01-Regresion Lineal

Adrian

20/1/2022

Introduccion a la regresion lineal

Objetivo: Describir la relacion entre la variable independiente (x) y la variable dependiente (y).

Habra que buscar una funcion y = f(x) cuya grafica se aproxime lo maximo posible a nuestros pares ordenados

Esta funcion nos dara un modelo matematico de como se comportan estas observaciones.

Primera opcion

Comprobar si satisfacen una relacion lineal: $y = ax + b$

imponiendo que la suma de los cuadrados de las diferencias entre los valores y_i y sus aproximaciones $\tilde{y_i} = ax_i + b$ sea minima.

Calcular una recta de regresion lineal

Lo ideal es trabajar con Data Frames.

```
body = read.table("../../../data/bodyfat.txt", header = T)
head(body, 3)
```

```
##   Density  Fat Age Weight Height Neck Chest Abdomen  Hip Thigh Knee Ankle
## 1  1.0708 12.3  23 154.25  67.75 36.2  93.1    85.2 94.5  59.0 37.3  21.9
## 2  1.0853  6.1  22 173.25  72.25 38.5  93.6    83.0 98.7  58.7 37.3  23.4
## 3  1.0414 25.3  22 154.00  66.25 34.0  95.8    87.9 99.2  59.6 38.9  24.0
##   Biceps Forearm Wrist
## 1   32.0    27.4  17.1
## 2   30.5    28.9  18.2
## 3   28.8    25.2  16.6
```

Vamos a trabajar con las variables fat y weight

```
body2 = body[,c(2,4)]
names(body2) = c("Grasa", "Peso")
str(body2)
```

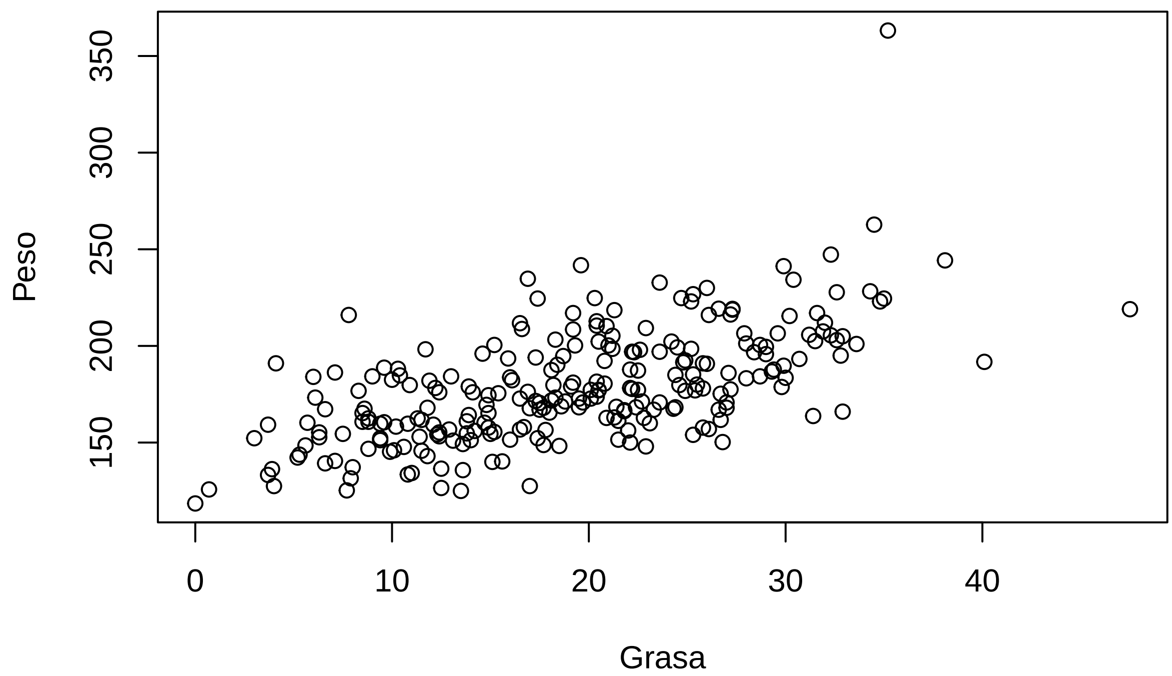
```
## 'data.frame':    252 obs. of  2 variables:
##  $ Grasa: num  12.3 6.1 25.3 10.4 28.7 20.9 19.2 12.4 4.1 11.7 ...
##  $ Peso : num  154 173 154 185 184 ...
```

```r
head(body2, 3)
```

```
##   Grasa   Peso
## 1  12.3 154.25
## 2   6.1 173.25
## 3  25.3 154.00
```

