Formas funcionales (parte 2)

2022-10-12

```
knitr::opts_chunk$set(warning = FALSE, message = FALSE)
library(tidyverse)
library(tidymodels)
```

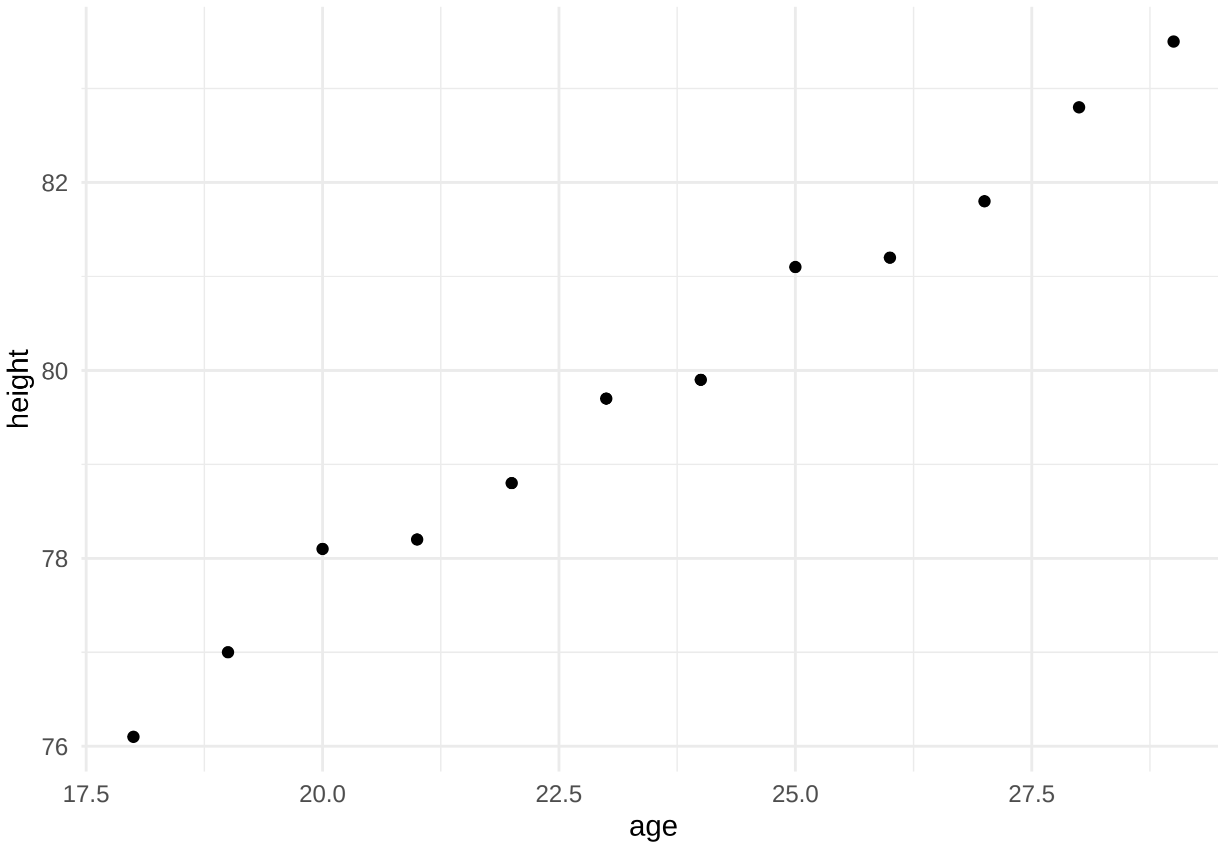
1.Lectura de datos

Para este ejercicio se emplearán los datos obtenidos del libro Data Analysis using regression and multi-level/hierarchical models de Andrew Gelman. En estos datos se busca generar una regresión lineal que trata de modelar la altura de un niño (en cm) en términos de su edad en meses:

```
library(tidyverse)
library(tidymodels)
library(readr)

datos <- readr::read_csv("https://raw.githubusercontent.com/savrgg/class_ITAM_metodos/main/notas_r/agea

# log - log
datos_loglog <-
  datos %>%
  mutate(age_log = log(age),
         height_log = log(height))

datos_loglog
```

```
## # A tibble: 12 x 4
##      age height age_log height_log
##    <dbl>  <dbl>   <dbl>      <dbl>
##  1    18   76.1    2.89       4.33
##  2    19   77      2.94       4.34
##  3    20   78.1    3.00       4.36
##  4    21   78.2    3.04       4.36
##  5    22   78.8    