Regresión lineal simple y polinomial: teoría y práctica

Pablo Vinuesa, CCG-UNAM. http://www.ccg.unam.mx/~vinuesa/

v2, 2 de Agosto, 2018

Contents

1 Regresión lineal simple y múltiple: teoría y práctica

Este tema es parte del **Taller 2 - Análisis exploratorio y estadístico de datos biológicos usando R**, de la Universidad Nacional Autónoma de México, impartido entre 30 de Julio y 3 de Agosto de 2018 en el Centro de Ciencias Genómicas. Para más información consultar la página del taller en: http://congresos.nnb.unam.mx/TIB2018/t2-analisis-exploratorio-y-estadistico-de-datos-biologicos-usando-r/. .

El matrial se distribuye desde el repositiorio GitHub curso_Rstas

La parte teórica está basada en (Crawley 2015), (Davies 2016) y (A. P. Field, Miles, and Field 2012).

Este documento está aún en construcción y es generado con R (R Core Team 2018), rstudio (RStudio Team 2016), knitr (Xie 2018), rmarkdown (Allaire et al. 2018), pandoc (MacFerlane 2016) y $LaTeX$.

1.1 Introducción: el concepto de regresión

El **análisis de regresión** engloba a un conjunto de métodos estadísticos que usamos cuando tanto la **variable de respuesta** como la **la(s) variable(s) predictiva(s)** son **contínuas** y queremos predecir valores de la primera en función de valores observados de las segundas. En esencia, el análisis de regresión consiste en ajustar un modelo a los datos, estimando coeficientes a partir de las observaciones, con el fin de predecir valores de la variable de respuesa a partir de una (**regresión simple**) o más variables (**regresión múltiple**) predictivas o explicativas.

El análisis de regresión juega un papel central en la estadística moderna y se usa para:

- **identificar** a las variables predictivas relacionadas con una variable de respuesta
- **describir** la forma de la relación entre estas variables y para derivar una función matemática óptima que modele esta relación
- **predecir** la variable de respuesta a partir de la(s) explicativas o predictoras

1.1.1 Tipos de regresión

El término regresión puede ser confuso porque existen muchas variantes especializadas de regresión. Además, *R* tiene muchas funciones para ajustar una gran gama de modelos de regresión.

Tabla 1. Algunos tipos básicos de regresión (hay muchos más)

Tipo de regresión	Uso típico
lineal simple	Predicción de una variable de respuesta cuantitativa a partir de una variable predictora cuantitativa
polinomial	Predicción de una variable de respuesta cuantitativa a partir de una variable predictora cuantitativa, donde la relación se modela como una función polinomial de orden n
lineal múltiple	Predicción de una variable de respuesta cuantitativa a partir de dos o más variables predictoras cuantitativas
multivariada	Predicción de más de una variable de respuesta cuantitativa a partir de una o más variables predictoras cuantitativas
logística	Predicción de una variable categórica a partir de una o más predictoras
de Poisson	Predicción de una variable de respuesta que representa un conteo a partir de una o más predictoras
no lineal	Predicción de una variable de respuesta cuantitativa a partir de una o más predictoras, donde el modelo no es lineal
robusta	Predicción de una variable de respuesta cuantitativa a partir de una o más predictoras, usando una aproximación resistente al efecto de observaciones influyentes

1.1.2 Modelado estadístico: selección de modelos, ajuste a los datos y varianza en la estima

Dadas dos variables cuantitativas existen virtualmente cientos de modelos que podrían describir la relación matemática entre ellas. El reto está en elegir el modelo que mejor se ajuste a estos datos para minimizar el error en la estima que se haga a partir del modelo.

Usamos los modelos para estimar el valor promedio de la variable de respuesta en función de parámetros estimados de los datos. De manera general, podemos predecir valores de la variable de respuesta usando esta fórmula:

$estima_i = (modelo) + error_i$.

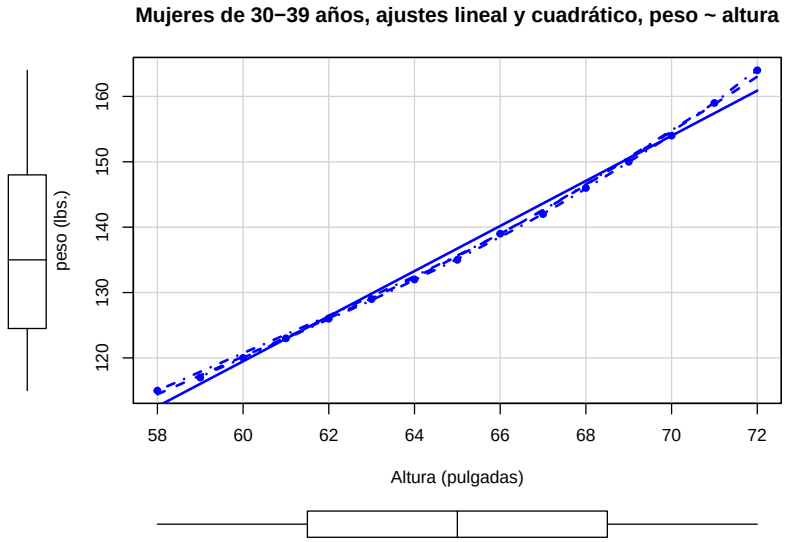
Para obtener la máxima precisión (mínimo $error_i$) en nuestra estima o predicción, tendremos que:

1. elegir una familia de modelos adecuados a nuestros datos (modelos lineales, polinomiales, exponenciales, no lineales ...)

2. determinar el grado de parametrización adecuado del modelo

3. obtener estimas de máxima verosimilitud de dichos parámetros

Sólo así podremos llegar a un compromiso óptimo entre realismo del modelo, grado de ajuste del modelo a los datos, y mínima varianza de la estima.
Veamos como ejemplo el ajuste de modelos lineal y cuadrático (polinomial) para predecir el peso de mujeres de 30-39 años en función de su altura (peso ~ altura) ¿Cuál de ellos se ajusta mejor a los datos?



- Estadísticas asociadas al modelo lineal: $\widehat{peso} = -87.52 + 3.45 * altura$

- Residual standard error: 1.525 on 13 degrees of freedom
- Multiple R-squared: 0.991, Adjusted R-squared: 0.9903
- F-statistic: 1433 on 1 and 13 DF, p-value: 1.091e-14
- Estadísticas asociadas al modelo polinomial: $\widehat{peso} = 261.88 - 7.35 * altura + 0.083 * altura^2$

- Residual standard error: 0.3841 on 12 degrees of freedom
- Multiple R-squared: 0.9995, Adjusted R-squared: 0.9994
- F-statistic: 1.139e+04 on 2 and 12 DF, p-value: $<$ 2.2e-16

Visualmente es claro que $el modelo polinomial (cuadrático) se ajusta mejor a los datos$, como indican los valores de R^2 ($coef.\ de\ determinación$) y los $valores - p$ asociados al estadístico$-F$.