Estimar una funcion lineal entre el peso y la grasa Es recomendable empezar con una representacion grafica.

```r
plot(body2)
```



Para calcular la recta de regresion

```r
# Modelo lineal que explica el peso en funcion de la grasa
# Primero van las dependientes
# Opcion 1:
lm(body2$Peso~body2$Grasa)
```

```
##
## Call:
## lm(formula = body2$Peso ~ body2$Grasa)
##
## Coefficients:
```

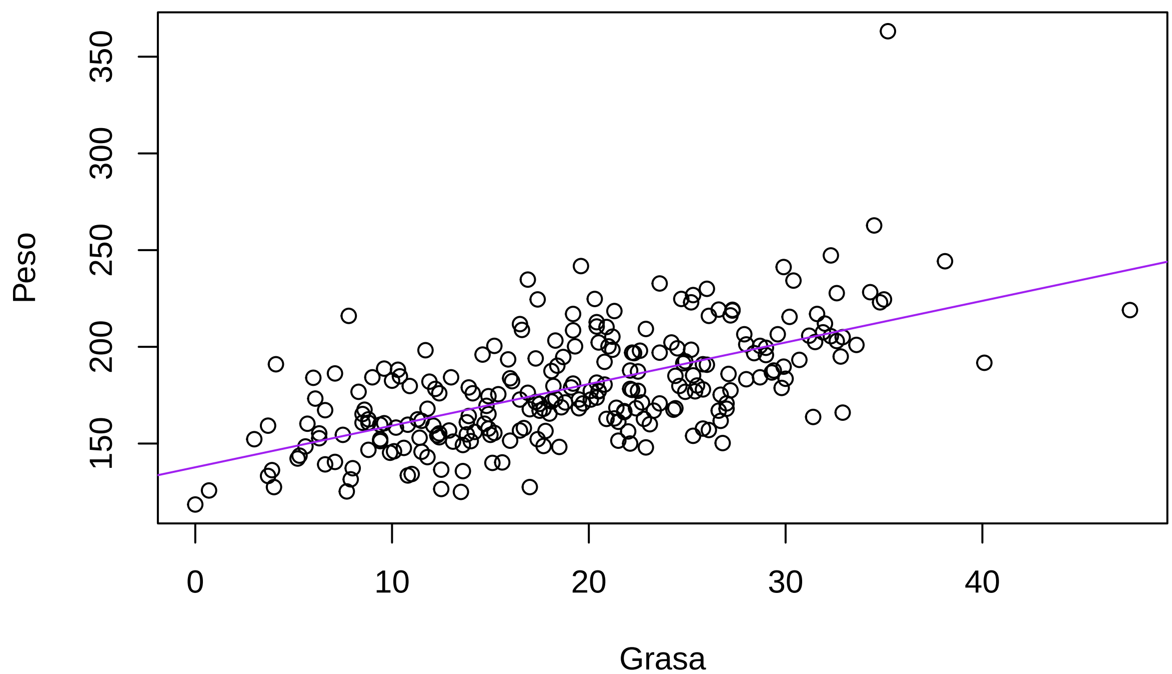
```
## (Intercept)  body2$Grasa
##     137.738        2.151
```

```r
# Opcion 2:
lm(Peso~Grasa, data = body2)
```

```
##
## Call:
## lm(formula = Peso ~ Grasa, data = body2)
##
## Coefficients:
## (Intercept)        Grasa
##     137.738        2.151
```

Ahora podemos superponer la funcion obtenida anteriormente en nuestro grafico usando abline()

```r
plot(body2)
abline(lm(Peso~Grasa, data = body2), col = "purple")
```



Coeficiente de determinacion

Util para evaluar numericamente si la relacion lineal es significativa o no. Se expresa con R^2.

