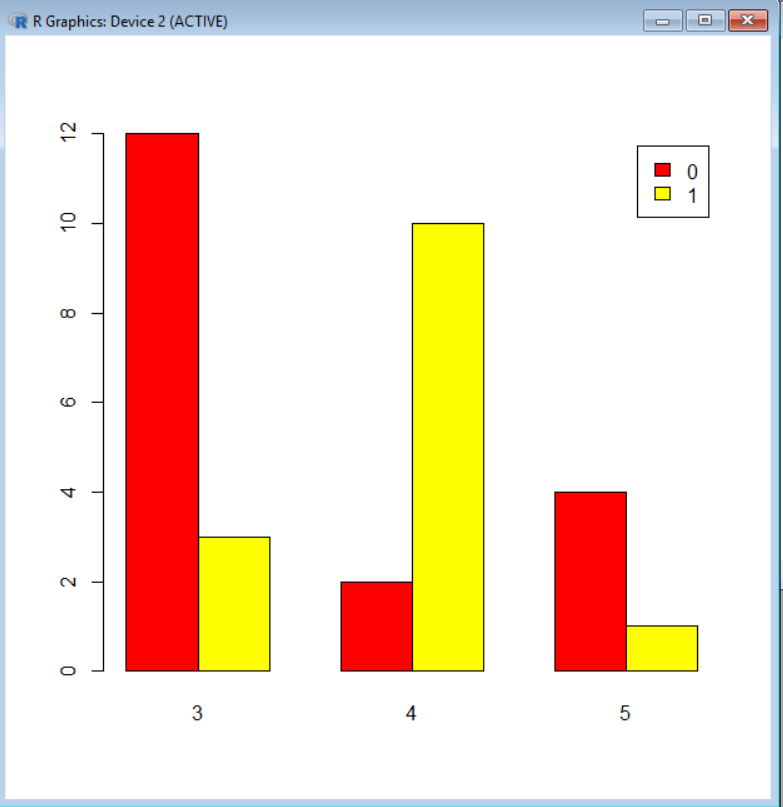
El resultado de eso sería lo siguiente:



En los gráficos de barras, además, pueden hacerse comparaciones o enfrentamientos, esto se implementaría de la siguiente manera:



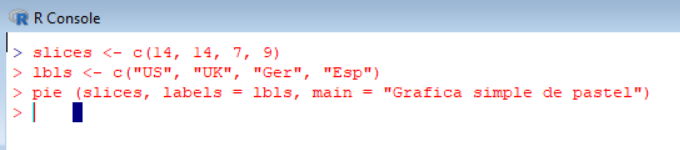
Y el resultado sería el siguiente:



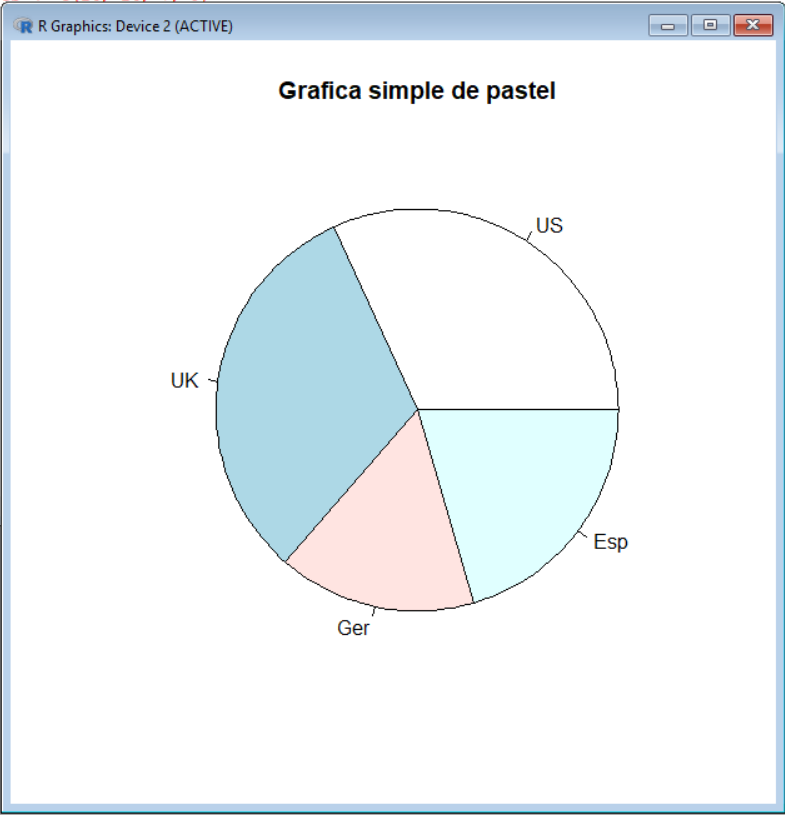
### **Gráfico de pastel**

Este tipo de gráfico también es bastante utilizado, en él un circulo se divide en sectores que representan una porción del todo. De ahí proviene “pastel”, ya que lo dividimos en cachos.

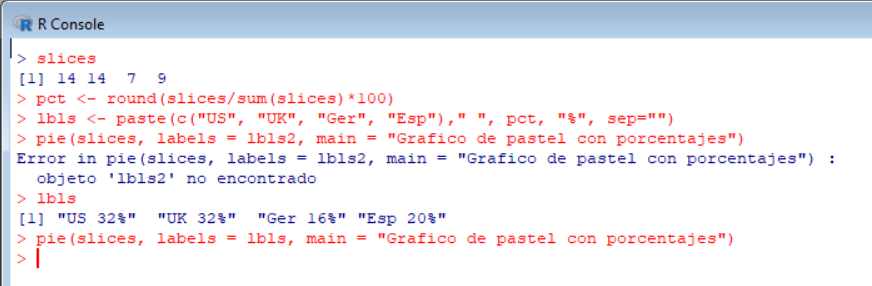
Un gráfico simple seria implementado de la siguiente forma:



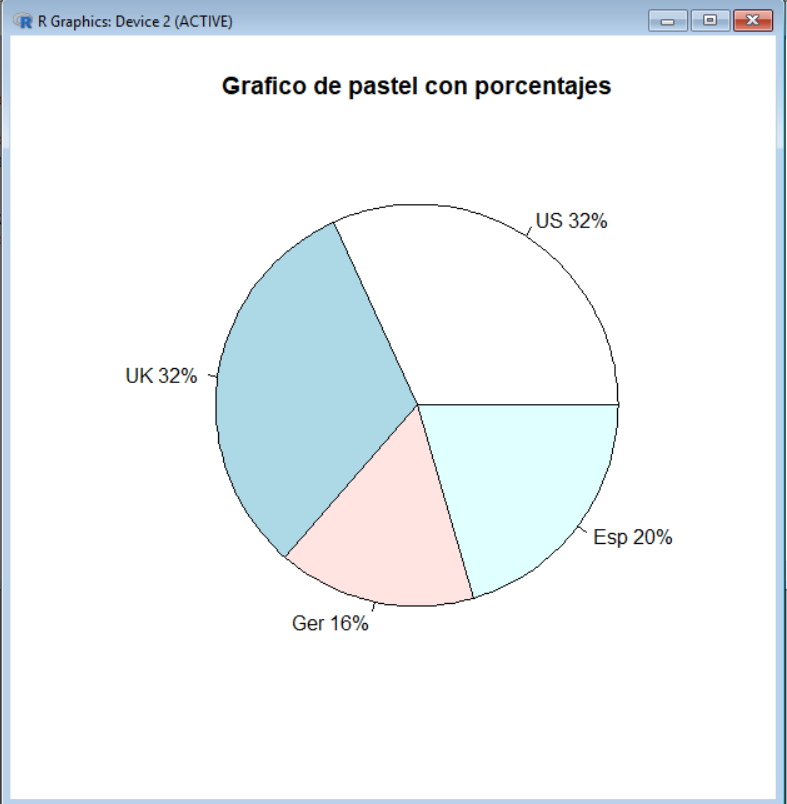
Lo cual tendría como resultado:



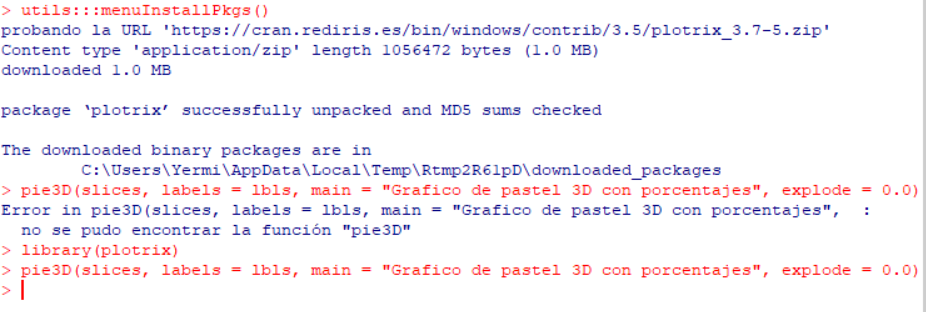
Si queremos añadir porcentajes al gráfico lo haríamos de la siguiente manera:



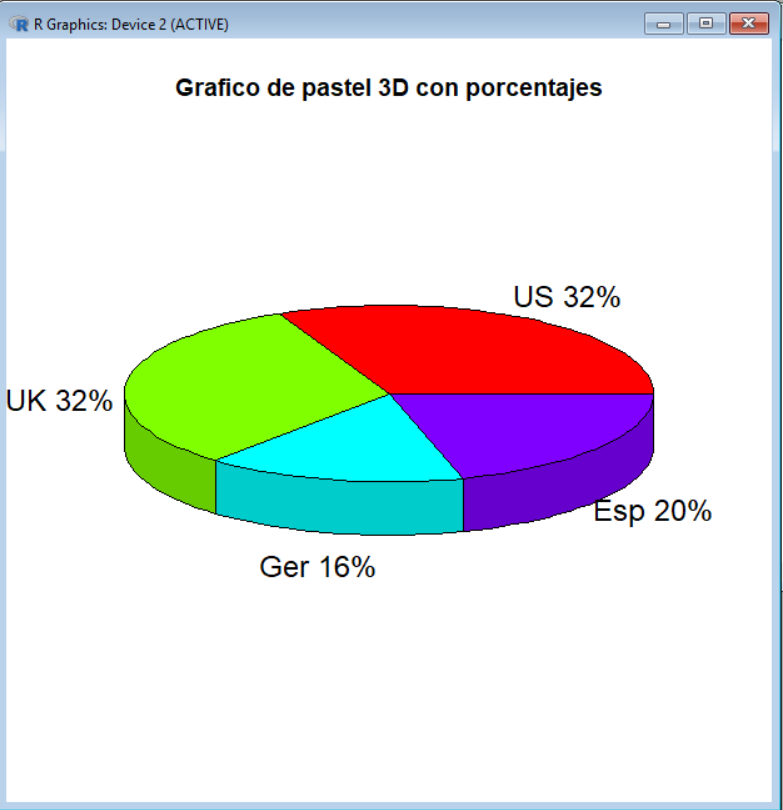
Y el resultado sería el siguiente:



Finalmente, si queremos realizar este gráfico en 3D, tendríamos que instalar el paquete de plotrix, seguidamente llamar a la librería plotrix, y después lanzar la sentencia que lo ejecuta.



El resultado sería el siguiente:

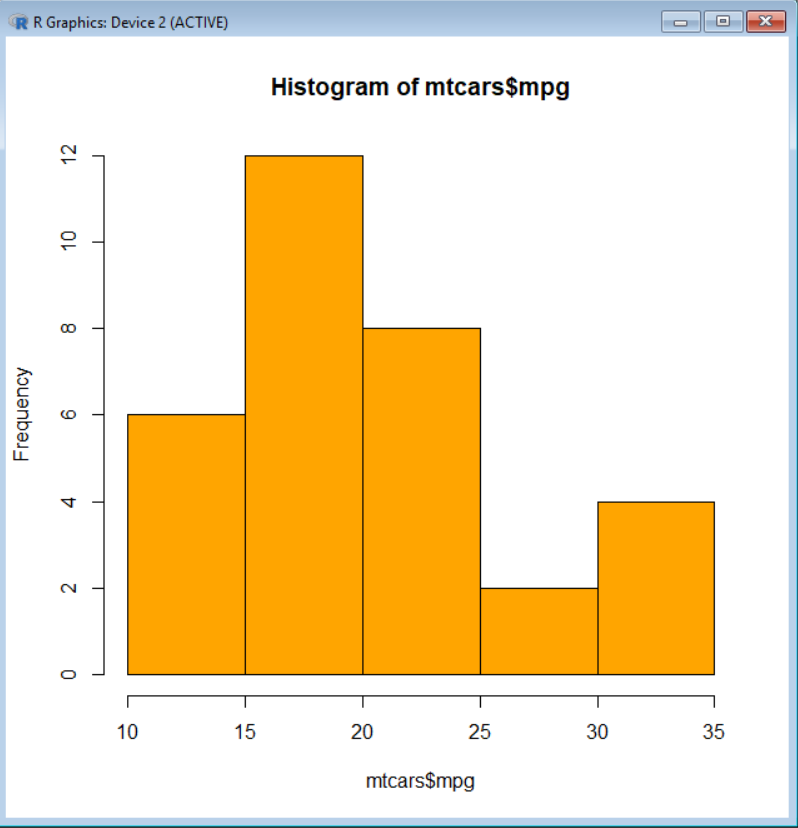


### **Histograma**

Un histograma es un gráfico de la representación de distribuciones de frecuencias, en él hacemos uso de rectángulos dentro de unas coordenadas. Para hacer este tipo de gráfico en R haremos uso del siguiente código:



Como resultado obtendremos un histograma de la siguiente forma:

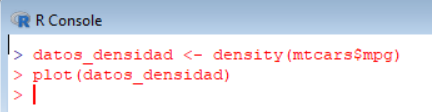


Este tipo de gráfico no puede ser representado en 3D y el histograma esta creado sobre un vector básico que nos ofrece R para observarlo.

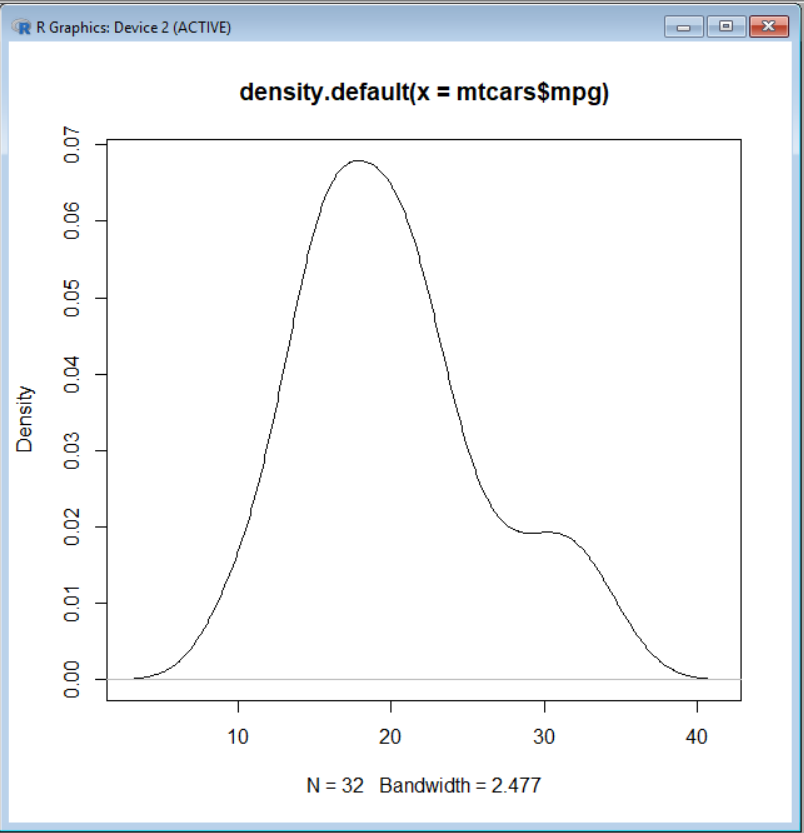
### **Gráfico de densidad de Kernel**

En este tipo de gráficos visualizamos la distribución de datos en un intervalo o periodo de tiempo continuo. Este gráfico es muy similar al histograma solamente que con este utilizamos un suavizado para trazar valores, de esta forma tenemos distribuciones más suaves.