¿Pero es este ajuste significativamente mejor? Comparando el ajuste de ambos modelos mediane el **criterio de información de Akaike (AIC)**, confirmamos que el modelo polinomial es mucho mejor: aunque tiene un parámetro libre más a estimar de los datos (el coeficiente $b2 * altura^2$) que el modelo lineal, el ajuste es $\sim$ 40 unidades de AIC mejor, por lo que rechazamos el modelo lineal en favor el cuadrático.

```
##      df      AIC
## fit   3 59.08158
## fit2  4 18.51270
```

Aprenderemos más adelante los detalles relativos a la **selección del mejor modelo de regresión**, que es *un aspecto esencial del modelado matemático.*

En este capítulo nos enfocaremos en métodos que encajan en el rubro de **modelos de regresión de mínimos cuadrados ordinarios (OLS)**. El énfasis estará en los **modelos lineales simple y polinomiales**, es decir, modelos en los que la analizamos el *comportamiento de la variable de respuesta cuantitativa en función de una variable predictora o regresora cuantitativa*, asumiendo una relación lineal o curvilínea entre ellas.

1.2 Regresión lineal simple

Se basa en modelos lineales con la fórmula genaral:

$$Y_i = (a + bX_i) + \epsilon_i$$

donde:

- **a** = punto de corte en el eje de ordenadas (y), cuando $x = 0$
- **b** = pendiente o gradiente de la recta, que son los **coeficientes de regresión**
- ϵ_i corresponde al **término de resíduos**, que representa la diferencia entre el valor observado y el estimado para el individuo i.

Los coeficientes de regresión los tenemos que estimar de los datos, usando el **método de mínimos cuadrados**, basado en las siguientes fórmulas, y el criterio de optimización de **máxima verosimilitud**.

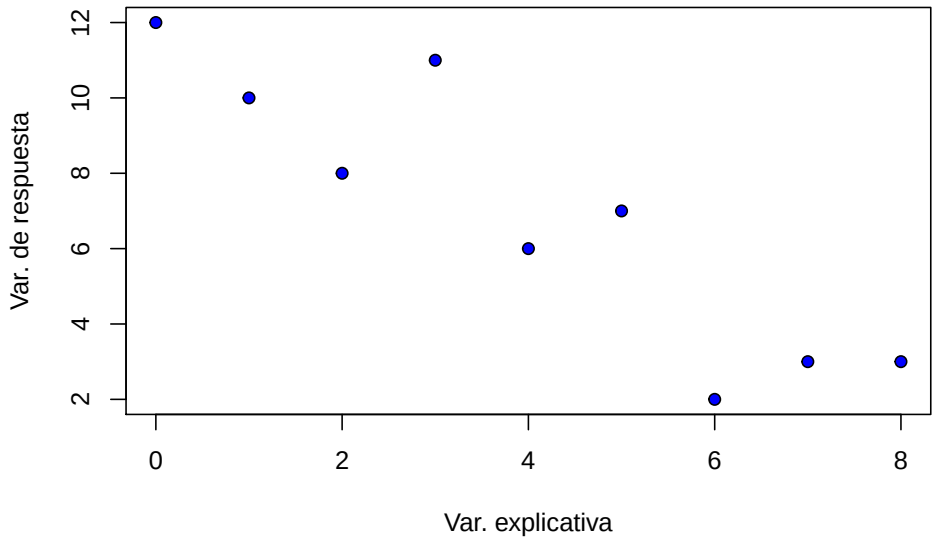
$$b = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(yi - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} = \frac{\sum xy - \frac{\sum x \sum y}{n}}{\sum x^2 - \frac{(\sum x)^2}{n}} = \frac{SSXY}{SSX} = \frac{suma\, corregida\, de\, productos}{suma\, de\, cuadrados\, de\, x}$$

$$a = \bar{y} - b\bar{x} = \frac{\sum y}{n} - b\frac{\sum x}{n}$$

Una vez estimados estos *coeficientes* los sustituímos en la fórmula general de la recta, definiendo así el *modelo lineal* que ajustamos en el ejemplo gráfico anterior: $\widehat{peso} = -87.52 + 3.45 * altura$. Nótese que en este caso el punto de corte no tiene mucho sentido ya que no encontraremos una persona de 0 pulgadas de altura.

1.2.1 Cálculo manual de los coeficientes de regresión de un modelo lineal

Es fácil estimar "a ojo" los coeficientes de regresión, razonablemente precisos, de un modelo lineal.
Ajustemos un modelo lineal a los siguientes datos:



En pseudocódigo los pasos a seguir son:

1. Evaluar el decremento de la variable de respuesta: ha decrecido de 12 a 2, es decir un decremento de -10
2. Determinar el rango de variación de la variable regresora: $x = +8$
3. Calcular la pendiente o gradiente de la variación de la variable de respuesta: $b \approx -10/8 = -1.25$

4. Determinar el punto de corte: ¿Cuánto vale y cuando $x = 0$?, $a \approx 12$

Por tanto la recta de regresión simple ajustada manualmente a los datos tiene la fórmula: $y = 12 - 1.25x$

Estos coeficientes no difieren mucho de los obtenidos al ajustar un modelo lineal con R:

```
# Noten que la var. de respuesta va a la izda de la tilde ' ~' y se lee como
# modelado de Y en función de X. Esta notación corresponde a lo que se conoce como
# una fórmula en R. Las fórmulas son centrales en el modelado estadístico y tienen una
# sintaxis particular y compleja, que hay que aprender para manejar R eficientemente
lm(dfr1$Y ~ dfr1$X)
```

```
##
## Call:
## lm(formula = dfr1$Y ~ dfr1$X)
##
## Coefficients:
## (Intercept)       dfr1$X
##      11.756       -1.217
```

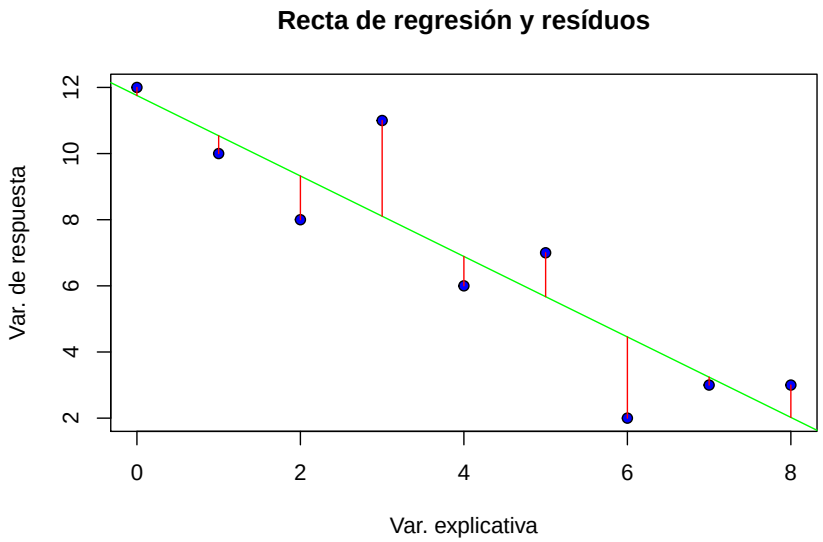
1.2.2 ¿Cómo calcula R los coeficientes de un modelo lineal de regresión?