Si $R^2 > 0.9$ consideraremos que es un ajuste bueno.

La funcion summary aplicada a lm nos muestra los contenido de este objeto. Encontramos Multiple R-squared, que es el R^2

```
# Coeficiente de determinacion
summary(lm(Peso~Grasa, data = body2))$r.squared
```

```
## [1] 0.3750509
```

Transformaciones logaritmicas

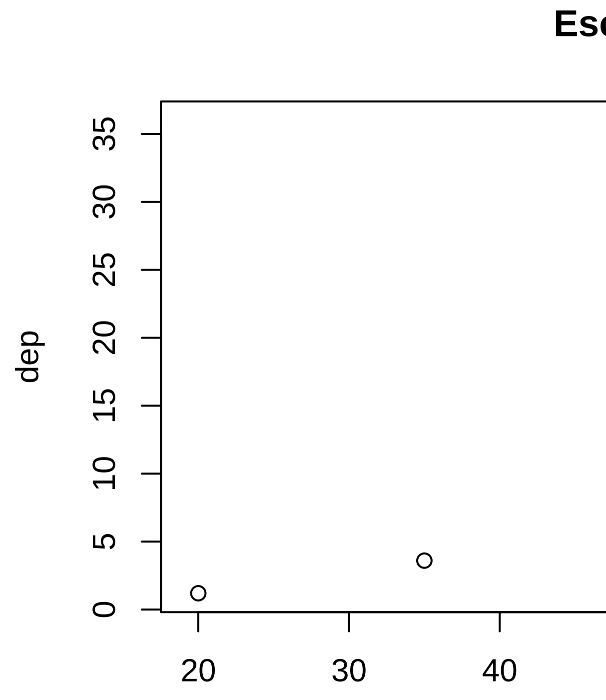
No siempre encontraremos dependencias lineales, tambien encontraremos otro tipo de dependencias como potencias o exponenciales.

Se pueden transformar a lineales mediante un cambio de escala. - Escala semilogaritmica: Eje de abscisas (x) esta en escala lineal y el de ordenadas en escala logaritmica - Escala doble logaritmica: Ambos ejes estan en escala logaritmica

Si $log(y) = ax + b$, entonces "y" sigue una ley exponencial frente a "x"

