

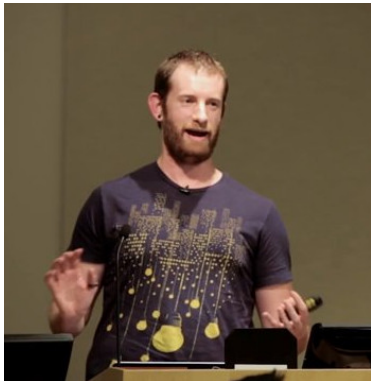
Introducción a “dplyr”

Mathieu Kessler

Departamento de Matemática Aplicada y Estadística
Universidad Politécnica de Cartagena

Cartagena

El paquete “dplyr”



Hadley Wickham

- Chief scientist at Rstudio.
- Autor de paquetes muy relevantes: “ggplot2”, “dplyr”, “tidyr”, etc..
- Autor de varios libros.

Preliminares

- Si no lo tenemos ya, creamos un proyecto `manipular_data_frames`. Si lo tenemos creado de la sesión anterior, lo abrimos... (Nota: nos aseguramos de que la codificación de texto para el proyecto es UTF-8, menú *Tools* → *Project Options* → *Code Editing*)
- Usamos la opción “New” del menú “File”, para crear un nuevo script R que llamamos `manipulaciones_de_data_frames.R`

Instalamos los paquetes pertinentes

Instalaremos los paquetes:

- `"tidyr"` : un paquete para preparar los datos en un formato adecuado para el análisis
- `"dplyr"` : un paquete para manipular datos de manera muy eficiente
- `"ggplot2"` : un paquete para realizar gráficas
- `"lubridate"` : un paquete para trabajar con fechas.

Todos estos paquetes han sido desarrollados por Hadley Wickham.

Instalamos los paquetes pertinentes

En el panel inferior izquierdo, usamos la pestaña “Packages”:

```
##> install.packages("dplyr")
```

le huecos que resultan de exploracion-huecos.org en

```
[LSE}]
```

R Markdown ↕

List▼

Environment is empty

```
** help
*** installing help indices
** building package indices
** testing if installed package can be loaded
* DONE (devtools)
```

```
The downloaded source packages are in
'/tmp/RtmpNQeZ3A/downloaded_packages'
R >
```

Files Plots Packages Help Viewer

Install Update Packrat

Name	Description
------	-------------

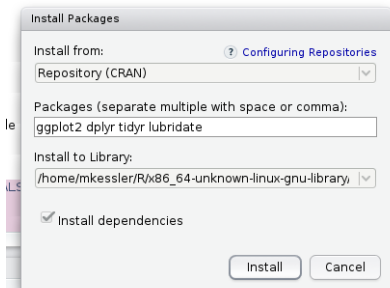
User Library

<input type="checkbox"/>	abind	Combine multi-dimensional arrays
<input type="checkbox"/>	ascii	Export R objects to several markup languages
<input type="checkbox"/>	assertthat	Easy pre and post assertions.
<input type="checkbox"/>	BH	Boost C++ Header Files
<input type="checkbox"/>	bitops	Bitwise Operations
<input type="checkbox"/>	brew	Templating Framework for Report Generation
<input type="checkbox"/>	Cairo	R graphics device using cairo graphics library for creating high-quality bitmap (PNG, JPEG, TIFF), vector (PDF, SVG, PostScript) and display (X11 and Win32) output.
<input type="checkbox"/>	car	Companion to Applied Regression

Instalamos los paquetes pertinentes

En el panel inferior izquierdo, usamos la pestaña “Packages”:

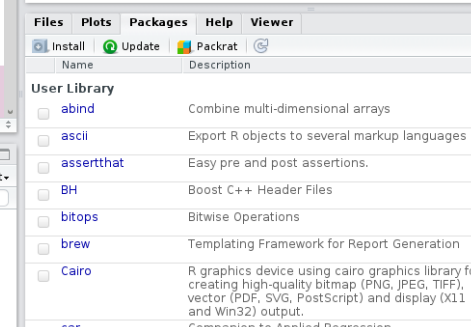
podemos hacer lo siguiente:



Environment is empty

```
** help
*** installing help indices
** building package indices
** testing if installed package can be loaded
* DONE (devtools)

The downloaded source packages are in
'/tmp/RtmpNQeZ3A/downloaded_packages'
R > |
```



Una vez instalados los paquetes...

Para poder usar las funcionalidades de un paquete, no es suficiente con instalarlo, es necesario cargarlo (con `library`) al principio de nuestra sesión...

```
library("dplyr")
```

Para hacer:

Introducid un primer bloque en vuestro script R que cargue el paquete “dplyr”.

El paquete “dplyr”

Los verbos de dplyr

dplyr introduce básicamente seis verbos que se aplican a un conjunto de datos:

- 1 **filter** : selecciona filas basadas en un criterio
- 2 **select** : selecciona columnas.
- 3 **arrange** : ordena las filas según los valores de una o varias columnas.
- 4 **mutate** : añade o transforma columnas al conjunto de datos.
- 5 **summarise** : calcula resúmenes de columnas según grupos de filas.
- 6 **rename** : cambia nombres de columnas.