R busca las estimas de máxima verosimilitud de los parámetros: $L \approx Pr(D|M)$. Es decir, habiendo seleccionado un modelo lineal, **necesitamos encontrar los valores de los parámetros del Modelo (a y b) que maximizan la probabilidad de observar los Datos**.

1.2.2.1 El método de mínimos cuadrados

Este método **garantiza encontrar el valor de máxima verosimilitud de los parámetros del modelo lineal para un conjunto de datos**, al *minimizar la magnitud de la diferencia entre los puntos y la recta*. Veamos ésto gráficamente, al añadir la recta más ajustada a los datos mediante la función *abline*() y dibujando líneas que representan los **resíduos**, es decir, la distancia de cada punto a la recta.

```
# 1. Graficamos los puntos
plot(dfr1$X, dfr1$Y, pch=21, bg="blue", ylab= "Var. de respuesta",
     xlab="Var. explicativa", main = "Recta de regresión y resíduos")

# 2. Añadimos a la gráfica de dispersión la recta de regresión
abline(lm(dfr1$Y ~ dfr1$X), col="green")

# Podemos obtener el valor predicho de Y para cada valor de X con predict()
fitted <- predict(lm(dfr1$Y ~ dfr1$X))

# Con un bucle for y la función lines() podemos representar los resíduos
for(i in 1:9)
{
  lines( c(dfr1$X[i],dfr1$X[i]), c(dfr1$Y[i], fitted[i]), col="red")
}
```




```
# Veamos los valores Y ajustados que calculamos previamente:
fitted
```

```
##         1         2         3         4         5         6         7
## 11.755556 10.538889  9.322222  8.105556  6.888889  5.672222  4.455556
##         8         9
##  3.238889  2.022222
```

Los resíduos describen la bondad de ajuste de la recta de regresión. Nuestro modelo de máxima verosimilitud se define como *el modelo que minimiza la suma de los cuadrados de estos resíduos.*

1.2.2.2 Simulación del algoritmo que usa R para minimizar la suma de cuadrados

R implementa un algoritmo para minimizar esta suma de cuadrados. Concretamente el algoritmo, mediante cáluculo diferencial (tomando derivadas) calcula el mínimo de la función de verosimilitud para cada coeficiente a partir de la fórmula:

$$d = y - \hat{y}$$

Dado $\hat{y}$ está sobre la recta $a + bx$, podemos reescribir la ecuación anterior como:

$$d = y - (a + bx) = y - a - bx$$

La estima por mínimos cuadrados de la pendiente de regresión (b) se calcula por tanto rotando la línea hasta que el **error de la suma de cuadrados**, SSE, sea minimizada. El algoritmo por tanto tiene que encontrar el mínimo de $\sum (y - a - bx)^2$. Tomando la derivada de SSE con respecto a b:

$$\frac{dSSE}{db} = -2 \sum x(y - a - bx)$$

Operando llegamos a que:

$$b = \frac{\sum xy - \frac{\sum x \sum y}{n}}{\sum x^2 - \frac{(\sum x)^2}{n}}$$

Que se abrevia como:

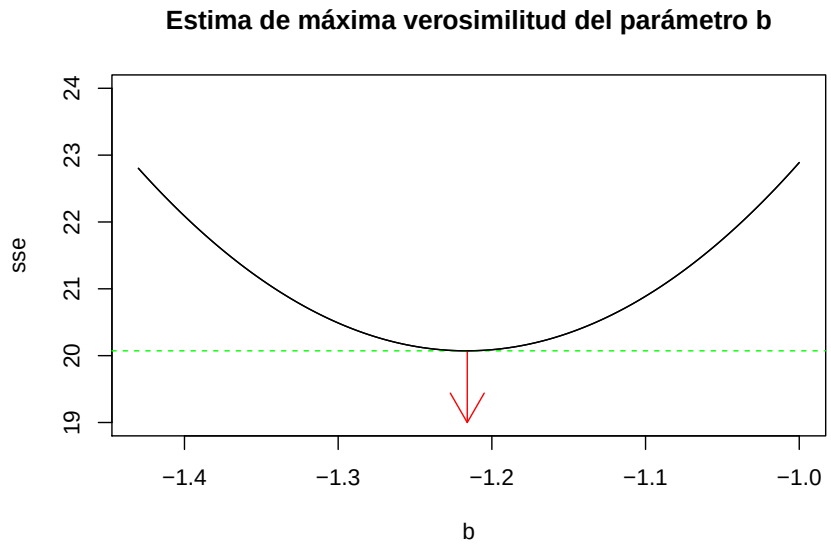
$$b = \frac{SSXY}{SSX} = \frac{suma\, corregida\, de\, productos}{suma\, de\, cuadrados\, de\, x}$$

Para que tengan una impresión de los cálculos involucrados, es útil graficar la suma de cuadrados de los resíduos SSE contra el valor del parámetro que estamos tratando de estimar. Veamos el caso de la *pendiente* b.

- Sabemos que la recta de regresión pasa por un punto en el centro de la nube de puntos, cuyas coordenadas son $(\bar{x}, \bar{y})$. La línea de mejor ajuste tendrá que pivotarse en torno a las medias de x e y, seleccionando aquella línea cuya pendiente b minimiza la suma de cuadrados de las distancias marcadas en rojo en la gráfica anterior.
- El valor de b que minimiza la SSE representa la estima de máxima verosimilitud de este parámetro. Veamos una simulación del algoritmo. Sabemos de nuestras estimas que $b \approx -1.25$.
- Por tanto vamos a localizar el valor de máxima verosimilitud en el rango de $-1.4 < b < -1.0$, estimando la SSE para cada valor de b en este rango.
- En pseudocódigo los pasos a realizar son:
- recorrer en pequeños incrementos el valor de b dentro del rango establecido
- calcular el punto de corte correspondiente $(a = \bar{y} - b\bar{x}\ )$
- estimar los valores ajustados de y para cada nivel de x $(y = a + bx)$
- calcular los resíduos asociados $(d = y - a - bx)$ a cada estima puntual
- elevar los resíduos al cuadrado y sumarlos $(\sum(y - a - bx)^2)$
- asociar este valor de $sse[i]$ con la estima actual de la pendient $b[i]$

Una vez ejecutado este algoritmo, produciremos una gráfica en forma de "U", con la suma de resíduos cuadrados SSE en el eje de ordenadas, expresada en función de la estima de la pendiente b. El mínimo de esta gráfica corresponderá a la estima de máxima verosimilitud de b para la cual se minimiza la SSE.

```
# estima de maxima verosimilitud de la pendiente

b <-  seq(-1.43,-1,0.002)
sse <- numeric(length(b))
for (i in 1:length(b)) {
a <- mean(dfr1$Y)-b[i]*mean(dfr1$X)
residual <- dfr1$Y - a - b[i]*dfr1$X
sse[i] <- sum(residual^2)
}

# generemos la gŕafica de sse ~ b
plot(b,sse,type="l",ylim=c(19,24),
     main = "Estima de máxima verosimilitud del parámetro b")

# añadamos una flecha en el mínimo
arrows(-1.216,20.07225,-1.216,19,col="red")

# dibujemos una recta verde que pase por el mínimo
abline(h=20.07225,col="green",lty=2)
lines(b,sse)
```




```
# el valor mínimo de sse lo obtenemos con la fución min()
min(sse)
```

```
## [1] 20.07225
```

```
# así obtenemos el valor de "b" asociado al valor mínimo de sse
b4minSSE <- b[which(sse==min(sse))]
b4minSSE
```

```
## [1] -1.216
```