Para elaborar este tipo de gráfico en R utilizamos el siguiente código, el cual nos servirá de ejemplo:



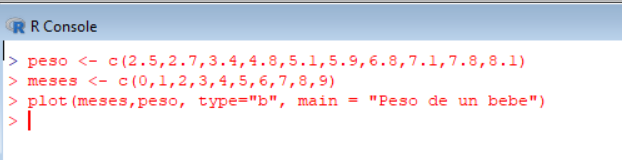
El resultado visual es el siguiente:



### **Gráfico de línea**

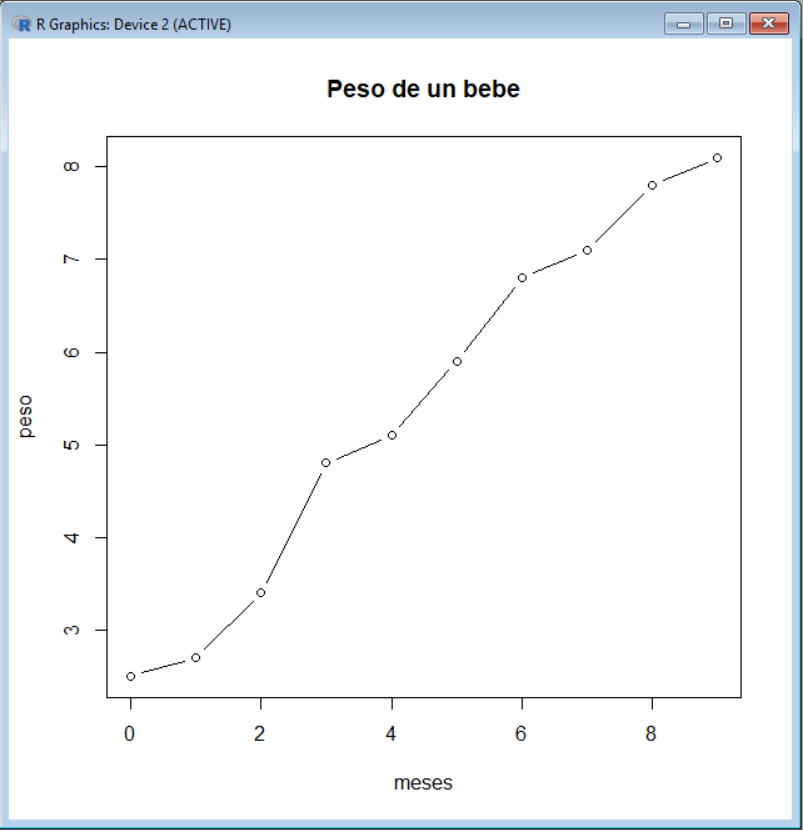
El gráfico de línea es un gráfico muy utilizado tanto en el ámbito escolar como en el ámbito de la estadística. En este tipo de gráficos señalamos con un punto los datos que nos dan y posteriormente unimos esos puntos con una línea fina.

Para representar este tipo de gráfico haremos uso del siguiente código:



En este caso definimos 2 variables, peso que seria la variable “y” y meses que sería la variable “x”, como ambos parámetros tienen un vector de la misma longitud (10), se representa un punto donde ambos coinciden y se unen con una línea, como ya he explicado anteriormente.

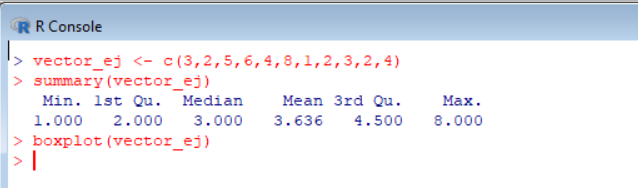
El resultado de ese código daría la siguiente gráfica:



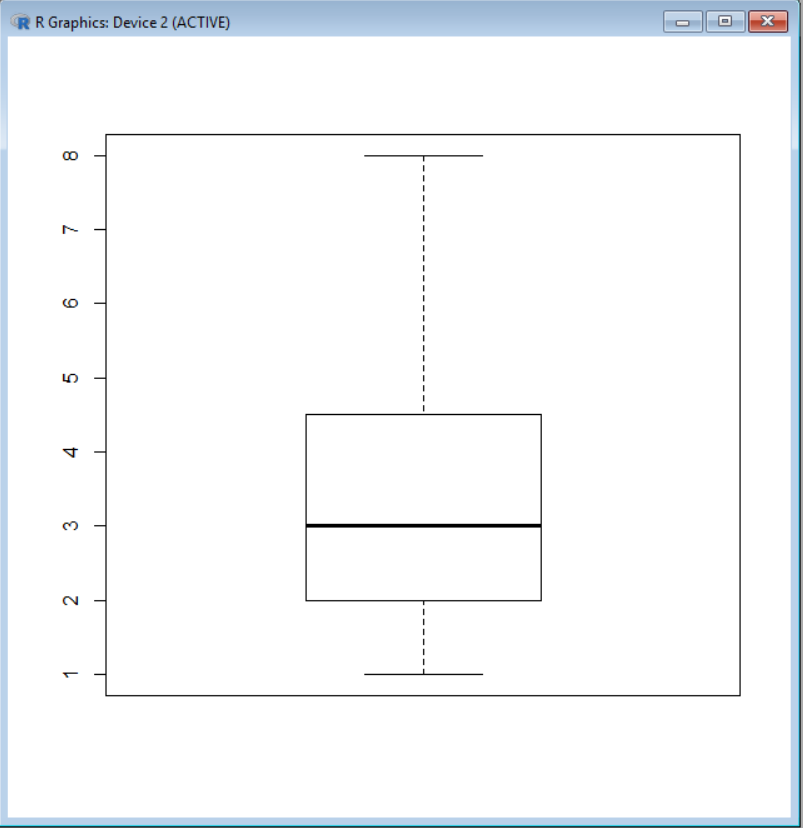
### **Diagrama de cajas**

El diagrama de cajas quizás es de los menos conocidos de todos los que hemos hablado, en ellos tratamos de representar gráficamente una serie de datos numéricos a través de sus cuartiles. De esta forma podemos ver fácilmente los cuartiles, la mediana, etc.

Para la representación en R de un diagrama de cajas usaremos este código:



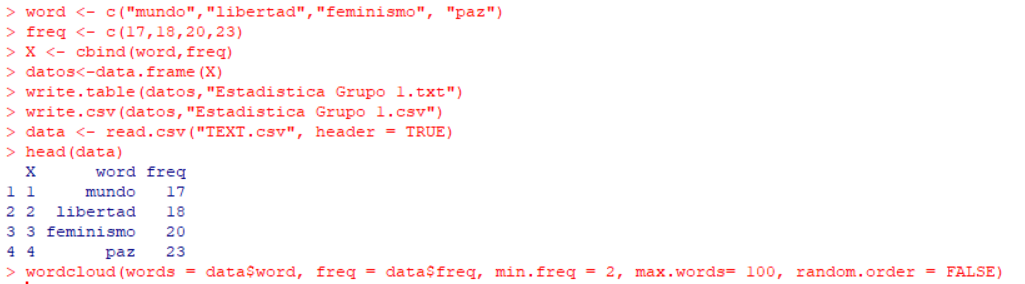
Le incluimos un vector, después sacamos los datos sobre ella y finalmente la representamos:



Como podemos ver, este gráfico representa los valores del vector que hemos introducido, de este grafico podemos sacar fácilmente que el mínimo es 1, el máximo es 8, el valor 2 es el primer cuartil (25% de los datos), el valor 3 es el segundo cuartil o mediana (50% de los datos) y el 4,5 sería el tercer cuartil (75% de los datos).