3.09       4.37
##  6    23   79.7    3.14       4.38
##  7    24   79.9    3.18       4.38
##  8    25   81.1    3.22       4.40
##  9    26   81.2    3.26       4.40
## 10    27   81.8    3.30       4.40
## 11    28   82.8    3.33       4.42
## 12    29   83.5    3.37       4.42
```

```
datos %>%
  ggplot(aes( x= age, y = height))+
  geom_point()+
  theme_minimal()
```


82
80
78
76
height
17.5
20.0
22.5
25.0
27.5
age

2.Modelo lineal tradicional:

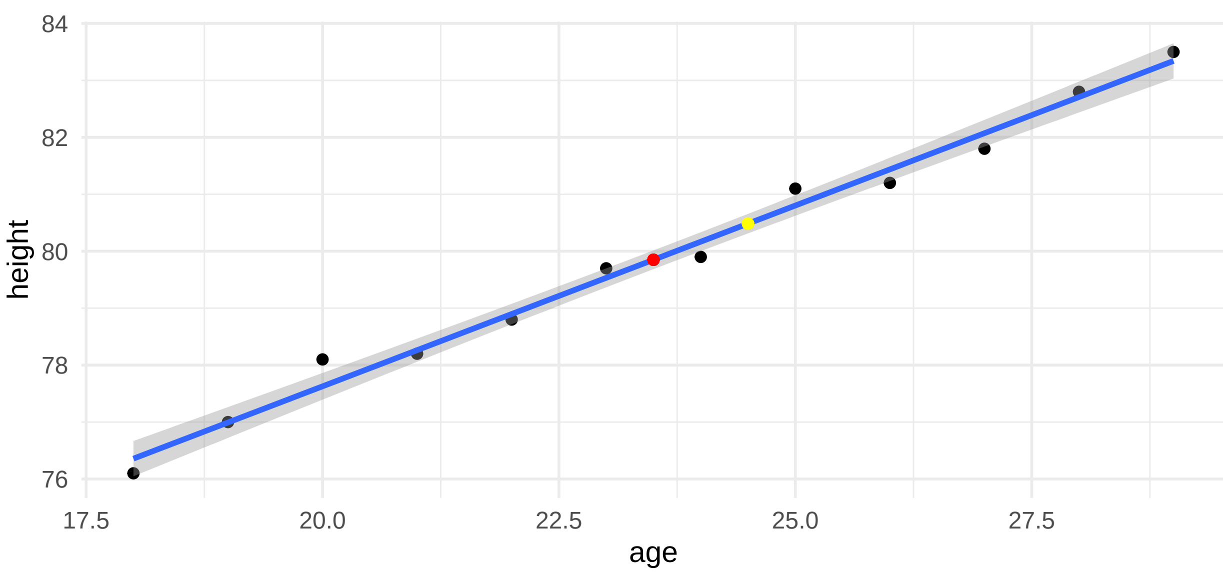
```
model_est <-
  linear_reg() %>%
  fit(height ~ age, data = datos_loglog)

tidy(model_est)
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)   64.9      0.508      128.   2.13e-17
## 2 age            0.635    0.0214      29.7 4.43e-11
```

```
glance(model_est)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC
##       <dbl>         <dbl> <dbl>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1     0.989         0.988 0.256      880. 4.43e-11     1  0.419  5.16  6.62
## # ... with 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