1.2.2.3 Fórmulas se suma de cuadrados corregidas y cálculo de los coeficientes de regresión

En resumen, en base a las fórmulas antes enunciadas para los coeficientes de regresión:

$$b = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(yi - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} = \frac{\sum xy - \frac{\sum x \sum y}{n}}{\sum x^2 - \frac{(\sum x)^2}{n}} = \frac{SSXY}{SSX} = \frac{suma\ corregida\ de\ productos}{suma\ de\ cuadrados\ de\ x}$$

$$a = \bar{y} - b\bar{x} = \frac{\sum y}{n} - b\frac{\sum x}{n}$$

y recordando las fórmulas de las **sumas de cuadrados y productos corregidas**, que son absolutamente centrales en todo lo que seguirá sobre regresión y análisis de la varianza:

- suma total de cuadrados SSY:

$$SSY = \sum y^2 - \frac{(\sum y)^2}{n}$$

- suma de cuadrados de x SSX:

$$SSX = \sum x^2 - \frac{(\sum x)^2}{n}$$

- suma de productos corregida:

$$SSXY = \frac{\sum xy - \frac{\sum x \sum y}{n}}{\sum x^2 - \frac{(\sum x)^2}{n}}$$

podemos calcular fácilmente los coeficientes de regresión, como muestra el siguiente bloque de código:

```
SSX <- sum(dfr1$X^2)-sum(dfr1$X)^2/length(dfr1$X)
SSY <- sum(dfr1$Y^2)-sum(dfr1$Y)^2/length(dfr1$Y)
SSXY <- sum(dfr1$X*dfr1$Y)-sum(dfr1$X)*sum(dfr1$Y)/length(dfr1$X)

cat("SSX =", SSX, "| SSY =", SSY, "| SSXY =", SSXY, "\n")
```

```
## SSX = 60 | SSY = 108.8889 | SSXY = -73
```

```
# por tanto:
b <- SSXY/SSX
a <- (sum(dfr1$Y)/length(dfr1$Y)) - (b*sum(dfr1$X)/length(dfr1$X))

cat("los coeficientes de regresión: a =", a, "| b =", b, "\n")
```

```
## los coeficientes de regresión: a = 11.75556 | b = -1.216667
```

1.2.3 Determinación de la bondad del ajuste: *suma de cuadrados*, R y R^2

Lo que hemos visto hasta ahora es sólo la "mitad de la historia". Además de los parámetros a y b, necesitamos calcular la incertidumbre asociada a cada una de estas estimas. Es decir, necesitamos calcular el **error estándar** de cada parámetro de nuestro modelo de regresión.

El error estándar (SE) es la *desviación estándar de la distribución de muestreo de un estadístico*. Para un estadístico (p.ej. la *media*), nos indica cuánta variación existe entre *muestras* de la misma *población*. Por tanto valores grandes de SE indican que el estadístico no representa adecuadamente a la población de la que procede la muestra.

Para determinar la **bondad del ajuste de un modelo** necesitamos compararlo con un modelo nulo, el más simple de todos, la *media*, que es un "modelo de no relación entre variables". Cuantitativamente la determinamos usando la siguiente ecuación general:

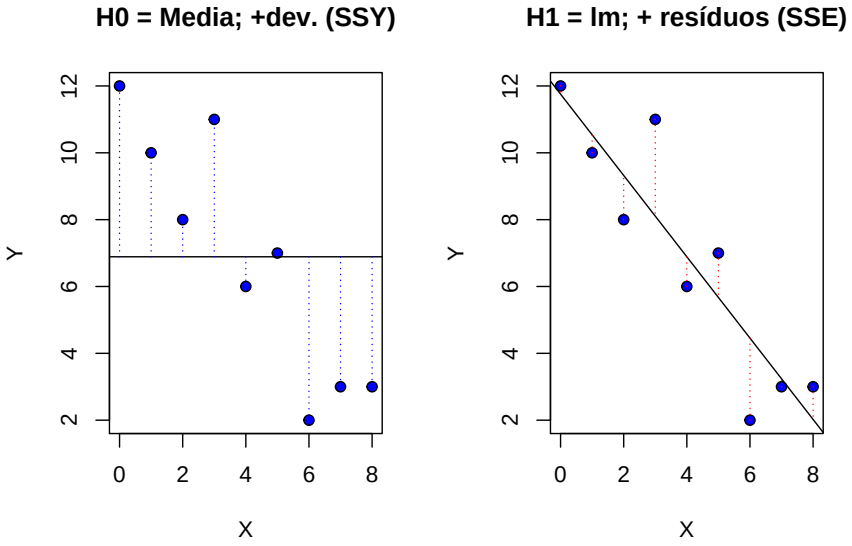
$$desviación = \sum (observación - modelo)^2$$

1.2.3.1 Explicación gráfica de los conceptos subyacentes a la bondad de ajuste: SSY, SSE y SSR

La mejor manera de explicar este concepto es hacerlo gráficamente:

```
opar <- par(no.readonly = TRUE)

par(mfrow = c(1,2))

## I. Gráfica de desviaciones de los datos de la media (para cálculo de SSY)
# 1. Graficamos los puntos (gráfico de dispersión)
plot(dfr1$X, dfr1$Y, pch=21, bg="blue", ylab= "Y", xlab="X",
     main = "H0 = Media; +dev. (SSY)")

# 2. Añadimos una recta, a la altura de la media de Y, como modelo nulo
abline(h=mean(dfr1$Y), col="black")

# 3. Con un bucle for y la función lines() podemos representar las desviaciones
#    de la media
for(i in 1:9)
```

```r
{
  lines( c(dfr1$X[i],dfr1$X[i]), c(mean(dfr1$Y), dfr1$Y[i]), col="blue", lty=3)
}

## II: Gráfica de resíduos (para cálculo de SSE)
# 2. Añadimos a la gráfica de dispersión la recta de regresión
plot(dfr1$X, dfr1$Y, pch=21, bg="blue", ylab= "Y", xlab="X",
     main = "H1 = lm; + resíduos (SSE)")

abline(lm(dfr1$Y ~ dfr1$X), col="black")

# Podemos obtener el valor predicho de Y para cada valor de X con predict()
fitted <- predict(lm(dfr1$Y ~ dfr1$X))


for(i in 1:9)
{
  lines( c(dfr1$X[i],dfr1$X[i]), c(dfr1$Y[i], fitted[i]), col="red", lty=3)
}
```



