TALLER TIDYVERSE Y TIDYR

Tidyverse es un conjunto de librerías o paquetes de R, diseñados especialmente para la ciencia de datos, permitiendo realizar procesos de visualización y modelación de datos, entre otros procesos. Las librerías en mención comparten estructuras comunes, lo cual los hace más fácil de entender, aunque se aleja considerablemente del esquema tradicional de códigos en R. Con el siguiente comando se pueden visualizar alrededor de 30 paquetes disponibles *tidyverse_packages()*.

1. Paquetes básicos de Tidyverse:

ggplot2, dplyr, tidyr, readr.

2. Paquetes intermedios de Tidyverse:

purrr, tibble, string, forcast

1.1 readr y readxl

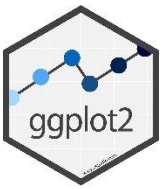
Readr es usado para leer (*read_csv*) y/o guardar (*write_csv*) archivos de texto plano como *csv*. readxl, es usado para leer archivos de Excel como *xlsx* (*read_excel*).

1.2 tidyr

Paquete para mejor la eficiencia o el orden de los datos *tidy*. Toma bases de datos *no tidy* y las convierte en *tidy*, paando tablas anchas a largas o largas a anchas (*gather* o *spread*, respectivamente). Mejora la eficiencia a la hora de organizar columnas (variables) y filas (observaciones y/o factores).

1.3 ggplot2

Paquete orientado a la gramática de figuras en una, dos o tres dimensiones. Por ser el paquete más antiguo, posee la mayor cantidad de variantes gramaticales.



1.4 dplyr

Paquete usado para manipular variables presentes en bases de datos, en procesos como selecciones, seleccionar algunas, estadísticos básicos, entre otros. Utiliza comandos como para manipular variables, como la selección (select), generar nuevas de algunas existentes (mutate), filtrar (filter), resumir (sumarize) y reordenar (arrange).

Función	Descripción
mutate()	Insertar variables de algunas existentes
select()	Selección de variables
filter()	Filtrar variables
summarise()	Resumir o reducir variables
arrange()	Organizar por magnitudes
group_by()	Agrupar
as_tibble()	Para convertir tablas a formato tidyr
rename()	Renombrar columnas

1.5 magrittr

Paquete para generar los operadores de **tuberías** o **pipes** % > %, que permiten realizar operaciones sin necesidad de realizar asignaciones previas. Permiten encadenar a varias operaciones en simultánea.

1.6 tibble

Permite cargar o construir bases de datos o data.frame, para luego ser guardadas en diferentes formatos.

1.7 stringr

Permite realizar ajustes a los nombres de las variables y demás datos categóricos en las bases de datos, utilizando al *str_* al inicio de cada comando.

Función	Descripción
str_replace()	Reemplazar patrones
str_c()	Combinar caracteres
str_detect()	Detectar patrones
str_extract()	Extraer patrones
str_sub()	Extraer por posición
str_length()	Longitud de la cadena

Ejemplo 1. Bases de datos con paquete "tidyr"