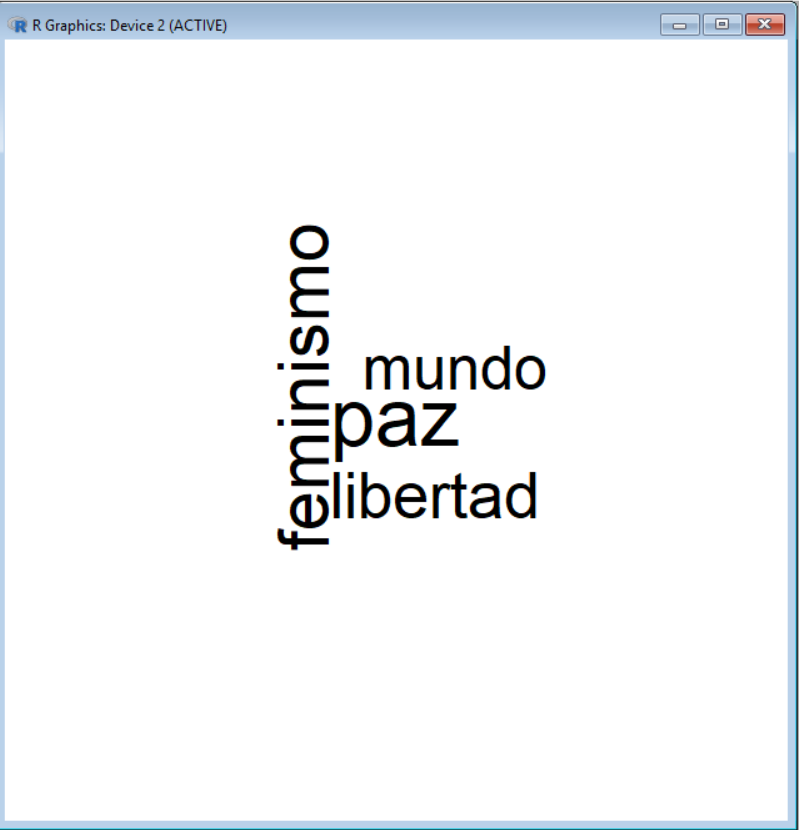
# Nube de palabras

Este tipo de gráfico es básicamente artístico a diferencia de todos los anteriores, tratamos de crear una imagen con un conjunto de palabras. El código que hemos utilizado en R es el siguiente:



Como podemos observar, hemos creado un CSV con dos columnas que son “Word” y “freq”. En la primera columna hemos metido las palabras que queríamos que apareciesen y en la segunda con la frecuencia con la que la queríamos. Además, hemos tenido que importar el paquete de “cloudword” ya que no viene preinstalado en R.

El resultado ha sido el siguiente:



Este tipo de gráfico ha sido de los más costosos ya que ha habido que hacer un CSV desde R porque no servía exportarlo del Excel.

## 4.3 Documentación de pruebas

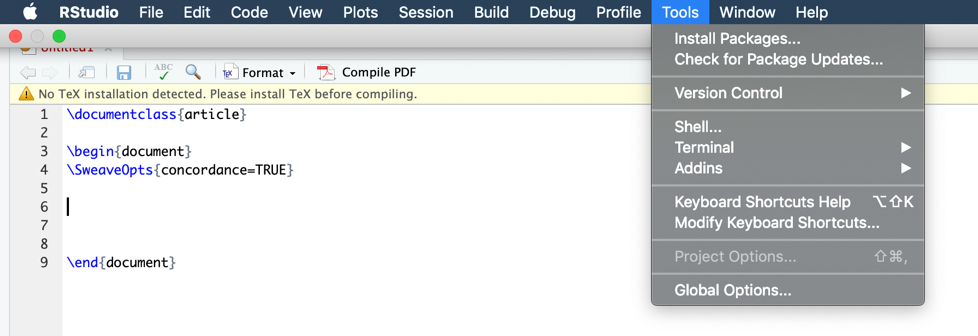
A continuación, vamos a ver los requisitos que se cumplen y los que no con el prototipo de visualización de datos con Ggplot2.

|  |  |  |  |
| --- | --- | --- | --- |
| **REQ.** | **DESCRIPCIÓN** | **√/**X | **Observaciones** |
| RF01 | Se deberán mostrar diferentes niveles de granularidad. | X | No se puede hacer en Ggplot2 |
| RF02 | Se deberá poder aumentar el nivel de abstracción. | X | - |
| RF03 | Se deberán poder mostrar dependencias y relaciones entre los datos de forma clara. | **√** | Se ven claramente las dependencias en los gráficos |
| RF04 | Se permitirá hacer zoom. | **√** | - |
| RF05 | Se mostrará información adicional al pasar el ratón por encima de elementos. | X | No se muestra información adicional, es una imagen estática que se genera |
| RF06 | Implementar checkboxes para filtrar la información se desee mostrar. | X | No se puede hacer con Ggplot2 |
| RF07 | Se podrá analizar y mostrar los datos de diferentes maneras, entre ellas en *conjuntos.* | **√** |  |
| RF08 | Se podrá analizar y mostrar los datos de diferentes maneras, entre ellas en *clusters.* | **√** | Se puede hacer visualización de clusters. Pero no se ha hecho en este prototipo concreto |
| RF09 | Se podrá analizar y mostrar los datos de diferentes maneras, entre ellas *ordenados.* | **√** | A través de comandos el usuario puede cambiar la configuración predefinida |
| RF10 | Se podrán comparar cambios cada cierto tiempo y entre versiones diferentes del software. | **√** | Nos debemos meter a ver las características de cada versión en internet. |
| RNF01 | Se podrá cambiar el color y la representación de los datos. | **√** | Con (theme) podremos cambiar la configuración de colores |
| RNF02 | Se mostrarán los datos de una manera sencilla que facilite la comprensión. | **√** | Predomina la simplicidad |
| RNF03 | El framework utilizara el lenguaje de programación R | **√** | Ggplot2 es una librería de visualización de datos en R |
| RNF04 | El framework será open source | **√** | Ggplot2 es Open Source |

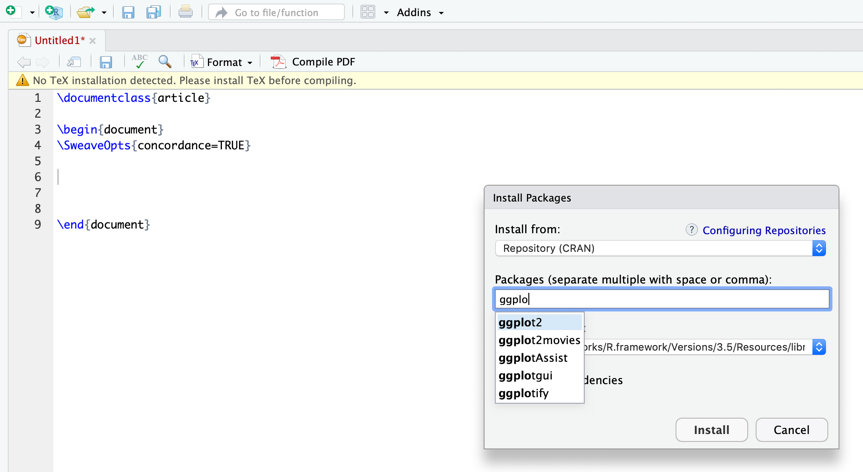
## 4.4 Documentación de instalación

Para la realización de el prototipo vamos a hacer uso de Rstudio y más concretamente, la librería Ggplot2 para la realización de las gráficas para su posterior comparación.

A continuación, vamos a explicar la instalación de la librería en cuestión. Lo primero de todo será instalar y abrir la aplicación de Rstudio, y después de esto nos deberemos dirigir al apartado de Tool e install packages como aparece en la captura inferior.



Nos aparecerá la pantalla emergente que vemos en la captura inferior, y deberemos introducir por teclado el nombre de la librería que queremos incluir. Después de esto instalamos y vemos por la consola si se ha instalado correctamente.







A partir de aquí ya podemos empezar a trabajar.

## 4.5 Manual de usuario

Ggplot2 es una librería de R. En el punto anterior hemos explicado su instalación. Esta librería se usa a través de la consola de RStudio.

Para hacer los diversos gráficos que hemos explicado anteriormente, solo debemos introducir el código correspondiente al gráfico que queremos implementar. Toda esta información aparece en el apartado de implementación de el prototipo.

Gráfico de barras: Con los siguientes comandos podemos hacer un gráfico de barras coloreados.

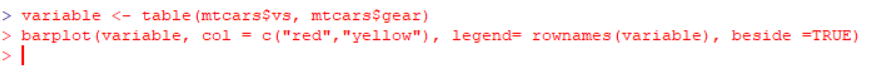
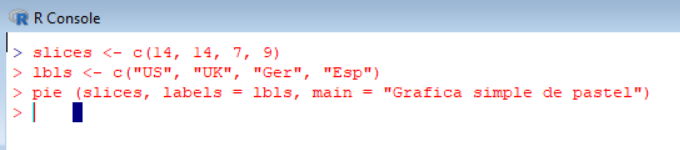


Gráfico pastel: dentro de las variables slices y lbls vamos a introducir los datos.