```
datos_loglog %>%
  ggplot(aes( x= age, y = height)) +
  geom_point()+
  geom_smooth(method = "lm")+
  geom_point(x = mean(datos_loglog$age), y = mean(datos_loglog$height), color = "red")+
  geom_point(x = mean(datos_loglog$age)+1, y = mean(datos_loglog$height)+0.634965, color = "yellow")+
  theme_minimal()
```



De esta regresión podemos ver que cuando se aumenta en 1 unidad la variable independiente (mes), entonces en promedio, la variable dependiente aumentará en 0.6449 unidades (cm).

3.Modelo Log-Log

Ahora, cuando aplicamos el modelo log-log, cambiará la interpretación. Por ejemplo:

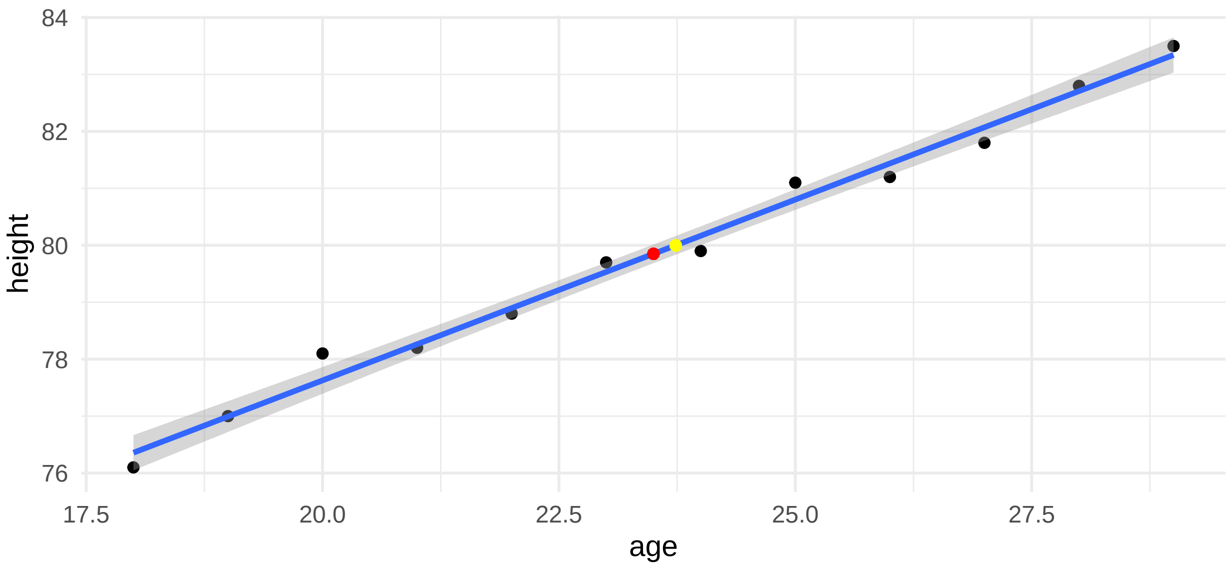
```
# log-log
model_est <-
  linear_reg() %>%
  fit(height_log ~ age_log, data = datos_loglog)

tidy(model_est)
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)    3.80    0.0214      177.  8.00e-19
## 2 age_log        0.184   0.00681      27.0 1.11e-10
```

```
glance(model_est)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared   sigma statistic  p.value    df logLik   AIC   BIC
##       <dbl>         <dbl>   <dbl>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1     0.986         0.985 0.00352      730. 1.11e-10     1   51.9 -97.7 -96.3
## # ... with 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

```
datos_loglog %>%
  ggplot(aes( x= age, y = height)) +
  geom_point()+
  geom_smooth(method = "lm")+
  geom_point(x = mean(datos_loglog$age), y = mean(datos_loglog$height), color = "red")+
  geom_point(x = mean(datos_loglog$age)*1.01, y = mean(datos_loglog$height)*1.001839486, color = "yellow
  theme_minimal()
```



Interpretación: Por cada 1% de diferencia en la variable independiente (mes), en promedio la diferencia en la variable dependiente es 0.18%.