Objetivo: Utilizar las opciones del paquete tidyr, para la manipulación de bases de datos previamente generadas. A continuación, se presentan datos hipotéticos de tres grupos biológicos obtenidos de tres campañas de muestreo realizadas por un estudiante de ciencias biológicas.

```
# Librerías requeridas
library(tidyverse)

# base de datos de censo
datos <- data.frame(Meses = c("Enero", "Junio", "Octubre"),
                    periodos = c("Sequía", "Lluvias1", "Lluvias2"),
                    Taxón1 = c(2, 1, 3),
                    Taxón2 = c(20, 25, 30),
                    Taxón3 = c(4, 4, 4))
datos
```

```
##     Meses periodos Taxón1 Taxón2 Taxón3
## 1   Enero   Sequía      2     20      4
## 2   Junio Lluvias1      1     25      4
## 3 Octubre Lluvias2      3     30      4
```

```
# Datos en formato alargado (gather)
datos.l <- datos %>% gather(key = Columnas, value = Valores)
datos.l
```

```
##    Columnas  Valores
## 1     Meses    Enero
## 2     Meses    Junio
## 3     Meses  Octubre
## 4  periodos   Sequía
## 5  periodos Lluvias1
## 6  periodos Lluvias2
## 7    Taxón1        2
## 8    Taxón1        1
## 9    Taxón1        3
## 10   Taxón2       20
## 11   Taxón2       25
## 12   Taxón2       30
## 13   Taxón3        4
## 14   Taxón3        4
## 15   Taxón3        4
```

```
# Datos en formato alargado (gather) sin meses
datos.l <- datos %>% gather(key = Columnas, value = Valores, -periodos)
datos.l
```

```
##    periodos Columnas Valores
## 1    Sequía    Meses   Enero
## 2  Lluvias1    Meses   Junio
## 3  Lluvias2    Meses Octubre
## 4    Sequía   Taxón1       2
## 5  Lluvias1   Taxón1       1
## 6  Lluvias2   Taxón1       3
## 7    Sequía   Taxón2      20
## 8  Lluvias1   Taxón2      25
## 9  Lluvias2   Taxón2      30
```

```
## 10   Sequía   Taxón3       4
## 11 Lluvias1   Taxón3       4
## 12 Lluvias2   Taxón3       4

# Datos en formato ancho (spread)
datos.a <- datos.l %>% spread(key = Columnas, value = Valores)
datos.a

##   periodos   Meses Taxón1 Taxón2 Taxón3
## 1 Lluvias1   Junio      1     25      4
## 2 Lluvias2 Octubre      3     30      4
## 3   Sequía   Enero      2     20      4
```