El paquete dplyr

Una manera de facilitar la comprensión de las secuencias de operaciones:

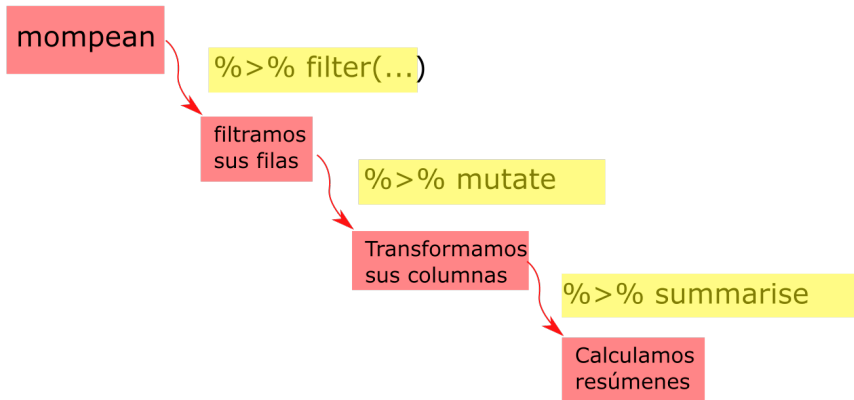
dplyr permite la sintaxis en cadena usando el operador %>%:

```
mompean %>% head
```

Es equivalente a

```
head(mompean)
```

Un típico esquema de trabajo con dplyr



Ejemplos de uso de dplyr con el conjunto de datos mompean

Cargamos el fichero de texto Mompean.txt usando `read.table` con las opciones adecuadas (ver transparencias anteriores).

Empezamos por ver sus columnas, podemos para ello usar la instrucción `glimpse` de dplyr. **What does “glimpse” mean in english?**

```
glimpse(mompean)
```

```
## Observations: 42,433
## Variables: 10
## $ fecha      <chr> "2006-01-01 00:00:00", "2006-01-01 01:00:00", "
## $ NO2        <int> 15, 18, 15, 19, 21, 21, 21, 21, 19, 21, 19, 20
## $ SO2        <int> 57, 22, 32, 81, 35, 19, 10, 19, 8, 21, 45, 11,
## $ ruido      <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA
## $ O3         <int> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA
## $ anio       <int> 2006, 2006, 2006, 2006, 2006, 2006, 2006, 2006
## $ mes        <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1
## $ dia_mes    <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1
## $ dia_semana <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
## $ hora       <int> 0. 1. 2. 3. 4. 5. 6. 7. 8. 9. 10. 11. 12. 13. :
```

Ejemplos de uso de dplyr con el conjunto de datos mompean: filtrar las filas.

- Supongamos que queremos quedarnos las mediciones de enero:

```
datos_enero <- mompean %>% filter(mes == 1)
```

- Si queremos seleccionar los datos diurnos (desde las 7:00 hasta las 22:00)

```
diurnos <- mompean %>% filter(hora >= 7 & hora <= 22)
```

- Entre ellos, seleccionamos los instantes que incumplen la normativa municipal de ruido (menos de 65 dB)

```
diurnos_ilegales <- diurnos %>% filter(ruido > 65)
```

- Si queremos seleccionar los nocturnos:

```
nocturnos <- mompean %>% filter(hora < 7 | hora > 22)
```

Ejemplos de uso de dplyr con el conjunto de datos mompean: seleccionar columnas

Seleccionar columnas

Cuando tenemos un conjunto de datos con muchas columnas, es útil ser capaz de seleccionar algunas o descartar otras. Lo hacemos con el verbo `select`.

- Nos queremos quedar sólo con fechaHora, NO2, SO2 y O3:

```
mompean_simple1 <- mompean %>% select(fecha, NO2, SO2, O3)
```

- Queremos descartar ruido: podemos usar el signo negativo:

```
mompean_simple2 <- mompean %>% select(-ruido)
```

- Queremos quedarnos con fechaHora, y con los dióxidos: SO2, NO2(estas dos columnas contienen "O2" en su nombre)

```
mompean_simple3 <- mompean %>% select(fecha, contains("O2"))
```

Ejemplos de uso de dplyr con el conjunto de datos mompean: calcular y transformar columnas

Para transformar columnas o calcular nuevas

Usamos el verbo `mutate`

- Queremos añadir una columna `oxidos` que sea la suma de `S02`, `N02` y `O3`, y además `prop_O2`, `prop_N02`, `prop_O3` que sean las fracciones de `S02`, `N02` y `O3` respecto a los óxidos totales

```
mompean <- mompean %>%  
  mutate(oxidos = S02 + N02 + O3,  
         prop_S02 = S02 / oxidos,  
         prop_N02 = N02 / oxidos,  
         prop_O3 = O3 / oxidos)
```

Podemos usar variables definidas previamente en el mismo `mutate`

Ejemplos de uso de dplyr con el conjunto de datos mompean: calcular y transformar columnas