4.Modelo Log-Lin

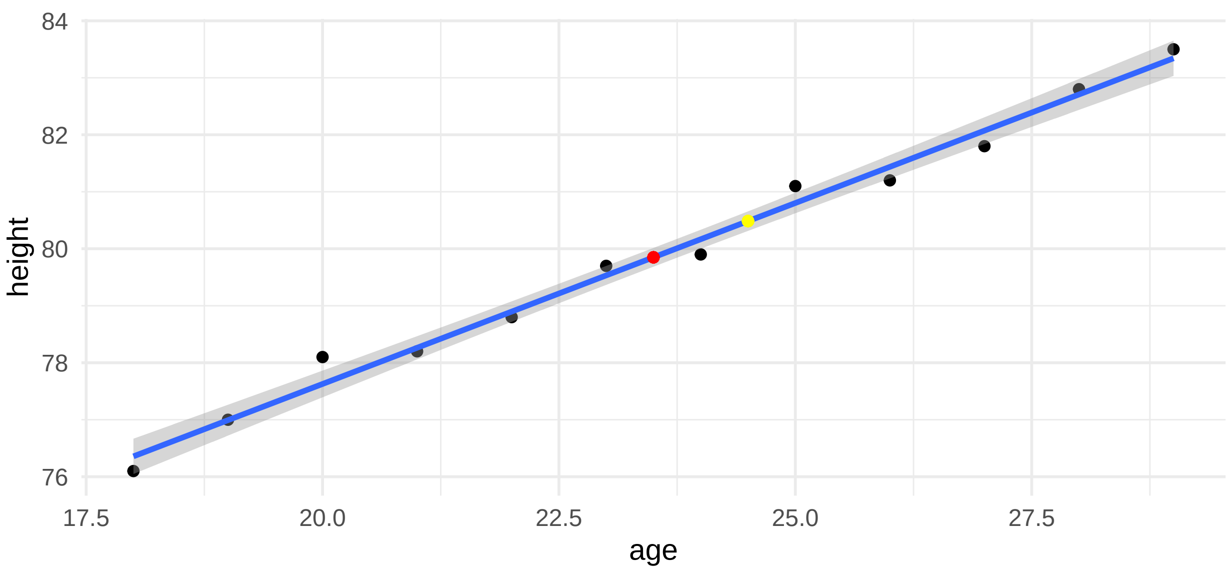
```
model_est <-
  linear_reg() %>%
  fit(height_log ~ age, data = datos_loglog)

tidy(model_est)
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)  4.19     0.00646      649.  1.85e-24
## 2 age          0.00796  0.000272      29.3 5.07e-11
```

```
glance(model_est)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared   sigma statistic  p.value    df logLik   AIC   BIC
##       <dbl>         <dbl>   <dbl>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1     0.988         0.987 0.00325      856.  5.07e-11     1   52.8 -99.6 -98.2
## # ... with 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