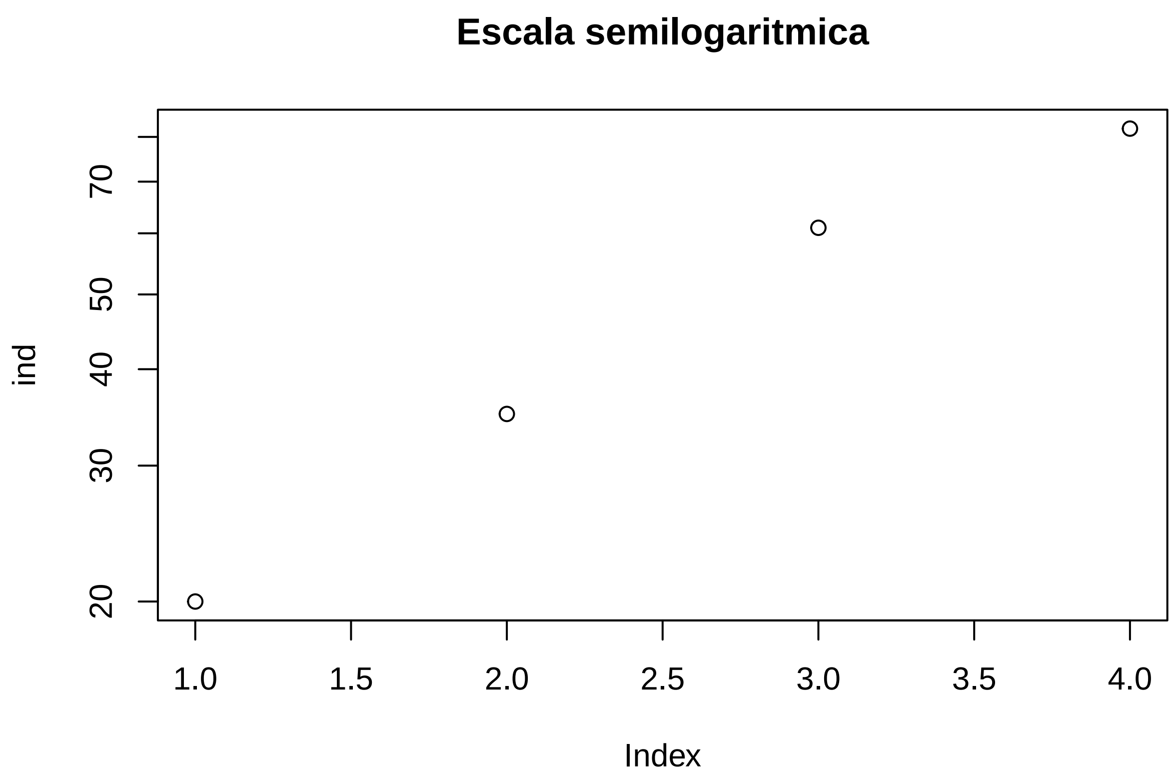
Si $log(y) = log(x) + b$, entonces "y" sigue una ley potencial frente a "x"

```
dep = c(1.2,3.6,12,36)
ind = c(20,35,61,82)

plot(ind, dep, main = "Escala lineal")
```



Ejemplo Transformaciones logaritmicas - Modelo Exponencial

```
plot(ind, log = "y", main = "Escala semilogaritmica")
```



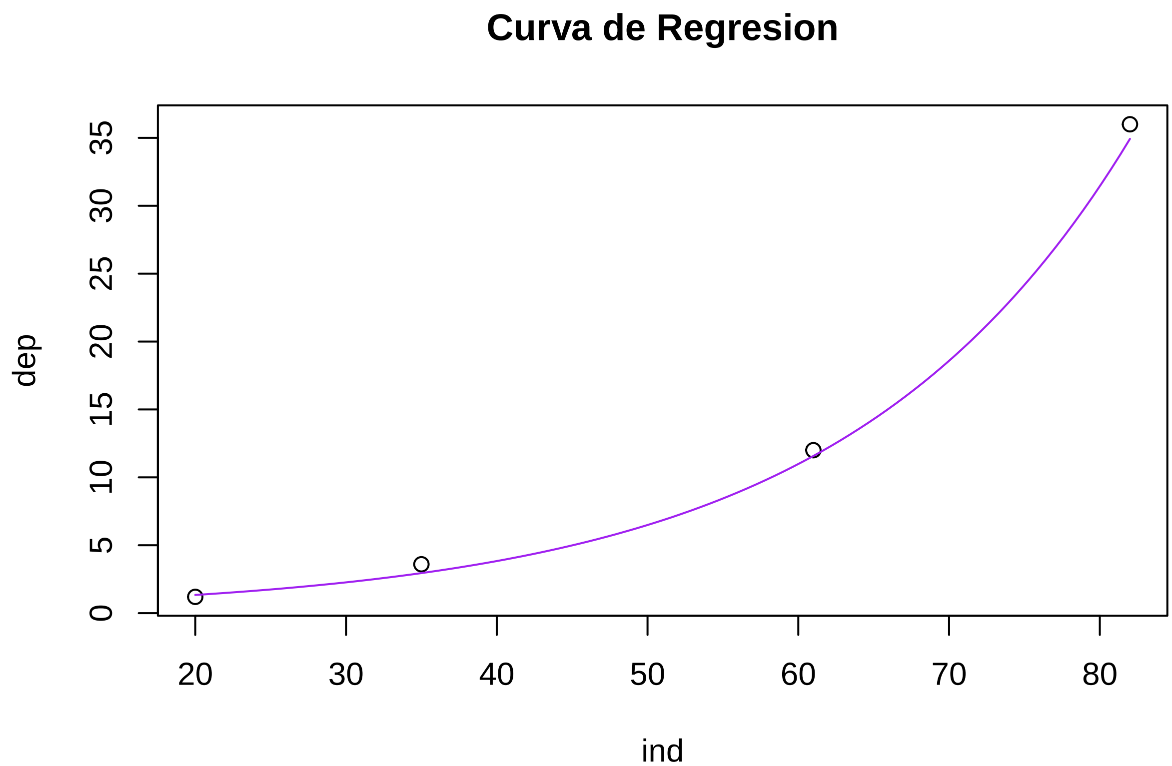

```
# Modelo lineal
lm(log10(dep)~ind)
```

```
##
## Call:
## lm(formula = log10(dep) ~ ind)
##
## Coefficients:
## (Intercept)          ind
##    -0.32951      0.02318
```

```
# Coeficiente de determinacion
summary(lm(log10(dep)~ind))$r.squared
```