Histograma:



Gráfico de densidad Kernel:

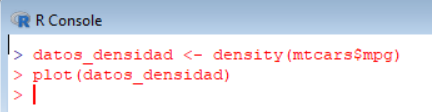


Gráfico de líneas:

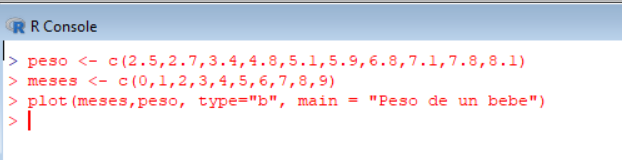
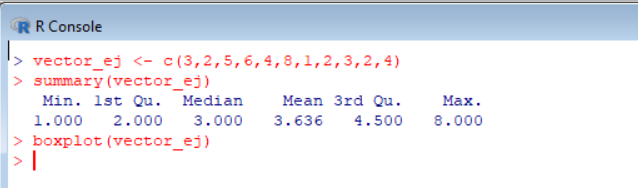
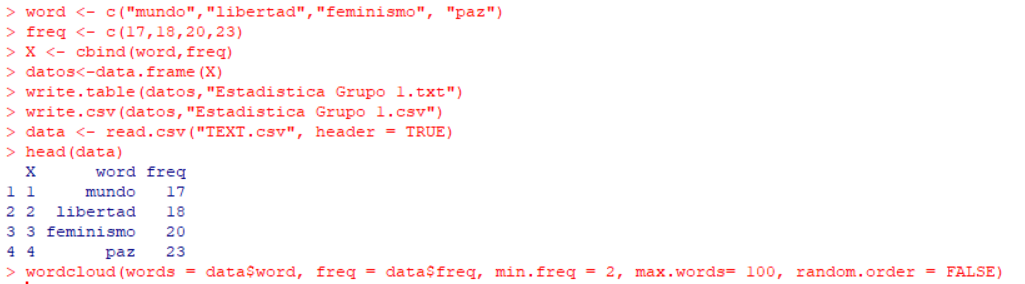


Gráfico de caja y bigotes:



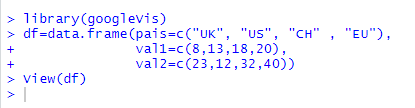
Nube de palabras: Si queremos que aparezca otra nube de palabras, las deberemos cambiar dentro del código.



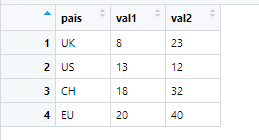
# 5. Proyecto de implementación de un prototipo del sistema utilizando la tecnología Googlevis

## 5.1 Documentación de diseño

En cuanto al diseño, para la representación de distintos tipos de grafos, lo primero que haremos será crear una variable que va a contener los datos que utilizaremos para su representación gráfica, introduciremos los datos de tal manera que se guarden en forma de tabla. Hay que destacar que esta tabla no se podrá utilizar en todos los ejemplos por lo que usaremos otro tipo de datos.

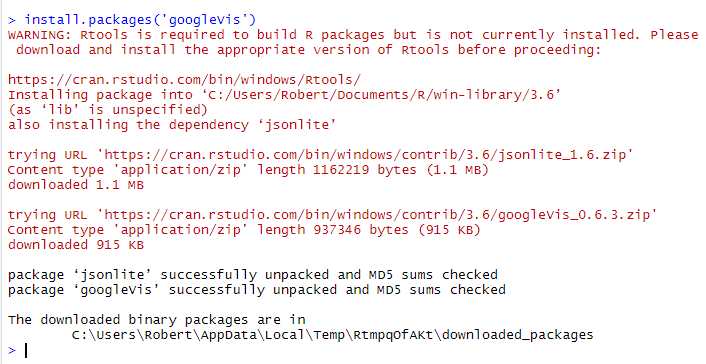


Aquí podemos observar la estructura de datos que contiene la variable creada (df).

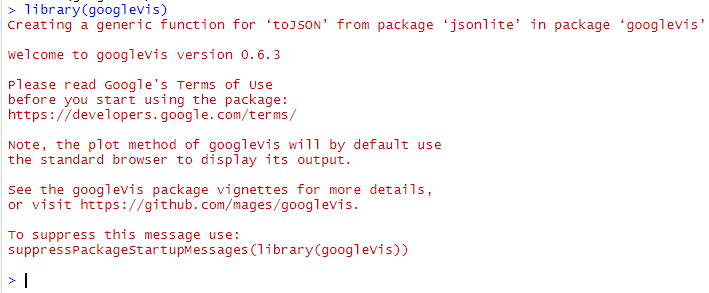


## 5.2 Documentación de construcción

En este apartado lo primero que hay que hacer es instalar el paquete GoogleVis en RStudio desde el mismo intérprete.



Instalado el paquete siempre y que se vaya a usar la librería de GoogleVis hay que introducir la siguiente línea de código.



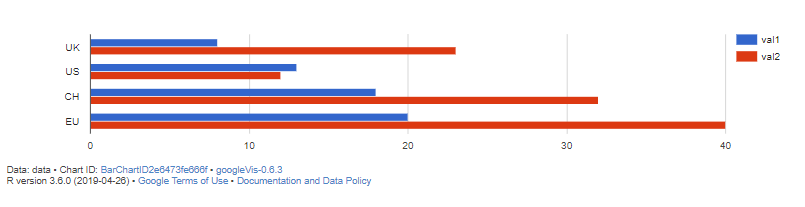
Observamos como nos permite ya programar en R con las librerías de GoogleVis disponibles.

### **Gráfico de barras**

Para realizar un gráfico de barras hará falta llamar la función correspondiente pasándole como argumento la tabla creada (gvisBarChart(df)). A continuación, lo único que quedaría por hacer es representar gráficamente (plot(Bar)).

https://lh3.googleusercontent.com/jbY2dtGUYZmsfPUMnamu1wBUraRDxFlaZN6N_PX5l7PnLtraYdt_5DHqDN-6w6N99FHLg28B748_oQwPfn7sYknSwqH-nYDIsxuBCw5b1IhJvQGxQZCMVOiVDKBuIrxnJpUW2gtnHbXQO8t5pQ

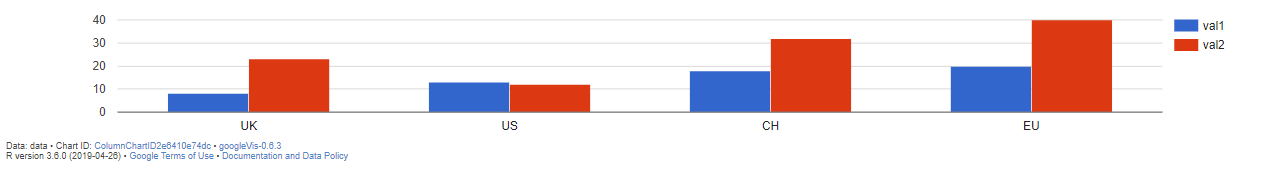
Obtenemos como resultado la siguiente gráfica:



### **Gráfico de columnas**

Para representar nuestra tabla utilizando un gráfico de columnas se sigue un procedimiento similar, lo único que varía es la función a la que hacemos la llamada (gvisColumnChart(df)).

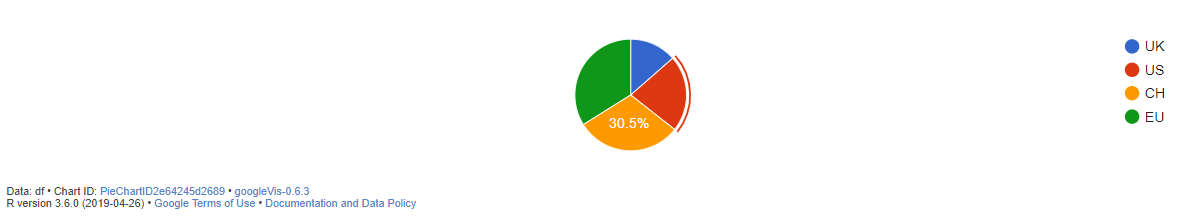
https://lh5.googleusercontent.com/aLqMd1_bw5s4e8JHsn1pY_k6sXcqhJ7S2LpkV0KBn9ekkxnTJhqNbZBTZhvddnnfHzP_byEKjRL84T815OXvMWOEDG_yCSAy0vec5eqEmcehn_sfthenJg5A1th7v3V8xQk-nd7nBZntUMVGmQ



### **Gráfico de pastel**

Cabe decir que en este gráfico hay imposibilidad para representar los dos valores correspondientes a cada país simultáneamente, por lo que la gráfica utiliza únicamente ‘val1’. Esta gráfica representa los valores en porcentajes sobre la suma total de los valores.