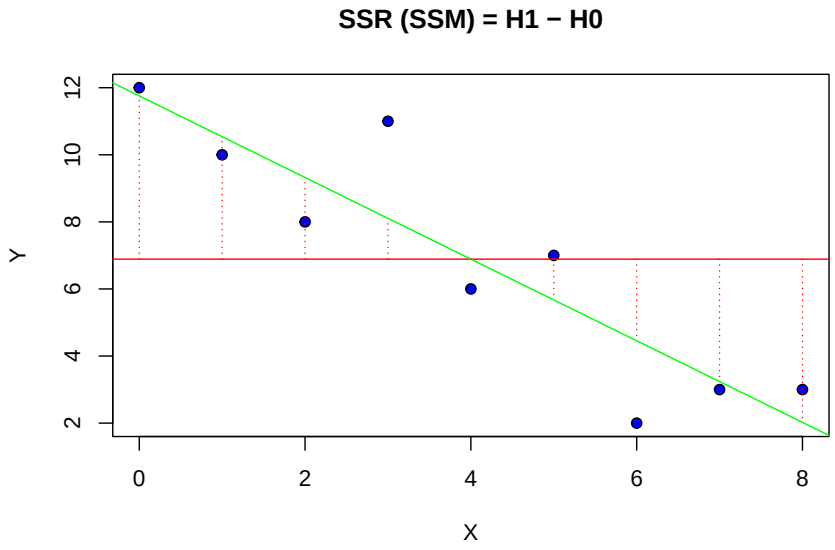

```r
par(opar)
```

- Definiciones
- SSY o SST es la **suma total de cuadrados**, ya que *representa la variación máxima o total*, respecto del modelo nulo (media)
- SSE es la **suma cuadrada de resíduos** y representa el *grado de error del modelo ajustado*, o *variación no explicada por el modelo*
- SSM o SSR es la **suma de cuadrados del modelo de regresión** y mide la *diferencia entre SSY y SSE*, indicando la ganancia en precisión de la estima al ajustar un modelo, con respecto a la media, o *variación explicada por el modelo*

Por tanto $SSY = SSR + SSE$ representa al particionado de la suma de cuadrados de una regresión.

La siguient gráfica muestra $SSR = SSY - SSE$ visualmente, es decir, las diferencias entre el valor promedio de Y y las estimas del modelo de regresión. Si SSR es grande, quiere decir que las predicciones hechas por el modelo son muy distintas a usar la media para predecir valores la variable de respuesta. Cuanto mayor sea SSR, mejor será la capacidad del modelo de predecir la variable de respuesta.

```r
## I. Gráfica de desviaciones de los datos de la media (para cálculo de SSR)
# 1. Graficamos los puntos
plot(dfr1$X, dfr1$Y, pch=21, bg="blue", ylab= "Y", xlab="X",
     main = "SSR (SSM) = H1 - H0")

# 2. Añadimos una recta, a la altura de la media de Y, como modelo nulo
abline(h=mean(dfr1$Y), col="red")

abline(lm(dfr1$Y ~ dfr1$X), col="green")
fitted <- predict(lm(dfr1$Y ~ dfr1$X))
# 3. Con un bucle for y la función lines() podemos representar las desviaciones
#    de la media
for(i in 1:9)
{
  lines( c(dfr1$X[i],dfr1$X[i]), c(mean(dfr1$Y), fitted[i]), col="red", lty=3)
}
```



1.2.3.2 Cálculo manual del particionado de las sumas de cuadrados en el análisis de regresión: análisis de la varianza

La idea que subyace a la fórmula $SSY = SSR + SSE$ es sencilla: tomamos la variación total de los datos originales (SSY) y la particionamos entre dos *componentes* que nos revelan el poder explicativo o predictor del modelo. La **variación explicada por el modelo** SSR se conoce como la *suma de cuadrados del modelo (de regresión)*, y la **variación no explicada** (SSE) se conoce como *suma de cuadrados de los errores*

Ya vimos cómo calcular SSY y SSE. Por tanto $SSR = SSY - SSE = \left(\sum y^2 - \frac{(\sum y)^2}{n}\right) - \left(\sum (y - a - bx)^2\right)$

Pero podemos hacerlo más sencillo, si en vez de tomar $SSE = \sum (y - a - bx)^2$, usamos el siguiente atajo:

$$SSR = b * SSXY$$

Podemos así calcular fácilmente SSR usando los valores de b y $SSXY$ previamente calculados:

$$SSR = -1.21667X - 73 = 88.81667$$

Teniendo SSY y SSR calculados, es trivial calcular SSE:

$$SSE = SSY - SSR = 108.8889 - 88.81667 = 20.07222$$

Ya tenemos los dos *componentes* (SSR y SSE) en los que se descompone la variación total de los datos (SSY). Hagamos un *análisis de la varianza*, la famosa **ANOVA**, para comparar estos dos componentes mediante un cociente F entre las varianzas asociadas a cada componente.

Estadísticos como F representan generalmente la cantidad de *varianza sistemática (del modelo) / varianza no sistemática (de los datos)*, es decir F se basa en la razón de SSR/SSE. Pero como las sumas de cuadrados dependen del número de puntos (diferencias sumadas), F usa la *media de la suma de cuadrados*, $MS = promedio\ de\ suma\ de\ cuadrados$.

$$F = \frac{MS_M}{MS_E} = \frac{media\ de\ los\ cuadrados\ del\ modelo}{media\ cuadrados\ de\ los\ errores\ o\ resíduos}$$

donde

$$MS_M = \frac{SSR}{g.l.} = \frac{SSR}{no.\ parámetros\ estimados\ en\ el\ modelo:\ b\ (1\ g.l.\ en\ ml\ simple)}$$

y

$$MS_E = \frac{SSE}{g.l.} = \frac{SSE}{no.\ observaciones - no.\ parámetros\ libres:\ a,\ b\ (n-2)}$$

Construyamos la **tabla de ANOVA** correspondiente, paso a paso:

Ya tenemos calculados las sumas de cuadrados de cada fuente de variación, como se muestra abajo.

Fuente	Suma de cuadrados	Grados de libertad	Media de cuadrados (Varianza)	cociente F
Regresión (modelo)	88.817 (SSR)			
Error (resíduos)	20.072 (SSE)			
Total (datos)	108.889 (SSY)			

Llenemos ahora la columna de los *grados de libartad* ($g.l.$). Recordemos que, esencialmente, los **grados de libartad** representan el número de 'entidades' que están libres de variar cuando estimamos algún parámetro estadístico a partir de los datos. Los $g.l$ generalmente se calculan como $n - número\ de\ parámetros\ libres\ a\ estimar\ de\ los\ datos$. Por tanto:

- $g.l.(SSY)$ Dado que $SSY = \sum (y - \hat{y})^2$, vemos que sólo necesitamos estimar **1** parámetro: $\hat{y}$; $g.l. = (n-1)$
- $g.l.(SSE)$ Dado que $SSE = \sum (y - a - bx)^2$, vemos que necesitamos estimar **2** parámetros: a y b; $g.l. = (n-2)$
- $g.l.(SSR)$ en el modelo de regresión lineal simple hemos estimado 1 parámetro b; $g.l. = 1$

Noten que la suma de $g.l.(SSR) + g.l.(SSE) = g.l.(SSY)$, como se ve en la tabla.

