

# CLASE 05 - Análisis exploratorio de datos

Curso Análisis de expresión diferencial de genes e  
investigación reproducible.

Dr. José Gallardo Matus | <https://genomics.pucv.cl/>

Pontificia Universidad Católica de Valparaíso

22 October 2022

# PLAN DE LA CLASE

## 1.- Introducción

- ▶ ¿Qué es un análisis exploratorio de datos?.
- ▶ Gráficas avanzadas con ggplot2.
- ▶ Manipular datos con tidyr: Tuberías.
- ▶ Manipular datos con dplyr: filtrar, seleccionar, eliminar muestras.

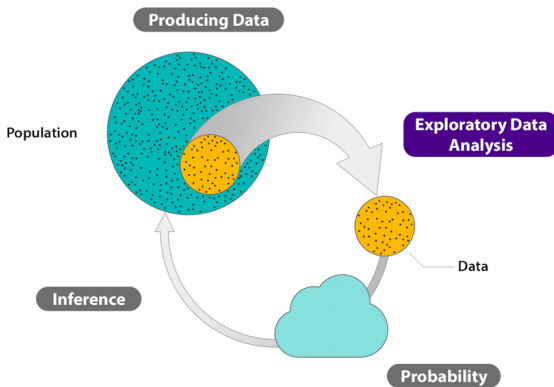
## 2.- Práctica con R y Rstudio cloud

- ▶ Realizar un análisis exploratorio de datos de ARN.
- ▶ Realizar gráficas avanzadas con ggplot2.
- ▶ Manipular de datos con tidyr y dplyr.

# ANÁLISIS EXPLORATORIO DE DATOS (EDA)

## ¿Qué es un análisis exploratorio de datos?

Procedimiento que permite visualizar y explorar las variables/datos de un estudio.



# ¿POR QUÉ ES NECESARIO HACER UN EDA?

## Principalmente para:

1. Investigar calidad de los datos brutos (ARN).
2. Eliminar muestras de baja calidad de (ARN).
3. Observar variación de los datos (expresión de genes).
4. Establecer un modelo básico de relación e interacción entre variables (expresión de genes).
5. Seleccionar una prueba estadística adecuada (expresión de genes).

# EDA ES UN PROCESO ITERATIVO

## ¿Cómo realizar un buen EDA?

1. Genera preguntas iniciales para explorar tus datos.
2. Resume, visualiza, transforma y modela tus datos.
3. Usa lo que aprendiste para generar nuevas preguntas.

## Preguntas clave, pero no las únicas

- ▶ ¿Existen errores, datos faltantes, muestras de baja calidad?
- ▶ ¿Qué tipo de variación existe en la/s variables de estudio?
- ▶ ¿Qué tipo de covariación o interacción existe entre las variables de estudio?
- ▶ ¿Cuál es el modelo más simple que explica la relación entre variables?

# GRÁFICAS CON GGLOT2

## ggplot2

Paquete de visualización de datos preferido para realizar análisis exploratorio de datos con R (Wickham en 2005).

### Ventajas

- Gran flexibilidad.
- Sistema para realizar gráficos completo y maduro.
- Una gran comunidad de desarrolladores.

### Características

- Los datos siempre deben ser un data.frame.
- Usa un sistema diferente para añadir elementos al gráfico.



# COMPARACIÓN GGLOT2 - GRAPHICS

Comparación de algunos comandos de gráficas entre los paquetes **graphics** y **ggplot2**

Función	graphics	ggplot2
Función genérica para graficar	plot()	ggplot()
Histogramas	hist()	geom_histogram()
Gráfica de cajas y bigotes	boxplot()	geom_boxplot()
Etiquetar ejes	xlab=" " , ylab=" "	labs(x=" ",y=" ")

# RECORDAR FORMATO CORRECTO DE DATOS

## **Tidy data (datos ordenados)**

- ▶ Cada columna es una variable.
- ▶ Cada fila es una observación.
- ▶ Cada celda es un simple dato o valor.