Ejemplo 2. Bases de datos con dos factores

Objetivo: Utilizar otras opciones del paquete *tidyverse*, para la manipulación de bases de datos previamente generadas. A continuación, se presentan datos hipotéticos de cuatro estudiantes, a los que se les tomaron dos mediciones en cuatro ocasiones. Son dos estudiantes de cada sexo.

```
# base de datos de censo
library(kableExtra)

datos <- data.frame    (n = 1:16,       # Consecutivo
         Estudiante = c("a","a","a","a", "b","b","b","b",
                        "c","c","c","c", "d","d","d","d"),       # Individuos
         Sexo =       c("f","f","f","f","f","f","f","f",
                        "m","m","m","m","m","m","m","m"),        # Sexo
         Variable_1 = c(1.2, 3.4, 4.5, 5.6, 1.2, 3.4, 4.5, 5.6,
                        0.8, 2.4, 1.8, 1.5, 1.6, 2.1, 1.2, 0.8),   # Valores
         Variable_2 =  c(2.4, 6.8, 9.0, 11.2, 2.4, 6.8, 9.0, 11.2,
                         1.6, 4.8, 3.6, 3.0, 3.2, 4.2, 2.4, 1.6))  # Valores
head(datos)
```

```
##   n Estudiante Sexo Variable_1 Variable_2
## 1 1          a    f        1.2        2.4
## 2 2          a    f        3.4        6.8
## 3 3          a    f        4.5        9.0
## 4 4          a    f        5.6       11.2
## 5 5          b    f        1.2        2.4
## 6 6          b    f        3.4        6.8
```

```
# Filtrar por sexo
datos.f <- datos [datos$Sexo == "f",]     # Formato tradicional
datos.f
```

```
##   n Estudiante Sexo Variable_1 Variable_2
## 1 1          a    f        1.2        2.4
## 2 2          a    f        3.4        6.8
## 3 3          a    f        4.5        9.0
## 4 4          a    f        5.6       11.2
## 5 5          b    f        1.2        2.4
## 6 6          b    f        3.4        6.8
## 7 7          b    f        4.5        9.0
## 8 8          b    f        5.6       11.2
```

```
datos.f <- datos %>% filter(Sexo == "f")  # Formato tidy
datos.f
```

```
##   n Estudiante Sexo Variable_1 Variable_2
## 1 1          a    f        1.2        2.4
## 2 2          a    f        3.4        6.8
## 3 3          a    f        4.5        9.0
## 4 4          a    f        5.6       11.2
## 5 5          b    f        1.2        2.4
## 6 6          b    f        3.4        6.8
## 7 7          b    f        4.5        9.0
## 8 8          b    f        5.6       11.2
```

```
# Filtrar por tipo de estudiante y por Sexo
datos.a <- datos.f [datos.f$Estudiante == "a",]     # Formato tradicional
datos.a
```

```
##   n Estudiante Sexo Variable_1 Variable_2
## 1 1          a    f        1.2        2.4
## 2 2          a    f        3.4        6.8
## 3 3          a    f        4.5        9.0
## 4 4          a    f        5.6       11.2
```

```
datos.a <- datos %>% filter(Sexo == "f", Estudiante == "a")  # Formato tidy
datos.a
```

```
##   n Estudiante Sexo Variable_1 Variable_2
## 1 1          a    f        1.2        2.4
## 2 2          a    f        3.4        6.8
## 3 3          a    f        4.5        9.0
## 4 4          a    f        5.6       11.2
```

```
# Filtrar en órden descendente (arrange y desc)
datos.des <- datos %>% arrange(desc(Variable_1))  # Formato tidy
datos.des
```

```
##     n Estudiante Sexo Variable_1 Variable_2
## 1   4          a    f        5.6       11.2
## 2   8          b    f        5.6       11.2
## 3   3          a    f        4.5        9.0
## 4   7          b    f        4.5        9.0
## 5   2          a    f        3.4        6.8
## 6   6          b    f        3.4        6.8
## 7  10          c    m        2.4        4.8
## 8  14          d    m        2.1        4.2
## 9  11          c    m        1.8        3.6
## 10 13          d    m        1.6        3.2
## 11 12          c    m        1.5        3.0
## 12  1          a    f        1.2        2.4
## 13  5          b    f        1.2        2.4
## 14 15          d    m        1.2        2.4
## 15  9          c    m        0.8        1.6
## 16 16          d    m        0.8        1.6
```

```
# Filtrar en órden ascendente (arrange)
datos.asc <- datos %>% arrange(Variable_2)  # Formato tidy
datos.asc
```

```
##     n Estudiante Sexo Variable_1 Variable_2
## 1   9          c    m        0.8        1.6
## 2  16          d    m        0.8        1.6
## 3   1          a    f        1.2        2.4
## 4   5          b    f        1.2        2.4
## 5  15          d    m        1.2        2.4
## 6  12          c    m        1.5        3.0
## 7  13          d    m        1.6        3.2
## 8  11          c    m        1.8        3.6
## 9  14          d    m        2.1        4.2
## 10 10          c    m        2.4        4.8
## 11  2          a    f        3.4        6.8
## 12  6          b    f        3.4        6.8
## 13  3          a    f        4.5        9.0
## 14  7          b    f        4.5        9.0
```

```
## 15  4          a    f        5.6       11.2
## 16  8          b    f        5.6       11.2
```

```
# Filtrar en órden ascendente solo a mujeres (arrange)
datos.asc <- datos %>%
             filter(Sexo == "f") %>%
             arrange(Variable_2)  # Formato tidy
datos.asc
```

```