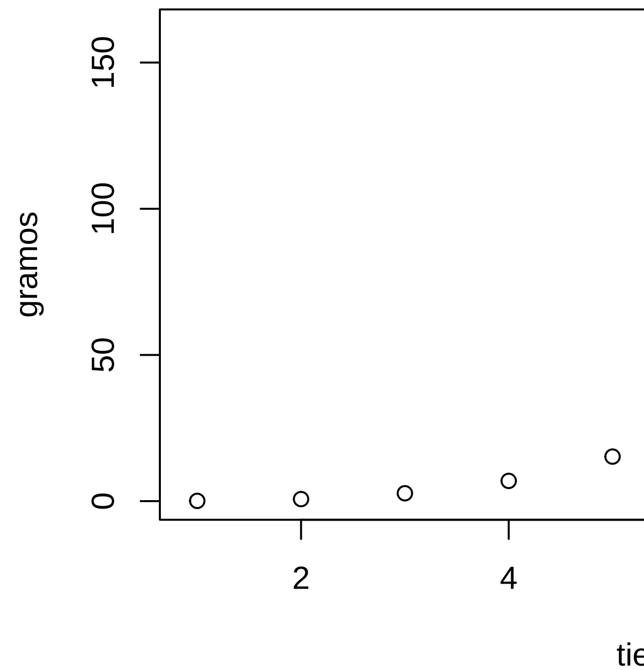
```
## [1] 0.9928168
```

```
plot(ind, dep, main = "Curva de Regresion")
curve(1.054^x*0.468, add = T, col = "purple")
```