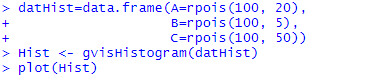
https://lh6.googleusercontent.com/APjHcIXFK5pONy2akVl-4Rm8w-Yu1Nz-6H1jKS5ovOZJh3IANbx1euBof28O_YlFhtghvhl0Qb3LZto6llT5nNr4pn65HeYAIvkaOTMcx3Dy7qXdQL4cmaTOjvPtuXT1VaNHAVLnZ8mVYiJR_Q



### **Histograma**

Para esta gráfica crearemos otra variable para que la interpretación de ésta se pueda apreciar de mejor manera.

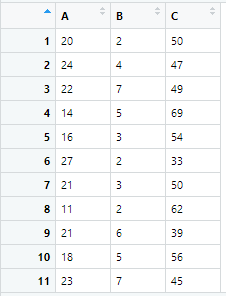
Almacenaremos en la variable creada una tabla que contiene 100 objetos diferentes con 3 atributos cada uno, en cada atributo se asigna un valor número, el resultado es una matriz [100 x 3].



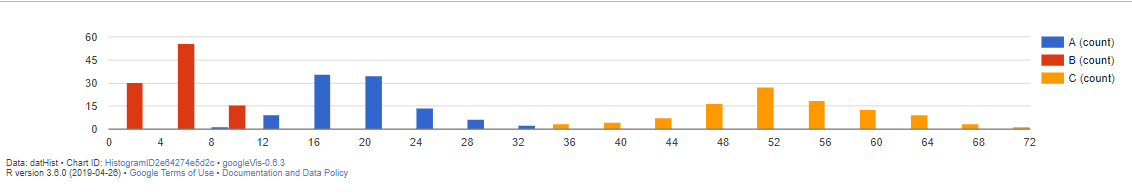
En A se generan valores entre 10-32.

En B generan valores entre 0-11.

En C generan valores entre 30-70.



Representamos gráficamente:



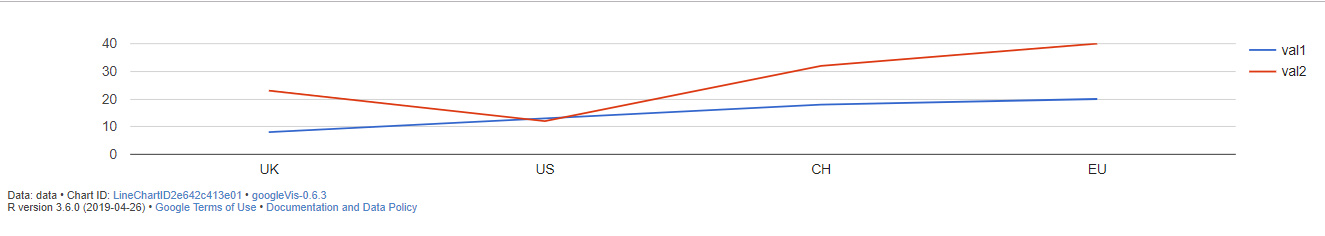
Observamos cómo se representa la tabla, en el eje de las ‘x’ se representan los valores y en el eje de las ‘y’ se representan las veces que se repite el valor, los atributos se reflejan por colores.

### **Gráfico de líneas**

La representación de este gráfico es muy sencilla, al igual que en los anteriores casos se utiliza una función para su representación, en este caso será ‘gvisLineChart(df)’.

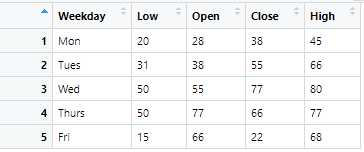
https://lh5.googleusercontent.com/hlW_hQ0gUzHXYG__enBY1jmB9JNPsrQ9QX7EjsZsphxhJp5KJpTPBe69HwklMLRWS13Y8K_Ni54jPBjAMo_uQGHRLXFcJZ6vYpih3-KcJroqooY1PqyXpXtzzcTGsFAZ9pBfsXMbv283EMfKAw

Observamos como representa los datos de la tabla sin problemas.



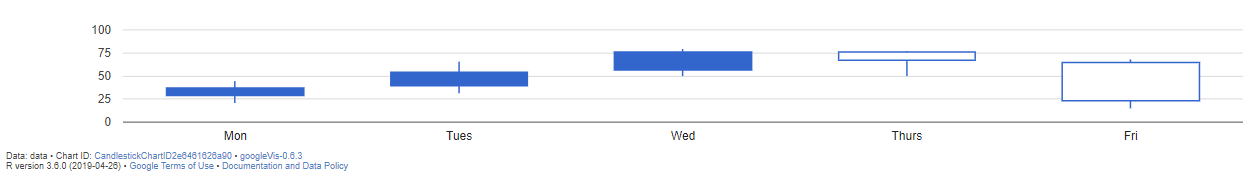
### **Diagrama de cajas**

Para esta gráfica haremos uso de una tabla ya disponible en las librerías de R (OpenClose):



Esta tabla nos proporciona diferentes valores para cada día de la semana.

https://lh5.googleusercontent.com/Dj-a5pfLDGk7VAPT8ER1FkMhZwX9obQUX62kmHXkVt_eZRLzz5KLm4fiDzTf1G1wE-qz-Lvm0Du7FTbMzJYjJYA-kCxm3kJ51R3uuAiEudLlA80QijONpiZFdqpleLmhwon-vdSNyIDqhjAz0g



Observamos que la gráfica resultante representa valores estadísticos como los mínimos y máximos de cada día.

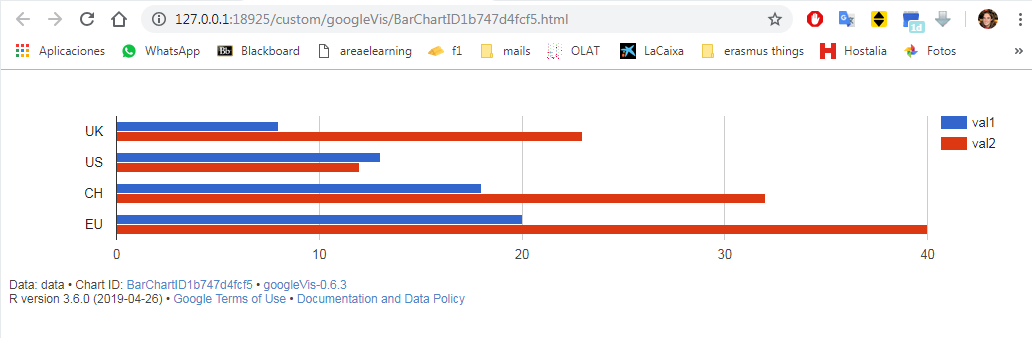
## 5.3 Documentación de pruebas

Cogeremos el caso 5.2.1 para ejecutar las pruebas en ellos. Recordamos que las pruebas tienen que superar los requisitos establecidos los puntos 2.1 y 2.2.