##   n Estudiante Sexo Variable_1 Variable_2
## 1 1          a    f        1.2        2.4
## 2 5          b    f        1.2        2.4
## 3 2          a    f        3.4        6.8
## 4 6          b    f        3.4        6.8
## 5 3          a    f        4.5        9.0
## 6 7          b    f        4.5        9.0
## 7 4          a    f        5.6       11.2
## 8 8          b    f        5.6       11.2
```

```
# Agregar variables derivadas (mutate)
datos.3 <- datos %>%
           mutate(Variable_3 = Variable_1 + Variable_2)
datos.3
```

```
##     n Estudiante Sexo Variable_1 Variable_2 Variable_3
## 1   1          a    f        1.2        2.4        3.6
## 2   2          a    f        3.4        6.8       10.2
## 3   3          a    f        4.5        9.0       13.5
## 4   4          a    f        5.6       11.2       16.8
## 5   5          b    f        1.2        2.4        3.6
## 6   6          b    f        3.4        6.8       10.2
## 7   7          b    f        4.5        9.0       13.5
## 8   8          b    f        5.6       11.2       16.8
## 9   9          c    m        0.8        1.6        2.4
## 10 10          c    m        2.4        4.8        7.2
## 11 11          c    m        1.8        3.6        5.4
## 12 12          c    m        1.5        3.0        4.5
## 13 13          d    m        1.6        3.2        4.8
## 14 14          d    m        2.1        4.2        6.3
## 15 15          d    m        1.2        2.4        3.6
## 16 16          d    m        0.8        1.6        2.4
```

```
# Combinar filtrado, adición, orden (filter + mutate + arrange)
datos.3 <- datos %>%
           filter (Sexo == "f") %>%
           mutate (Variable_3 = Variable_2 * 12) %>%
           arrange(desc(Variable_3))
datos.3
```

```
##   n Estudiante Sexo Variable_1 Variable_2 Variable_3
## 1 4          a    f        5.6       11.2      134.4
## 2 8          b    f        5.6       11.2      134.4
## 3 3          a    f        4.5        9.0      108.0
## 4 7          b    f        4.5        9.0      108.0
## 5 2          a    f        3.4        6.8       81.6
## 6 6          b    f        3.4        6.8       81.6
```

```
## 7 1          a    f        1.2        2.4       28.8
## 8 5          b    f        1.2        2.4       28.8

# Promedios para los diferentes factores (group_by, sumarise)
datos.r <- datos %>%
           group_by (Estudiante, Sexo) %>%         # group_by = Factores agrupad
os
           summarise(Var_2.m = mean(Variable_2))   # Var_2.m = promedios de Vari
able_2
datos.r

## # A tibble: 4 × 3
## # Groups:   Estudiante [4]
##   Estudiante Sexo  Var_2.m
##   <chr>      <chr>   <dbl>
## 1 a          f        7.35
## 2 b          f        7.35
## 3 c          m        3.25
## 4 d          m        2.85

# Promedios para los diferentes factores (group_by, sumarise)
datos.r <- datos %>%
           group_by (Estudiante, Sexo) %>%           # Factores agrupados
           summarise(Var_2.m   = mean(Variable_2),   # Promedios
                     Var_2.de  = sd(Variable_2),     # Desviaciones
                     Var_2.var = var(Variable_2),    # Varianzas
                     Var_2.n   = n(),                # Tamaños de las muestras
                     Var_2.ee  = sd(Variable_2)/sqrt(n()),        # Errores estándar
                     Var_2.máx = max(Variable_2, na.rm = TRUE),# Tamaños de las muestras
                     Var_2.mín = min(Variable_2, na.rm = TRUE))   # Errores estándar

datos.r

## # A tibble: 4 × 9
## # Groups:   Estudiante [4]
##   Estudiante Sexo  Var_2.m Var_2.de Var_2.var Var_2.n Var_2.ee Var_2.máx Var_
2…¹
##   <chr>      <chr>   <dbl>    <dbl>     <dbl>   <int>    <dbl>     <dbl>   <dbl>
## 1 a          f        7.35     3.76     14.1        4    1.88       11.2     2.4
## 2 b          f        7.35     3.76     14.1        4    1.88       11.2     2.4
## 3 c          m        3.25     1.33      1.77       4    0.665       4.8     1.6
## 4 d          m        2.85     1.11      1.24       4    0.556       4.2     1.6
## # … with abbreviated variable name ¹Var_2.mín

# Promedios para los diferentes factores (group_by, sumarise)
datos.r <- datos %>%
  group_by (Estudiante, Sexo) %>%           # Factores agrupados
  summarise(Var_2.m   = mean(Variable_2),   # Promedios
            Var_2.de  = sd(Variable_2),     # Desviaciones
            Var_2.var = var(Variable_2),    # Varianzas
            Var_2.n   = n(),                # Tamaños de las muestras
            Var_2.ee  = sd(Variable_2)/sqrt(n()))  # Errores estándar
datos.r

## # A tibble: 4 × 7
## # Groups:   Estudiante [4]
##   Estudiante Sexo  Var_2.m Var_2.de Var_2.var Var_2.n Var_2.ee
##   <chr>      <chr>   <dbl>    <dbl>     <dbl>   <int>    <dbl>
## 1 a          f        7.35     3.76     14.1        4    1.88
```

```
## 2 b          f        7.35     3.76     14.1        4    1.88
## 3 c          m        3.25     1.33      1.77       4    0.665
## 4 d          m        2.85     1.11      1.24       4    0.556
```

Graficar los datos.

```
# Figura de cajas y bigotes
x11()
ggplot(datos, aes(x=Sexo, y=Variable_2)) +
  geom_boxplot()
```

```
# Ajuste de colores
ggplot(datos, aes(x=Sexo, y=Variable_2)) +
  geom_boxplot(col="Blue", fill="orange") +
  labs(x="Sexo", y= "Variable 2") +
  theme(axis.text= element_text(size=14))
```