Fuente	Suma de cuadrados	Grados de libertad	Media de cuadrados (Varianza)	cociente F
Regresión (modelo)	88.817 (SSR)	1		
Error (resíduos)	20.072 (SSE)	7 $(n-2)$		
Total	108.889 (SSY)	8 $(n-1)$		

Necesitamos llenar la 4a. columna, reservada para las varianzas. Recoredemos que:

$$var = s^2 = \frac{suma\ de\ cuadrados}{grados\ de\ libertad}$$

Esta columna (4) es muy fácil de llenar, ya que tenemos los valors que necesitamos pre-calculados en las columnas 2 y 3. La *var total* no nos hace falta, pero sí el *cociente* $-$ $F = \frac{88.817}{2.86764}$, que incluímos tambien en la **tabla de ANOVA**, completándola.

Fuente	Suma de cuadrados	Grados de libertad	Media de cuadrados (Varianza)	cociente F
Regresión (modelo)	88.817 (SSR)	1	88.817 (var. regresión)	30.974
Error (resíduos)	20.072 (SSE)	7 ($n-2$)	$s^2 = 2.86764$ (var. error)	
Total	108.889 (SSY)	8 ($n-1$)		

Calculemos ahora la significancia del *cociente* $-$ F. Recordemos que:

- $H_0 : b = 0$, la pendiente de la recta de regresión es cero (el modelo no difiere de la media)
- $H_1 : b \neq 0$, usando una alternativa de doble cola asumimos que la pendiente es positiva o negativa

Podemos buscar el **valor crítico de** F en la tabla correspondiente, usando $g.l._{n}umerador = 1$ y $g.l._{d}enominador = 7$ y veríamos algo como ésto:

$g.l.(denom)$	p	**1** ... $(g.l.(num.))$
1	0.05	161.45
1	0.01	4052.18
...	...	...
7	0.05	**5.59**
7	0.01	**12.25**

Dado que nuestro estadístico F es mucho mayor que el valor crítico de $F_{(1,7)}$, rechazamos la hipótesis nula con $p < .001$.

Pero estamos *aprendiendo estadística usando R*, ¿verdad? ¿O cargas contigo las tablas de valores críticos para múltiples estadísticos?

Recordemos que los intervalos de confianza se definen como *quantiles* del 95% ó 99% ... Podemos calcular los quantiles para F usando la función $qf(.95, glNum, glDenom)$. La p asociada al estadístico F la calculamos como $1 - pf(F, glNum, glDenom)$, como se muestra seguidamente:

```
glNum <- 1
glDenom <- 7
Fstat <- 30.974

CI95_F <- qf(.95, glNum, glDenom)
CI99_F <- qf(.99, glNum, glDenom)

p <- 1-pf(Fstat,glNum,glDenom)

cat("CI95% = ", CI95_F, "| CI99% = ", CI99_F, " | p = ", p, "\n")
```

```
## CI95% =  5.591448 | CI99% =  12.24638  | p =  0.0008460725
```

Por tanto, la probabilidad de observar un $cociente - F$ de 30.974 bajo la H_0 es muy poco probable, de hecho $p < 0.001$, por lo que rechazamos H_0 contundentemente.

1.2.3.3 Cálculo en R del particionado de las sumas de cuadrados en el análisis de regresión: *summary.aov()*

Podemos fácilmente comprobar en R lo que hemos hecho a mano en la sección anterior, usando la función *summary.aov()* a la que le pasamos el objeto que regresa *lm()*:

```
# ajustemos un modelo lineal
fit1 <- lm(dfr1$Y ~ dfr1$X)

# summary.aov() nos da la tabla de ANOVA
summary.aov(fit1)
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## dfr1$X       1  88.82   88.82   30.97 0.000846 ***
## Residuals    7  20.07    2.87
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

1.2.3.4 Estima de la significancia de los coeficientes de regresión y cálculo de los errores asociados

Aunque el resultado del análisis de la varianza sea altamente significativo, no es particularmente interesante. En el contexto del *análisis de regresión* lo que realmente nos interesa conocer es el **tamaño de los efectos (estimas de a y b)**, su **significancia** y **errores** asociados a su estima.

- Los **errores** asociados a la estima **de los coeficientes** se calculan así:

$$SE_b = \sqrt{\frac{s^2}{SSX}} = \sqrt{\frac{2.867}{60}} = 0.2186$$

$$SE_a = \sqrt{\frac{s^2 \sum x^2}{n * SSX}} = \sqrt{\frac{2.867 * 204}{9 * 60}} = 1.0408$$

- Cálculo de la **significancia de b**

Ya vimos que en el peor de los escenarios $b = 0$, es decir, que un cambio unitario en la variable predictora no se asocia con cambio alguno en la variable de respuesta. Por tanto **si una variable predice significativamente una respuesta, su coeficiente de regresión tiene que ser significativamente distinto a 0**.

Esta hipótesis la podemos evaluar con una $prueba - t$, donde $H_0 : b = 0$.

Al igual que el $cociente - F$, el estadístico-t se basaen el cociente entre la varianza explicada y la no explicada o error. El test será significativo si b es grande comparado con la magnitud del error en la estima:

$$t = \frac{b_{observada} - b_{esperada}}{SE_b} = \frac{b_{observada} - 0}{SE_b} = \frac{-1.21667}{0.2186} = -5.565737$$

En regresión lineal simple, los **grados de libertad para los predictores (p)** se calculan como: $N - p - 1$, donde N tamaño de muestra, y p es el número de predictores (1 en este caso: b), es decir: $gl = N - 2$.

Podemos determinar la significancia del estadístico t con este código:

```
# 1. Podemos determinar los quantiles de 0.025 y 0.975, para 7 gl
qt(c(0.25, .975), df=7)
```

```
## [1] -0.7111418  2.3646243
```

```
# La significancia de t, para gl=7, asumiendo una distribución de doble cola
2*pt(-5.565737, df=7)
```

```
## [1] 0.0008457987
```

Por tanto, la estima del parámetro b es significativamente diferente de 0, rechazando la H_0 con $p < 0.001$.

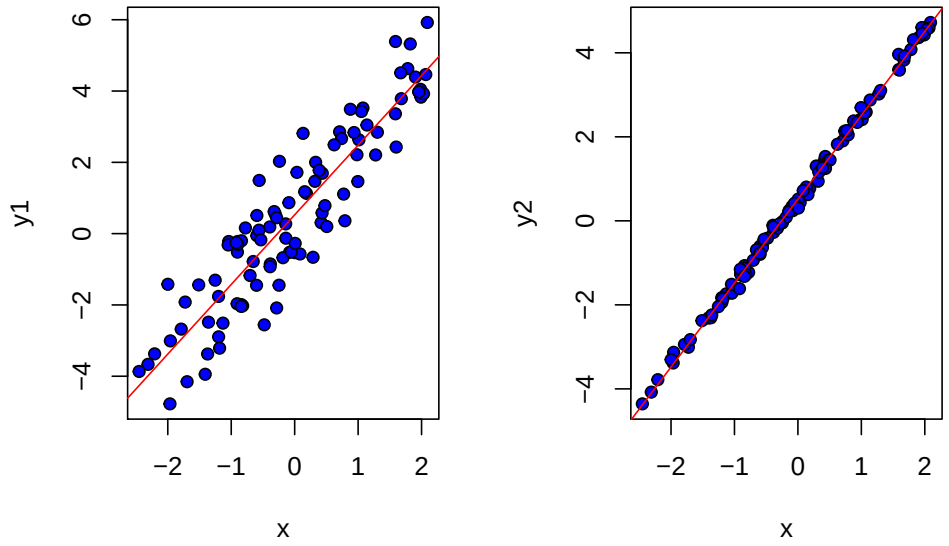
- De manera muy conveniente, toda esta **información sobre el modelo lineal** nos la da la función *summary*():