Para ello, ejecutamos el código del punto 5.2.1 (del gráfico de barras):



Y obtenemos lo siguiente:



Vamos pues, a tomar la tabla del punto 2, especificando que requisitos se cumplen y cuales no:

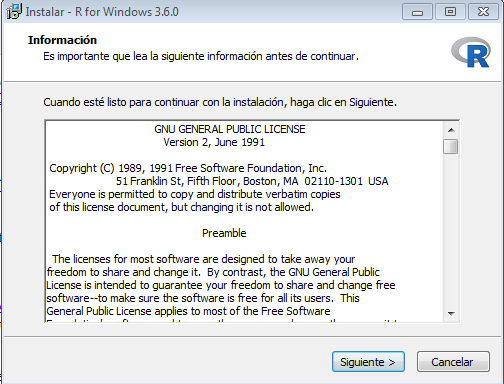
|  |  |  |  |
| --- | --- | --- | --- |
| **REQ.** | **DESCRIPCIÓN** | **√/**X | **Observaciones** |
| RF01 | Se deberán mostrar diferentes niveles de granularidad. | X | GoogleVis no permite esto |
| RF02 | Se deberá poder aumentar el nivel de abstracción. | X | GoogleVis no permite esto |
| RF03 | Se deberán poder mostrar dependencias y relaciones entre los datos de forma clara. | **√** | Se aprecia la relación entre datos perfectamente |
| RF04 | Se permitirá hacer zoom. | **√** | Google Chrome permite hacer zoom |
| RF05 | Se mostrará información adicional al pasar el ratón por encima de elementos. | **√** |  |
| RF06 | Implementar checkboxes para filtrar la información se desee mostrar. | X | GoogleVis no permite esto |
| RF07 | Se podrá analizar y mostrar los datos de diferentes maneras, entre ellas en *conjuntos.* | **√** | Si, seleccionando los colores de la parte derecha del gráfico |
| RF08 | Se podrá analizar y mostrar los datos de diferentes maneras, entre ellas en *clusters.* | X | GoogleVis no permite esto |
| RF09 | Se podrá analizar y mostrar los datos de diferentes maneras, entre ellas *ordenados.* | **√** | Por defecto |
| RF10 | Se podrán comparar cambios cada cierto tiempo y entre versiones diferentes del software. | **√** | Basta con leer los cambios de las nuevas versiones cada vez que google publica una actualización |
| RNF01 | Se podrá cambiar el color y la representación de los datos. | **√** | - |
| RNF02 | Se mostrarán los datos de una manera sencilla que facilite la comprensión. | **√** | - |
| RNF03 | El framework utilizara el lenguaje de programación R | **√** | GoogleVis usa R como framework |
| RNF04 | El framework será open source | **√** | GoogleVis es open source |

## 5.4 Documentación de instalación

### Paso 0

Para comenzar, la base de GoogleVis, como bien sabemos, es R-Studio. Es un paquete de 80mb que se puede descargar desde aquí (<https://cran.r-project.org/bin/windows/base/>). La versión con la que vamos a trabajar es la 3.6.0

Una vez abierto el instalador, nos saldrá una pantalla como esta:



**Ilustración 1**

En ella, haremos click 6 veces en “Siguiente”, y se comenzará a instalar.

Una vez instalado el programa, haremos click en el nuevo elemento del menú inicio:

* R i386 3.6.0

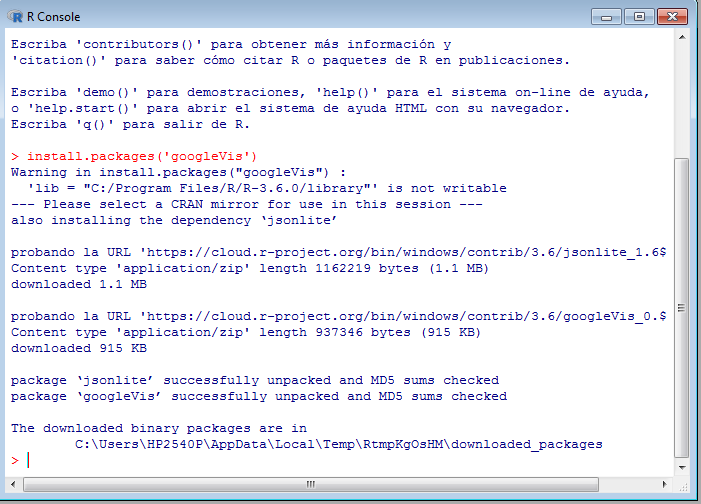
Bien, ya tenemos instalado y ejecutado nuestro compilador de R. Ahora es cuestión de implementar el paquete de GoogleVis. Esto lo haremos a través de 3 comandos.

### Paso 1

El primero de ellos será:

install.packages('googleVis')

Copiamos y pegamos el comando anterior. El programa nos pregunta si queremos crear una nueva librería, ya que la carpeta de instalación de R es de sólo lectura (por estar en “archivos de programa”). Le decimos que sí, y comenzará a descargar, como se muestra en la siguiente imagen:



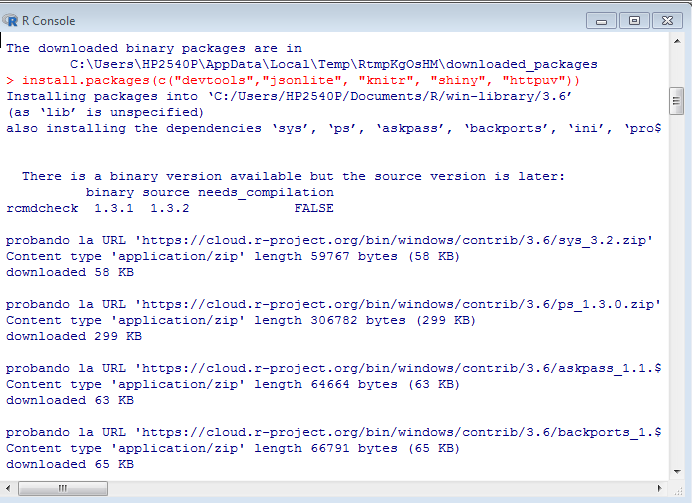
**Ilustración 2**

### Paso 2

Una vez finalizado esto, necesitamos instalar unos paquetes más, de los que depende GoogleVis. Para ellos, usaremos el comando:

install.packages(c("devtools","jsonlite", "knitr", "shiny", "httpuv"))

Copiamos y pegamos, y al apretar “Enter”, comenzará la descarga e instalación automáticas, como muestra la siguiente imagen:



**Ilustración 3**

### Paso 3

Antes de instalar los ejecutables de GoogleVis, debemos instalar el paquete Rtools, el cual tenemos que descargar e instalar desde el siguiente link:

<https://cran.r-project.org/bin/windows/Rtools/Rtools35.exe>

Importante: Tenemos que cerrar R antes de instalar el paquete

Una vez abierto, el instalador, haremos click de nuevo 6 veces en siguiente (tal y como se muestra en la Ilustración 4.

### Paso 4

Volvemos a abrir R para introducir el último comando, con el que queda instalado GoogleVis:

library(devtools)

install\_github("mages/googleVis)

### Paso 5 (opcional)

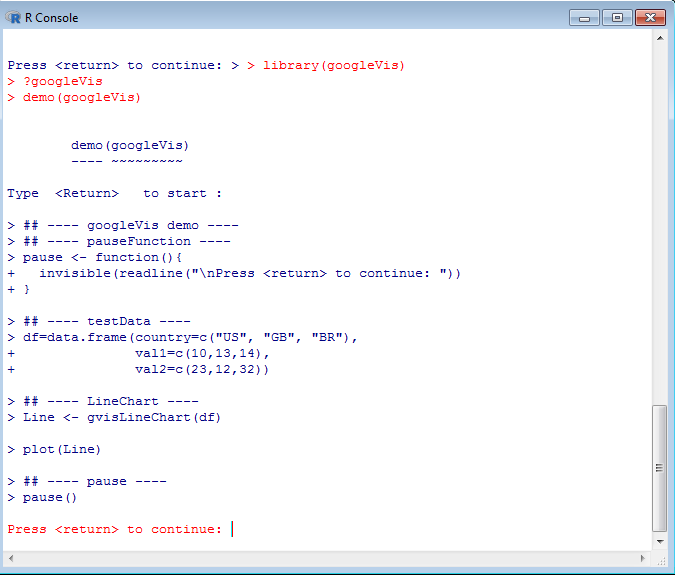
Para garantizar que GoogleVis está bien instalado, podemos poner los siguientes 3 comandos:

library(googleVis)

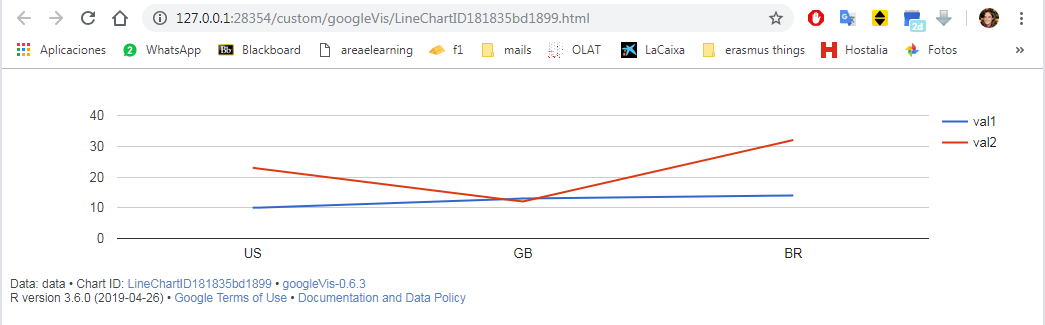
?googleVis

demo(googleVis)

Se abrirá tanto la página de introducción de GoogleVis, como una peuqeña g´rafica demo que nos confirma que googleVis está correctamente instalado:



**Ilustración 5**



**Ilustración 6**

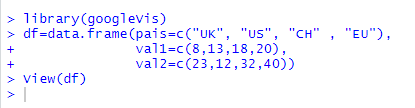
## 5.5 Manual de usuario

En el manual de usuario, nos vamos a ayudar de las pruebas del punto 5.3, para explicar cómo usar el software de manera sencilla e intutiva.