## **Messy data (desordenados)**

- ▶ Cualquier conjunto de datos que no cumple alguno de estos criterios.



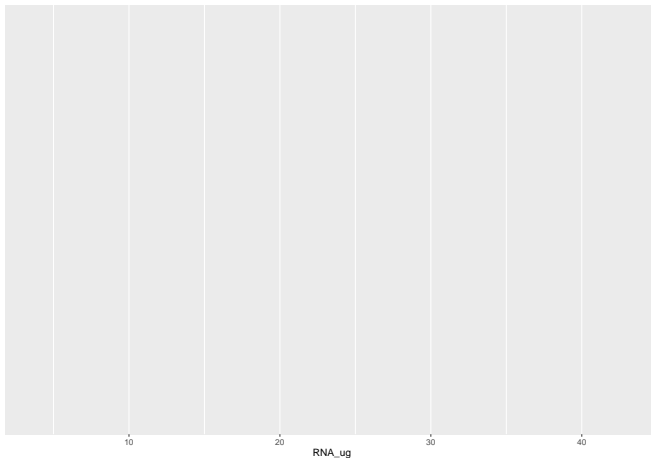
# EJEMPLO DATOS ORDENADOS

```
## # A tibble: 6 x 6
##   Type  Sample weight_mg RNA_ug `260_280` `260_230`
##   <chr> <chr>         <dbl>  <dbl>      <dbl>      <dbl>
## 1 Pig   WF1_D           39.3    5.2         2         1.1
## 2 Pig   WF1_S           24.7    4.4         2         1
## 3 Pig   WF2_D           30      6.6         2         1.5
## 4 Pig   WF2_S           21     13.6         2         1
## 5 Pig   WF3_D           53     14.5         2         1.5
## 6 Pig   WF3_S           50      7          2         1.6
```

# ¿CÓMO FUNCIONA GGPLOT2?

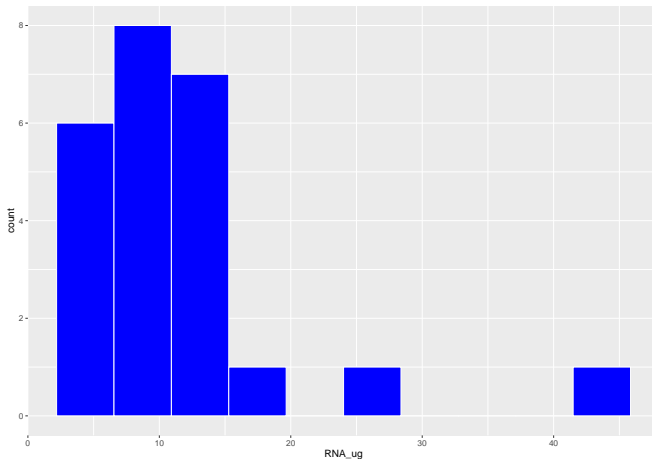
**ggplot2 funciona por capas**

```
ggplot(RNA_sample, aes(RNA_ug))
```



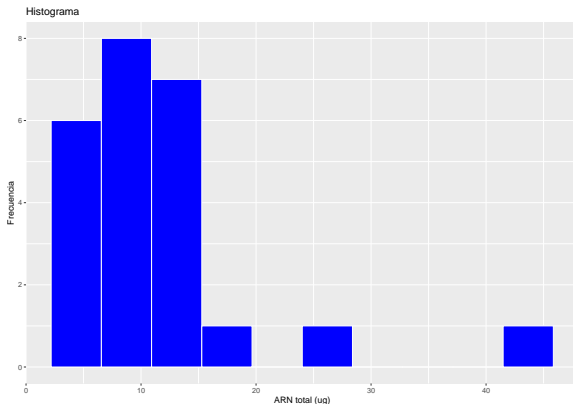
# HISTOGRAMAS CON GGPLOT2

```
ggplot(RNA_sample, aes(RNA_ug)) +  
  geom_histogram(color="white", fill="blue", bins = 10)
```