Para transformar columnas o calcular nuevas

Usamos el verbo `mutate`

- Queremos añadir una columna `oxidos` que sea la suma de `S02`, `N02` y `O3`, y además `prop_O2`, `prop_N02`, `prop_O3` que sean las fracciones de `S02`, `N02` y `O3` respecto a los óxidos totales

```
mompean <- mompean %>%  
  mutate(oxidos = S02 + N02 + O3,  
         prop_S02 = S02 / oxidos,  
         prop_N02 = N02 / oxidos,  
         prop_O3 = O3 / oxidos)
```

Podemos usar variables
definidas previamente
en el mismo mutate

Ejemplos de uso de dplyr con el conjunto de datos mompean: calcular y transformar columnas

- Queremos añadir una columna `prop_ruido` que contenga el porcentaje de Ruido respecto a la norma municipal (65 en horario de 7 a 22, y 55 en el resto de horas)

```
mompean <- mompean %>%  
mutate(prop_ruido = ifelse(hora >= 7 & hora <= 22 ,  
                           ruido / 65 * 100,  
                           ruido / 55 * 100))
```

Hemos usado la instrucción `ifelse`:

```
mompean <- mompean %>%  
mutate(prop_ruido = ifelse(hora >= 7 & hora <= 22 ,  
                           ruido / 65 * 100,  
                           ruido / 55 * 100))
```

Condición lógica: puede ser TRUE o FALSE:

Ejemplos de uso de dplyr con el conjunto de datos mompean: resumir según grupos

Resumir según grupos.

A menudo necesitamos calcular resúmenes agrupando los datos según los valores de una o varias variables. Para ello, usamos el verbo `summarise` que precedemos por `group_by`.

- Queremos obtener el valor promedio de `S02` según día de la semana:

```
mompean %>%  
  group_by(dia_semana) %>%  
  summarise(S02_medio = mean(S02))
```

El resultado es "NA" para todos los días! "NA" quiere decir "Non Available". En este caso, la media es NA, porque, para cada día, falta alguna medición en algún momento. Se soluciona si añadimos la opción `na.rm = TRUE`, es decir "na remove" a `mean`.

Ejemplos de uso de dplyr con el conjunto de datos mompean: resumir según grupos

Resumir según grupos.

A menudo necesitamos calcular resúmenes agrupando los datos según los valores de una o varias variables. Para ello, usamos el verbo `summarise` que precedemos por `group_by`.

- Queremos obtener el valor promedio de `S02` según día de la semana:

```
mompean %>%  
  group_by(dia_semana) %>%  
  summarise(S02_medio = mean(S02, na.rm = TRUE))
```

El resultado es "NA" para todos los días! "NA" quiere decir "Non Available". En este caso, la media es NA, porque, para cada día, falta alguna medición en algún momento. Se soluciona si añadimos la opción `na.rm = TRUE`, es decir "na remove" a `mean`.

Ejemplos de uso de dplyr con el conjunto de datos mompean: resumir según grupos

Podemos añadir más de una variables de agrupamiento: por ejemplo si queremos calcular la media por día de la semana pero distinguiendo entre los meses del año:

```
mompean %>%  
  group_by(mes,dia_semana) %>%  
  summarise(SO2_medio = mean(SO2, na.rm = TRUE))
```

En lugar de mean podemos usar cualquier función que produzca un único valor: por ejemplo max, min, var, sd, n.

```
mompean %>%  
  group_by(mes,dia_semana) %>%  
  summarise(SO2_medio = mean(SO2, na.rm = TRUE),  
    numfilas = n())
```

Ejemplos de uso de dplyr con el conjunto de datos mompean: resumir según grupos

- `group_by` se puede usar también antes de hacer un `mutate`: añade una columna cuyos valores están calculados según los grupos.
En este caso usamos las “windows functions”, que devuelven un vector. Veremos algún ejemplo en algún caso práctico.
- Es muy interesante tener la “**Cheat sheet**” (chuleta) sobre “Data wrangling” de Rstudio:

Ejemplos de uso de dplyr con el conjunto de datos mompean: ordenar usando arrange

Para ordenar según una o varias variables

Podemos usar arrange

- Si queremos ordenar el resumen de medias de SO2 por mes, por valores crecientes de SO2 promedio:

```
mompean %>%  
  group_by(mes) %>%  
  summarise(SO2_medio = mean(SO2, na.rm = TRUE)) %>%  
  arrange(SO2_medio)
```

- Si queremos que sea por valores decrecientes de SO2 promedio:

```
mompean %>%  
  group_by(mes) %>%  
  summarise(SO2_medio = mean(SO2, na.rm = TRUE)) %>%  
  arrange(desc(SO2_medio))
```

Ejemplos de uso de dplyr con el conjunto de datos mompean: ordenar usando arrange

Para ordenar según una o varias variables

Podemos usar arrange

- Podemos ordenar por más de una columna. Por ejemplo, si queremos ordenar el conjunto Mompean por hora creciente, y después por mes creciente:

```
mompean %>%  
  arrange(hora, mes)
```

- O incluso por una columna creciente y otra decreciente:

```
mompean %>%  
  arrange(hora, desc(mes))
```