```
datos_loglog %>%
  ggplot(aes( x= age, y = height)) +
  geom_point()+
  geom_smooth(method = "lm")+
  geom_point(x = mean(datos_loglog$age), y = mean(datos_loglog$height), color = "red")+
  geom_point(x = mean(datos_loglog$age)+1, y = mean(datos_loglog$height)*1.007956085, color = "yellow")+
  theme_minimal()
```



Es decir, si aumentas en 1 unidad la variable independiente (mes), entonces en promedio la variable dependiente aumenta en 0.795%

5.Modelo Lin-Log

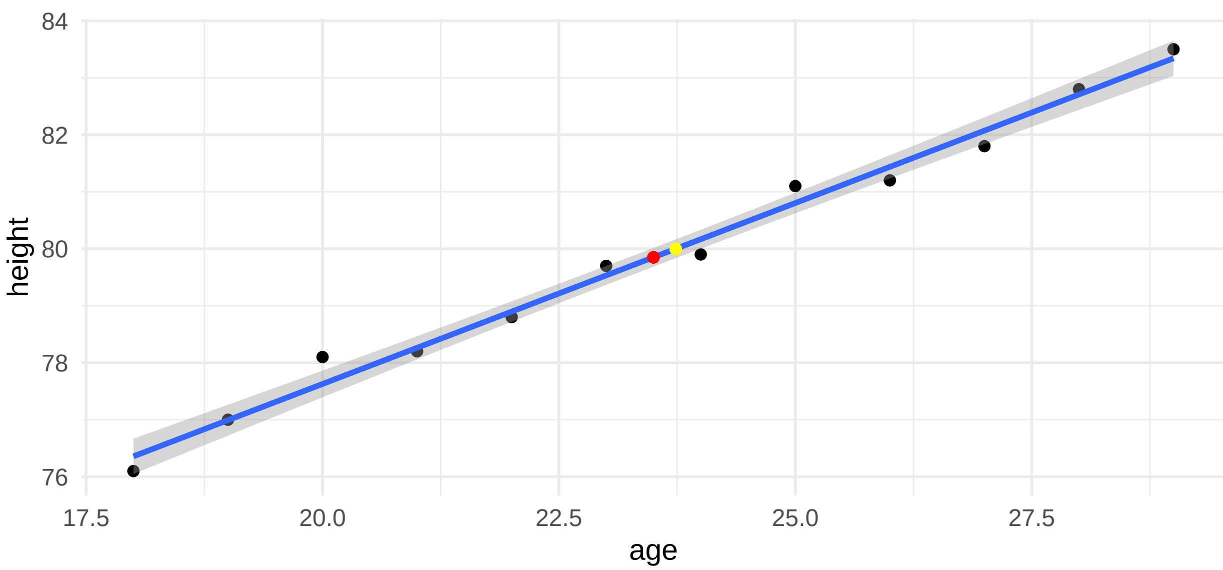
```
model_est <-
  linear_reg() %>%
  fit(height ~ age_log, data = datos_loglog)

tidy(model_est)
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)     33.7     1.80       18.7 4.07e- 9
## 2 age_log         14.7     0.571      25.7 1.85e-10
```

```
glance(model_est)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC
##       <dbl>         <dbl> <dbl>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1     0.985         0.984 0.295      659. 1.85e-10     1  -1.29  8.59  10.0
## # ... with 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

```
datos_loglog %>%
  ggplot(aes( x= age, y = height)) +
  geom_point()+
  geom_smooth(method = "lm")+
  geom_point(x = mean(datos_loglog$age), y = mean(datos_loglog$height), color = "red")+
  geom_point(x = mean(datos_loglog$age)*1.01, y = mean(datos_loglog$height)+0.146677, color = "yellow")+
  theme_minimal()
```



Es decir, un aumento de 1% en la variable independiente, aumenta en promedio 0.146676% la dependiente (¿por que no sería en este caso 14.6676?)

6.Teoria

6.1. Modelo Log-Log (Elasticidad constante)

Sea el modelo:

$$Y_i = \beta_0 X_1^{\beta_1} \exp^{e_i}$$

,

al aplicar logaritmo de ambos lados podemos transformarlo como:

$$ln(Y_i) = ln(\beta_0) + \beta_1 \ln(X_1) + e_i$$

En particular podemos reescribir a $ln(\beta_0)$ como $alpha$, a $ln(Y_i)$ como Y' y a $ln(X_i)$ como X' y tendríamos un caso particular del modelo de regresión:

$$Y' = \alpha + \beta_1 X_1' + e_i$$

Este modelo también recibe el nombre de modelo log-log, doble-log o log-lineal. Una caracteristica atractiva del modelo log-log es que el coeficiente de la pendiente β_1 mide la elasticidad de Y con respecto de X, es decir: $\frac{\Delta\%Y}{\Delta\%X}$. Una característica especiales del modelo log-log es que el modelo supone que el coeficiente de la elasticidad entre Y y X se mantiene contante en el tiempo (elasticidad constante).