```
fit1 <- lm(dfr1$Y ~ dfr1$X)

# la salida de summary() nos da los tamaños de los efectos, es decir, los
# coeficientes de regresión (a, b), los errores asociados, su significancia (t-test) +
# el resultado de la prueba F (ANOVA)
summary(fit1)
```

```
##
## Call:
## lm(formula = dfr1$Y ~ dfr1$X)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.4556 -0.8889 -0.2389  0.9778  2.8944
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  11.7556     1.0408  11.295 9.54e-06 ***
## dfr1$X       -1.2167     0.2186  -5.565 0.000846 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.693 on 7 degrees of freedom
## Multiple R-squared:  0.8157, Adjusted R-squared:  0.7893
## F-statistic: 30.97 on 1 and 7 DF,  p-value: 0.0008461
```

1.2.3.5 Cálculo del grado de ajuste, R^2

Nos queda un asunto muy importante que considerar. Dos rectas de regresión pueden tener la misma pendiente y puntos de corte, pero derivarse de relaciones completamente diferentes entre las variables:

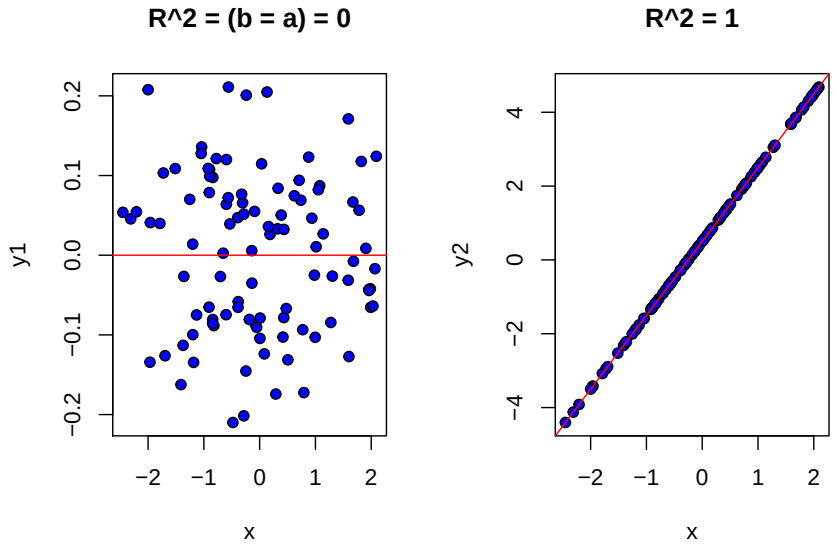



```
## Coeficientes para fit1:  0.5 2 Coeficientes para fit2:  0.5 2
```

Claramente el ajuste del modelo a los datos en la gráfica derecha es mucho mejor que en la izquierda. Llevado al extremo ideal, los puntos caerían todos sobre la recta, reflejando una *dispersión* $= 0$. En el otro extremo, x podría no explicar ninguna de la variación observada en y. En dicho caso las variables no estarían correlacionadas, el ajuste del modelo lineal sería 0 y la magnitud de la dispersión sería de 100%.

Combinando lo que hemos aprendido sobre SSY, SSR y SSE debería quedar claro que podemos derivar una métrica que mida *la fracción de la variación total en y explicada por la regresión.* A esta medida es a la que llamamos **coeficiente de determinación:** $R^2 = \frac{SSR}{SSY}$.

R^2 varía entre 0, cuando la regresión no explica ninguna de la variación obserbada en y ($SSE = SSY$; dispersión = 100%), y 1, cuando la regresión explica toda la variación en y ($SSR = SSY$; dispersión = 0%). Gráficamente:




```
## * Coeficientes para fit1:  0 0
## * Coeficientes para fit2:  0.5 2
```

Esta R^2 corresponde al *coeficiente de determinación* que vimos en el tema de correlación. Generalmente lo expresaremos como porcentaje ($R^2 * 100$), y **representa la cantidad de varianza en la variable de respuesta explicada por el modelo (SSR) relativa al total de la variación existente en los datos**

(SSY **o** SST). Por tanto, R^2 representa el **porcentaje de la variación en la variable de respuesta que puede ser explicada por el modelo**.

Además, tratándose de regresión simple, podemos calcular el *coeficiente de correlación de Pearson* entre las dos variables como $r = \sqrt{R^2}$.

1.3 Diagnóstico del modelo de regresión y validación de supuestos

Un último aspecto central en el *análisis de regresión lineal* es comprobar que **el modelo no viole los supuestos de los que depende**.

1.3.1 Supuestos

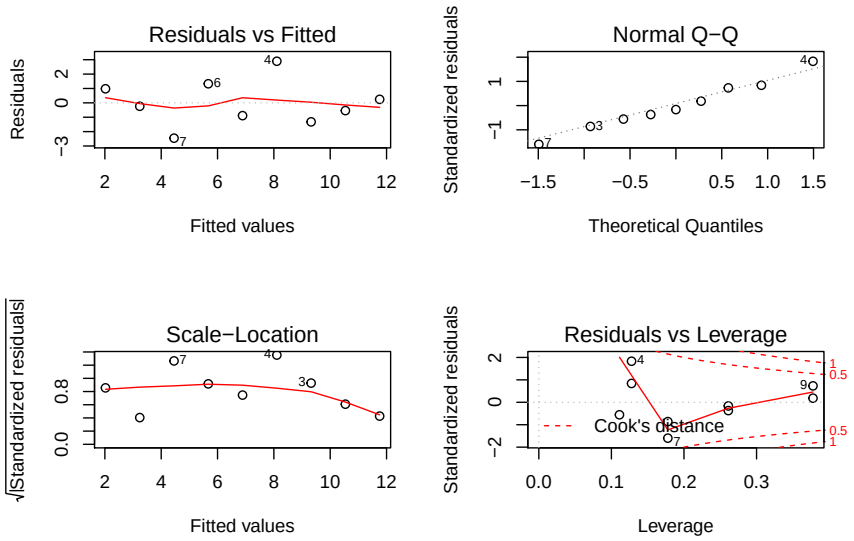
Particularmente críticos son:

1. **linearidad** la variable de respuesta está relacionada linealmente con las variables predictoras
2. **constancia de las varianzas (homocedasticidad)** : la varianza de la variable de respuesta no varía con los niveles de la predictora
3. **normalidad de los errores**
4. **ausencia de puntos marcadamente aberrantes (outliers) palanca o influyentes**

1.3.2 Diagnóstico gráfico del modelo

Una primera opción para comprobar si el modelo cumple estos supuestos es mediante la generación de cuatro *gráficas de diagnósitico del modelo* a partir de la información que viene integrada en el objeto que regresa la función *lm*() usando la función *plot*()