# CAMBIAR ETIQUETAS DE EJES

```
ggplot(RNA_sample, aes(RNA_ug))+  
  geom_histogram(color="white", fill="blue", bins = 10)+  
  labs(title="Histograma", x="ARN total (ug)",  
        y="Frecuencia")
```

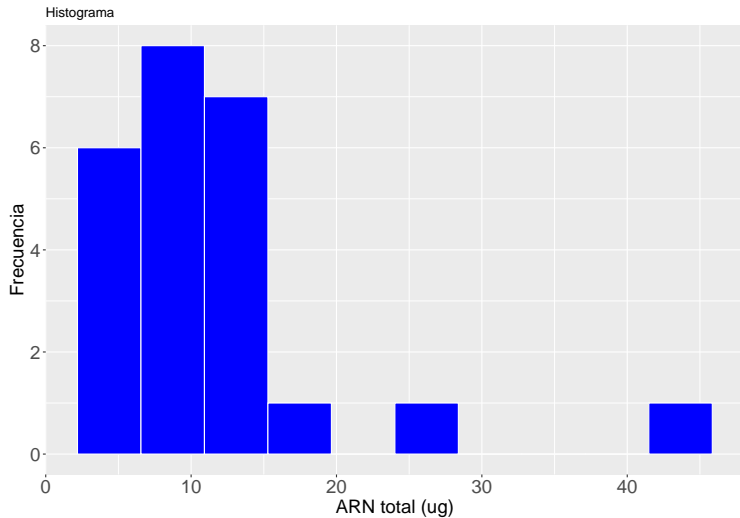


# CAMBIAR TAMAÑO DE ETIQUETAS

```
My_Theme = theme(  
  axis.title.x = element_text(size = 18),  
  axis.text.x = element_text(size = 18),  
  axis.title.y = element_text(size = 18),  
  axis.text.y = element_text(size = 18))  
  
plot_1 <- ggplot(RNA_sample, aes(RNA_ug))+  
  geom_histogram(color="white", fill="blue", bins = 10)+  
  labs(title="Histograma", x="ARN total (ug)",  
        y="Frecuencia")
```

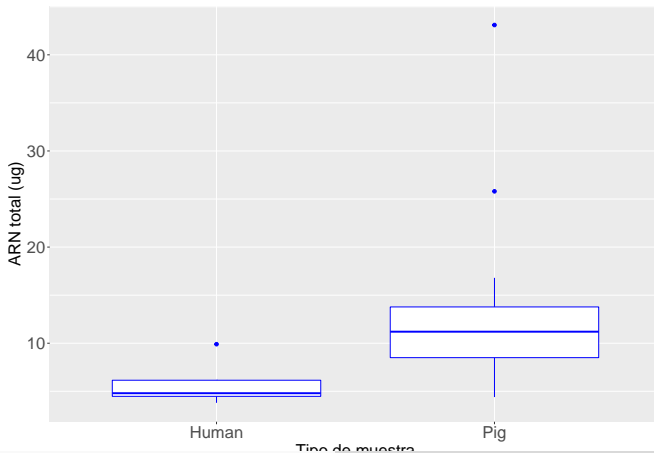
# HISTOGRAMA FINAL

```
plot_1 + My_Theme
```



# BOXPLOT CON GGPLOT2

```
ggplot(RNA_sample, aes(x=Type, y= RNA_ug))+  
  geom_boxplot(color="blue")+  
  labs( x="Tipo de muestra", y="ARN total (ug)") +  
  My_Theme
```



# MANIPULACIÓN DE DATOS

## Tareas comunes durante esta etapa:

- ▶ Filtrar datos por categorías.
- ▶ Remover muestras con baja calidad.
- ▶ Imputar datos faltantes.
- ▶ Agrupar datos por algún criterio.
- ▶ Seleccionar y calcular estadísticos.
- ▶ Generar variables derivadas a partir de variables existentes.
- ▶ Transformar variables.