Entonces tendríamos la elasticidad-edad

6.2 Modelo Log-Lin (Crecimiento exponencial)

Este es un modelo que comúnmente se llama semilogaritmico y busca medir la tasa de crecimiento. Cuando tenemos modelos que involucran tasas de crecimiento de variables (por ejemplo población, PIB, oferta monetaria, empleo, productividad), tenemos un modelo del estilo:

$$Y_t = Y_0(1 + r)^t$$

En el caso de la población, Y_0 podriamos verla como la población en el año $t = 0$ y donde r es la tasa de crecimiento compuesta. Por ejemplo, si en el año 2010 había 100 millones de habitantes en México, en el año 2011, con una tasa de crecimiento de $r = 0.02$, $Y_1 = 100(1 + .02)^1 = 102$ es decir en 2011 habría 102 millones de habitantes. Siguiendo el ejemplo, con la misma tasa de crecimiento en el año 2012 habría $Y_2 = 100(1 + .02)^2 = 104.04$ millones de habitantes. De la fórmula presentada arriba, al aplicar logaritmo natural:

$$log(Y_t) = log(Y_0) + tlog(1 + r)$$

Si nombramos $\beta_0 = log(Y_0)$ y $\beta_1 = log(1 + r)$, podemos escribirlo como:

$$log(Y_t) = \beta_0 + \beta_1 t$$

Este modelo es lineal, pero la diferencia es que la variable dependiente tiene aplicado el logaritmo. Estos modelos se conocen como semilog porque solo una variable aparece en forma logaritmica. En este modelo, el coeficiente de la pendiente mide el cambio propocional constante relativo en Y para un cambio absoluto en el valor de la regresora:

$$\beta_1 = \frac{\Delta\%Y}{\Delta X}$$

Esto es un cambio porcentual o tasa de crecimiento en Y ocasionada por un cambio absoluto en X (en algunos libros se conoce como la semielasticidad)

6.3 Modelo Lin-Log (Rendimientos decrecientes)

A diferencia del modelo pasado, en este caso la variable que tiene logaritmo es la variable independiente:

Sea el modelo: $Y_i = \beta_0 + \beta_1 log(X_i)$

Este modelo se conoce como **modelo lin-log**.

Cuando se transforma el modelo de esta manera, la correspondiente β_1 mide el cambio absoluto de Y vs el cambio porcentual de X, es decir: $\frac{\Delta Y}{\Delta \% X}$

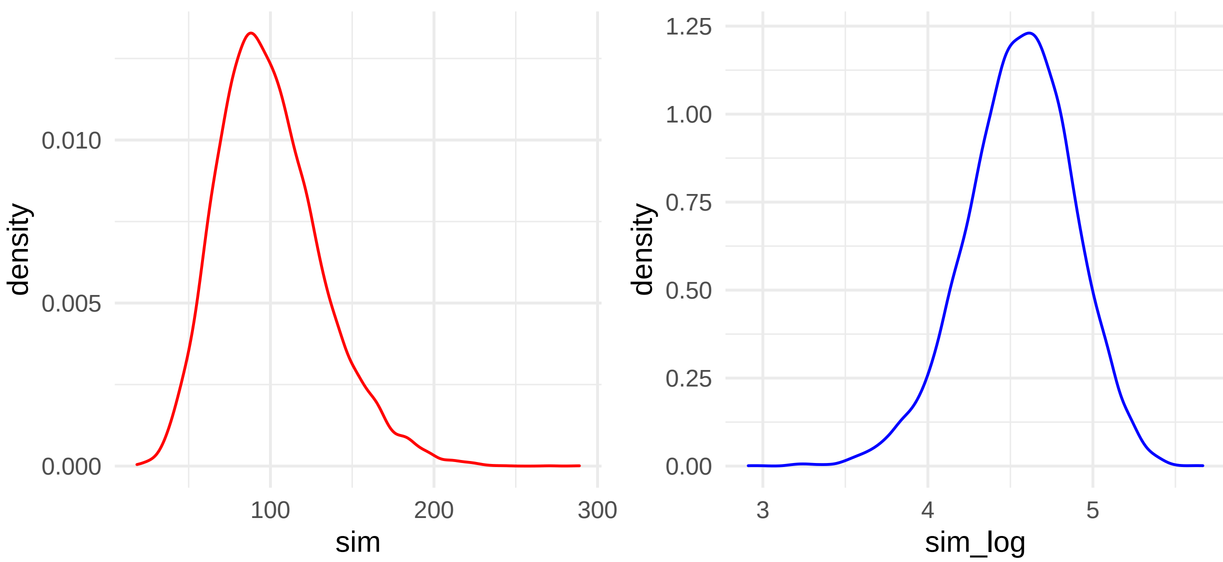
¿Cuándo es útil el modelo lin-log? Aunque no se ven aplicaciones inmediatas, hay casos particulares donde los podemos utilizar. Por ejemplo en el modelo de gasto de Engel se postuló que el gasto total que se dedica a los alimentos tiende a incrementarse en progresión aritmética, mientras que el gasto total en progresión geométrica