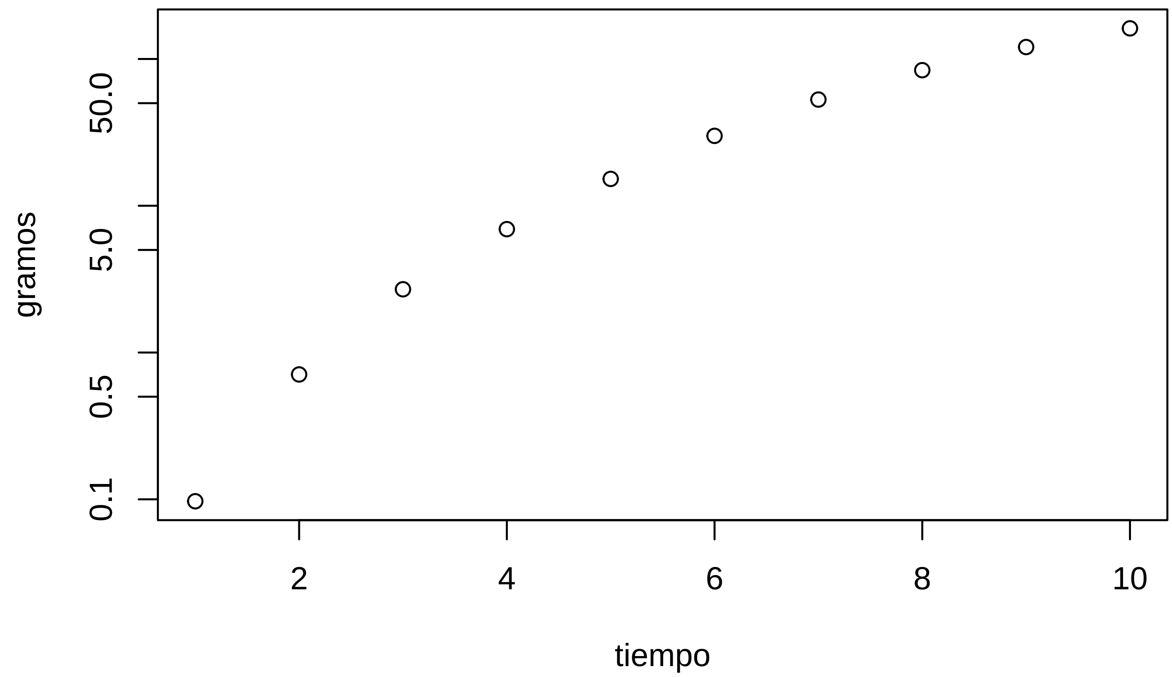

```
tiempo = 1:10
gramos = c(0.097,0.709,2.698,6.928,15.242,29.944,52.902,83.903,120.612,161.711)

d.f = data.frame(tiempo, gramos)
```

```
plot(d.f)
```

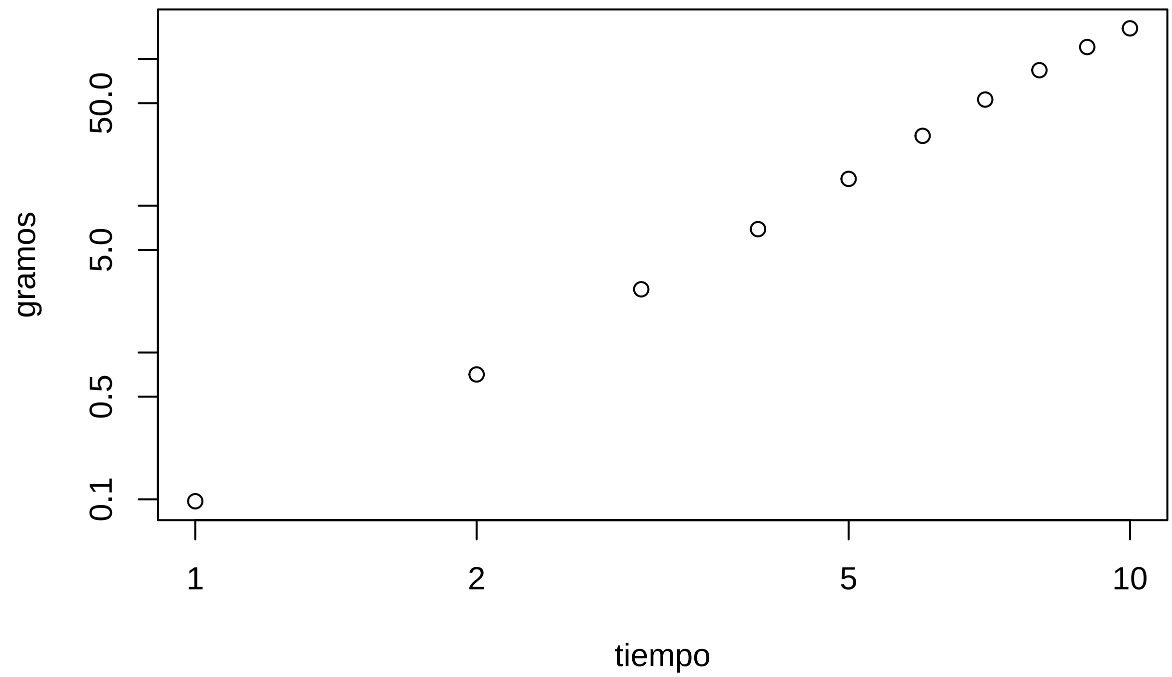


Ejemplo Transformaciones logaritmicas - Modelo Potencial

```
# Escala semilogaritmica
plot(d.f, log="y")
```




```
# Escala doblelogaritmica
plot(d.f, log="xy")
```