# PAQUETES CLAVE

Importar

transformar

Visualizar



# EL OPERADOR PIPE: %>%.

En programación **pipe** es una técnica que permite pasar información de un proceso o programa a otro por etapas.

Evita pipe cuando: a) Deseas manipular varios objetos a la vez. b) Un paso intermedio genera un objeto que luego deseas analizar separadamente.

datos  funcion(...)

datos  funcion(1)  Funcion(2)

# PAQUETE DPLYR: FUNCIONES BÁSICAS

**select()**: Permite extraer o seleccionar variables/columnas específicas de un data.frame.

**filter()**: Para filtrar desde una tabla de datos un subconjunto de filas. Ej. solo un nivel de un factor, observaciones que cumplen algún criterio (ej.  $> 20$ ).

**mutate()**: Permite calcular/generar nuevas variables “derivadas”. Útil para calcular proporciones, tasas.

**group\_by()**: Permite agrupar filas con base a los niveles de alguna variable o factor.

**summarize()**: Permite obtener medidas resumen de las variables.

# SELECT()

- ▶ Selección de variables.

Datos %>% select(Type, RIN)

Type	Sample	RIN
Pig	WF11_M	7.8
Pig	WF12_M	3.1
Pig	WF13_M	9.3
Human	WF3_D	8.9
Human	WF7_D	2.6
Human	WF11_M	8.7



Type	RIN
Pig	7.8
Pig	3.1
Pig	9.3
Human	8.9
Human	2.6
Human	8.7

# FILTER()

- Filtrar por categorías o factores.

Datos %>% filter(Type="Pig")

Type	Sample	RIN
Pig	WF11_M	7.8
Pig	WF12_M	3.1
Pig	WF13_M	9.3
Human	WF3_D	8.9
Human	WF7_D	2.6
Human	WF11_M	8.7



Type	Sample	RIN
Pig	WF11_M	7.8
Pig	WF12_M	3.1
Pig	WF13_M	9.3

# FILTER()

- Filtrar por valor numérico.

Datos %>% filter(RIN>=7)

Type	Sample	RIN
Pig	WF11_M	7.8
Pig	WF12_M	3.1
Pig	WF13_M	9.3
Human	WF3_D	8.9
Human	WF7_D	2.6
Human	WF11_M	8.7




Type	Sample	RIN
Pig	WF11_M	7.8
Pig	WF13_M	9.3
Human	WF3_D	8.9
Human	WF11_M	8.7

# MUTATE()

- Calcula y agrega nuevas variables.

Datos %>% mutate(RNAng=(RNAug\*1000))

Type	Sample	RNAug		Type	Sample	RNAug	RNAng
Pig	WF11_M	5.5		Pig	WF11_M	5.5	5500
Pig	WF12_M	11.6		Pig	WF12_M	11.6	11600
Pig	WF13_M	10.8		Pig	WF13_M	10.8	10800
Human	WF3_D	3.8		Human	WF3_D	3.8	3800
Human	WF7_D	9.9		Human	WF7_D	9.9	9900
Human	WF11_M	4.9		Human	WF11_M	4.9	4900

# GROUP\_BY() + SUMMARIZE()

```
Datos %>% grup_by(Type) %>%  
  summarize(n=n(),  
            promedio=mean(RIN))
```

Type	Sample	RIN
Pig	WF10_M	9.1
Human	WF3_D	8.9
Pig	WF11_M	7.8
Human	WF11_M	8.7
Pig	WF13_M	9.3



Type	Sample	RIN
Pig	WF10_M	9.1
Pig	WF11_M	7.8
Pig	WF13_M	9.3
Human	WF3_D	8.9
Human	WF11_M	8.7



Type	n	promedio
Pig	3	8.73
Human	2	8.80



# RESUMEN DE LA CLASE

1. Realizamos análisis exploratorio de datos.
2. Realizamos gráficas con ggplot2.
3. Manipulamos datos con tidyr y dplyr.
4. Utilizamos tuberías o pipe %>%.