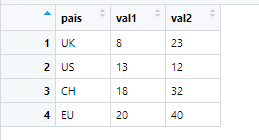
A continuación, pasamos a explicar todas las características del software:

### Creación de una tabla de datos

Lo primero que haremos será crear una variable que va a contener los datos que utilizaremos para su representación gráfica, introduciremos los datos de tal manera que se guarden en forma de tabla.



Aquí podemos observar la estructura de datos que contiene la variable creada (df).



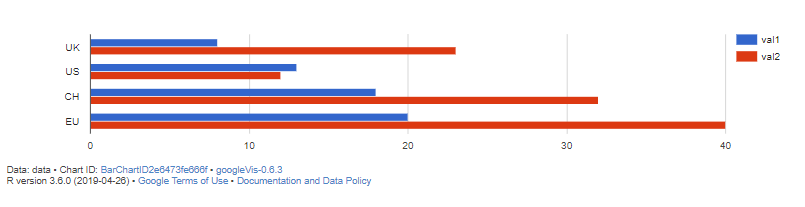
Una vez creada la tabla, podemos sacar gráficas de varias maneras. En este manual, se incluyen dos ejemplos:

### Gráficas. Ejemplo 1

Para realizar un gráfico de barras hará falta llamar la función correspondiente pasándole como argumento la tabla creada (gvisBarChart(df)). A continuación, lo único que quedaría por hacer es representar gráficamente (plot(Bar)).



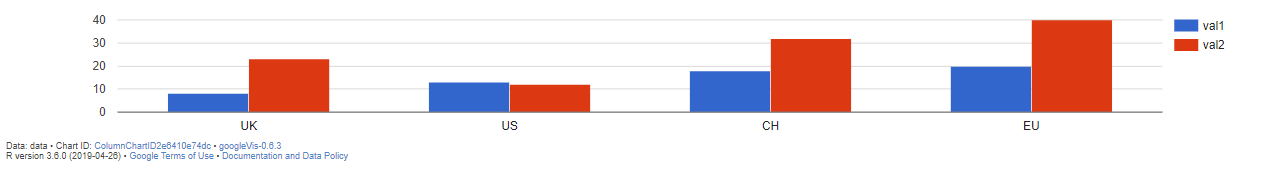
Obtenemos como resultado la siguiente gráfica:



### Gráficas. Ejemplo 2

Para representar nuestra tabla utilizando un gráfico de columnas se sigue un procedimiento similar, lo único que varía es la función a la que hacemos la llamada (gvisColumnChart(df)).





### Funcionalidades

En este apartado trataremos todas las funcionalidades del *software.* No se ha copiado la tabla del punto 2 porque algunas funcionalidades no hace falta explicarlas, ya que vienen dadas de manera implícita (como “representación de datos”), por lo que se adjuntan las funcionalidades que requieren alguna acción por parte del usuario.

### Zoom

Para hacer zoom, simplemente tendremos que apretar la tecla “Ctrl”, mientras movemos la rueda del ratón hacia delante o hacia atrás para ajustar el zoom

### Información adicional

Bastará con pasar el mouse por encima de las barras de los gráficos para obtener información adicional sobre éstos.

### Seleccionar información por conjuntos

Simplemente bastará con hacer “click” en los colores de la parte derecha de la pantalla para agrupar los datos por conjuntos

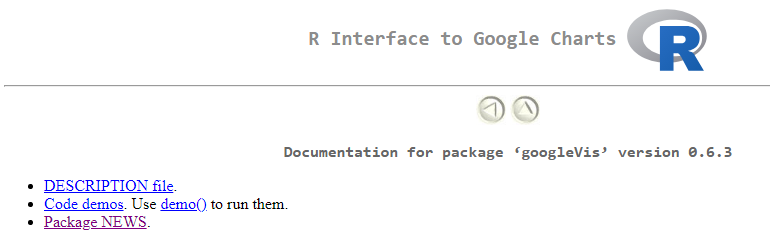
### Consultar cambios en las versiones de software

Para consultar las correciones y mejoras del software, seguiremos unos sencillos pasos:

* Introduciremos el código “?googleVis” en la ventana de comando de R
* Viajaremos hasta el final de la página que se nos abre, y haremos click en “index”



* En la parte izquierda de la pantalla, haremos click en “Package NEWS”



Y en esta página podemos consultar todas las nuevas funcionalidades y el histórico de las mismas.

googleVis 0.6.3 [2017-11-09]

----------------------------

Bug fixes

o Corrected URL for gvisCalendar documentation by Google

o Allow for numeric colorvar input for gvisBubbleChart to create

a gradient color scale

googleVis 0.6.2 [2017-01-01]

----------------------------

Changes

o Updated links to omegahat from omegahat.org

to omegahat.net as requested by CRAN

# 6. Comparación de las dos implementaciones

## 6.1 Evaluación de los criterios en la implementación usando el framework Ggplot2

|  |  |
| --- | --- |
| **CRITERIO** | **EVALUACIÓN** |
| 1- Facilidad de uso | Fácil e intuitivo. |
| 2- Documentación | Es el framework mas popular y por lo tanto el que tiene mas documentacion. |
| 3- Tiempo invertido | 15 minutos de media por gráfico. |
| 4- Facilidad en la instalación | Fácil de instalar mediante cmd de windows |
| 5- Rendimiento | Rendimiento positivo, ninguna ralentización significativa. |
| 6- Optimización | No se aprecian diferencias entre diferentes ordenadores |

## 6.2 Evaluación de los criterios en la implementación usando la tecnología GoogleVis

|  |  |
| --- | --- |
| **CRITERIO** | **EVALUACIÓN** |
| 1- Facilidad de uso | Fácil teniendo una base de conocimiento sobre lenguajes de programación. |
| 2- Documentación | Mucha, tanto en la wiki oficial como proporcionada por la comunidad. |
| 3- Tiempo invertido | 10 minutos de media por cada gráfico. |
| 4- Facilidad en la instalación | Muy fácil de instalar. |
| 5- Rendimiento | Rendimiento positivo, sin ralentizarse en ningún momento. |
| 6- Optimización | No se aprecian diferencias entre diferentes ordenadores. |

# 7. Comparación de la implementación de las tecnologías

|  |  |  |  |
| --- | --- | --- | --- |
| **CRITERIO** | **GOOGLE VIS** | **GGPLOT2** | **COMENTARIOS** |
| Facilidad de uso | 8 | 7 | Ambos frameworks son fáciles de usar si posees una base previa de programación |
| Documentación | Mucha | Mucha | Al ser dos de los frameworks más populares, es muy fácil encontrar documentación de calidad. |
| Tiempo invertido | 10 min/grafico | 15 min/grafico | Tiempo medio bajo en ambos. |
| Facilidad instalación | Fácil de instalar | Fácil de instalar | Se instalan desde cmd de una forma fácil especificada en sus páginas oficiales. |
| Rendimiento | 10 | 10 | Rendimiento sin ralentizaciones. |
| Optimización | 9 | 9 | Hemos probado ambos frameworks en todos nuestros ordenadores y no se aprecian diferencias de rendimiento. |

# 8. Conclusiones

Hemos escogido los dos framewoks más populares, por lo tanto ambos tienen niveles de calidad similares, tanto googleVis como Ggplot2 usan R como lenguaje de programación y por ello tienen funcionalidades muy similares.

En cuanto a la implementación se nota que son dos herramientas muy potentes, las cuales tienen un potencial muy elevado. En esta práctica hemos alcanzado un nivel medio, pero esto nos sirve de inicio para la elaboración de gráficos más complejos.

Podemos concluir que ambos son frameworks de visualización de datos de alta calidad, optimizados y llenos de funcionalidades, por lo tanto la elección entre uno u otro es cuestión de preferencias personales