Obtener el modelo lineal

```
# Modelo lineal
lm(log10(gramos)~log10(tiempo), data = d.f)
```

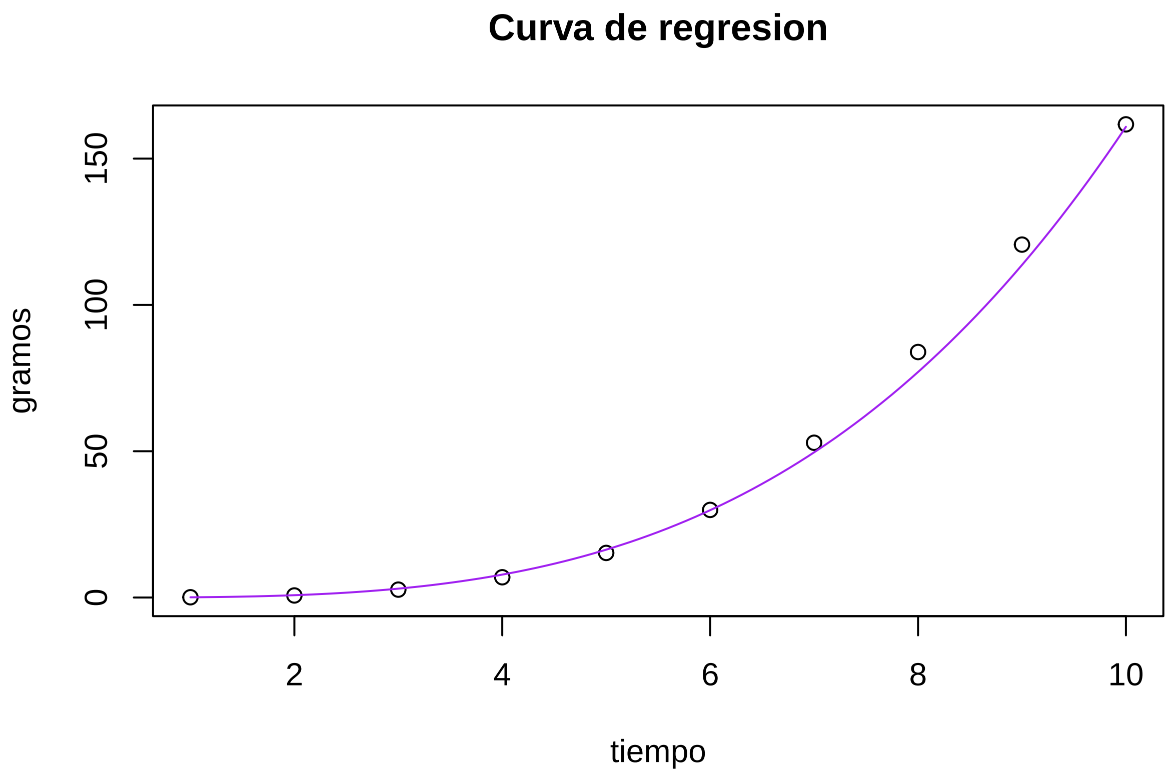
```
##
## Call:
## lm(formula = log10(gramos) ~ log10(tiempo), data = d.f)
##
## Coefficients:
##   (Intercept)  log10(tiempo)
##        -1.093          3.298
```

```
# Coeficiente de determinacion
summary(lm(log10(gramos)~log10(tiempo), data = d.f))$r.squared
```

```
## [1] 0.9982009
```

Curva de Regresion

```
plot(d.f, main = "Curva de regresion")
curve(x^(3.298)*0.081, add = T, col = "purple")
```


Curva de regresion
gramos
0
50
100
150
2
4
6
8
10
tiempo