6.4 Otras propiedades de aplicar logaritmo

En general otro beneficio que tenemos de aplicar la transformación logarítmica es que se emplea para reducir la heteroscedasticidad, así como la asimetria.

Por ejemplo:

```
library(gridExtra)
library(moments)
datos <-
  data.frame(sim = rgamma(10000, shape = 10, scale = 10)) %>%
  mutate(sim_log = log(sim))

grid.arrange(
datos %>%
  ggplot()+
  geom_density(aes(x = sim), color = "red")+
  theme_minimal(),
datos %>%
  ggplot()+
  geom_density(aes(x = sim_log), color = "blue")+
  theme_minimal(), ncol = 2
)
```




```
moments::skewness(datos$sim)
```

```
## [1] 0.6647848
```

```
moments::skewness(datos$sim_log)
```

```
## [1] -0.3205516
```

```
  #geom_density(aes(x = sim_log), color = "red")
```

7. Resumen

```r
knitr::kable(tibble(
  Modelo = c("Lineal", "Lineal estandarizado", "log-log", "lin-log", "log-lin"),
  `Si x aumenta` = c("1 unidad", "1 sd", "1%", "1%", "1 unidad"),
  `Entonces y incrementa` = c("b1 unidades", "b1 sd", "b1 %", "b1 unidades", "b1%")
))
```

Modelo	Si x aumenta	Entonces y incrementa
Lineal	1 unidad	b1 unidades
Lineal estandarizado	1 sd	b1 sd
log-log	1%	b1 %
lin-log	1%	b1 unidades
log-lin	1 unidad	b1%

8.Ejercicio

Utilizando los datos de House Price, busque predecir el precio de la casa *SalePrice* usando el total de metros cuadrados construidos de la casa (TotalBsmtSF+X1stFlrSF+X1stFlrSF):

```r
datos <-
  read_csv("https://raw.githubusercontent.com/savrgg/class_ITAM_metodos/main/notas_r/HousePrice.csv") %>
  data.frame() %>%
  select(SalePrice, TotalBsmtSF, X1stFlrSF, X2ndFlrSF)
datos %>% head
```

```
##   SalePrice TotalBsmtSF X1stFlrSF X2ndFlrSF
## 1    208500         856       856       854
## 2    181500        1262      1262         0
## 3    223500         920       920       866
## 4    140000         756       961       756
## 5    250000        1145      1145      1053
## 6    143000         796       796       566
```

1) Determine el modelo original, grafiquelo e interprételo

2) Determine el modelo log-log, grafiquelo e interprételo

3) Determine el modelo log-lin, grafiquelo e interprételo

4) Determine el modelo lin-log, grafiquelo e interprételo

5) Determine si la variable es normal (Jarque-Bera & qqplot) y si no lo es calcule el logaritmo para ver como mejora.