```
opar <- par(no.readonly = TRUE)

# ajustamos modelo lineal
modelo <- lm(dfr1$Y ~ dfr1$X)

# imprimimos las gráficas en un arreglo de 2x2
par(mfrow = c(2,2))
plot(modelo)
```



```r
par(opar)
```

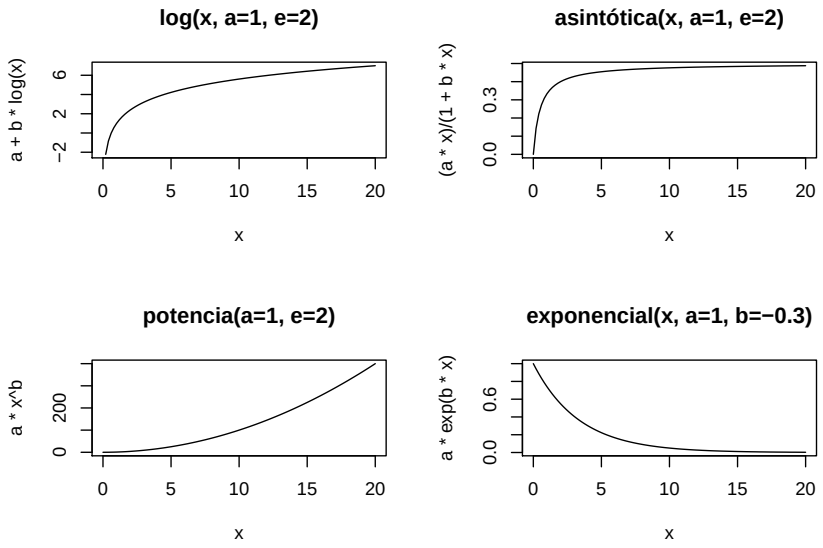
- La primera gráfica (arriba izda.) nos ayuda a decidir si las *variables están linealmente relacionadas*. Si es así, no debería de existir una relación sistemática entre los resíduos (errores) y los valores predichos (ajustados). Es decir, **el modelo debería de capturar toda la varianza sistemática de los datos, dejando sólo ruido estocástico sin explicar**. Por tanto, esta gráfica debe de verse como "las estrellas en el firmamento", sin un patrón claro de asociación. Si es así, sugiere además que se cumple el *supuesto de homocedasticidad*.
- La segunda es la **gráfica de quantil-quantil normal**. Los puntos deberían seguir la diagonal si los resíduos están normalmente distribuidos. Si aparecen patrones tipo "S" o "banana", posiblemente necesitemos ajustar otro modelo.
- La tercera gráfica es como la primera, pero usando una escala diferente, $\sqrt{|residuos\ estandarizados|}$, y sirve para comprobar la **homocedasticidad**, la cual se cumple si los puntos forman una *banda estocástica en torno a la horizontal*.
- La última gráfica (abajo, dcha.) trata sobre la **identificación de puntos influyentes, aberrantes y con efecto palanca**. Vemos que la observación 4 es la que tiene el mayor resíduo estandarizado, la 7 tiena la mayor $Cook - D$ y la nueve al mayor efecto palanca.
  - *observaciones influyentes*: son aquellas con un impacto desproporcionado en la determinación de los parámetros del modelo. Se identifican usando la distancia de Cook, $Cooks - D$. Son de preocupar los puntos con $Cooks - D > 1$
  - *punto aberrante*: es una observación que no es predicha satisfactoriamente por el modelo de regresión (con un valor positivo o negativo grande del resíduo)
  - observación con alto *efecto palanca* tienen una combinación inusual de valores predictores. Es decir, son aberrantes en el espacio predictor.

Las dos gráficas de la primera fila son las más importantes.

1.4 Transformación de datos y selección de modelos

Conviene recordar que $y = a + bx$ no es el único *modelo de dos parámetros* que existe para describir la relación entre dos variables contínuas. Tenemos que tener ésto en cuenta, ya que la **selección del modelo** adecuado es un *aspecto de importancia crítica en el análisis de regresión*. Es decir, si los datos no presentan una relación lineal, podemos encontrar *modelos curvilíneos* con igual grado de parametrización que la recta, los cuales se ajustan mejor a *datos con curvatura*.

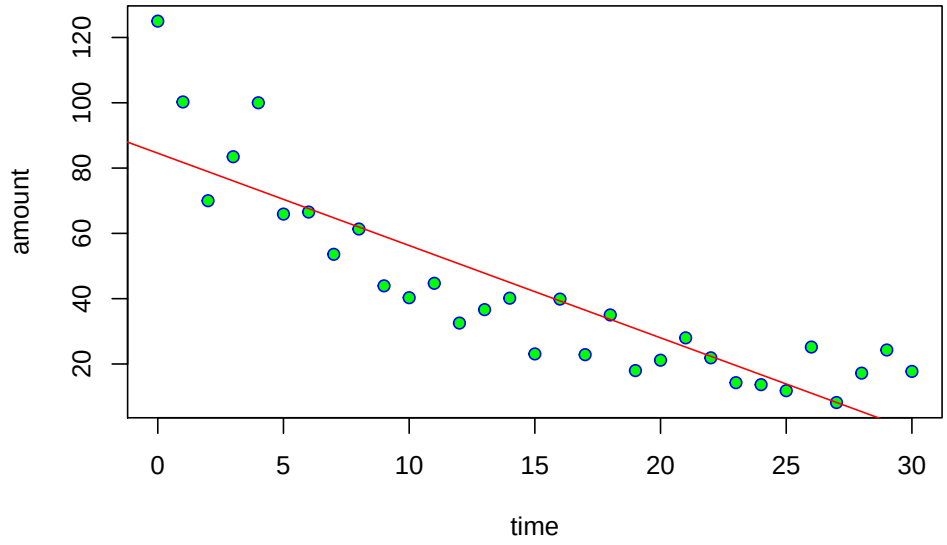
Veamos algunos de estos modelos no lineales de 2 parámetros. Nota: en todos los casos se ha llamado a la función $curve(f, 0, 20)$ para graficar las funciones (f) indicadas sobre los ejes de ordenadas, con los coeficienes a y b indicados en los títulos.



Es sencillo **estimar los parámetros de estos modelos**, si *las funciones se pueden transformar de tal manera que sus parámetros se conviertan en lineales.*

1.4.1 Linearización de un modelo exponencial y evaluación del ajuste

Veamos un ejemplo de datos del *decaimiento de emisiones radioactivas en función del tiempo.* Vamos a graficarlos y a ajustar una recta de mínmos cuadrados a los datos.



Vemos que el patrón de dispersión presenta una marcada **curvatura**. La mayoría de los puntos para $t < 5$ y $t > 25$ tienen resíduos positivos, mientras que para los valores intermedios son mayormente negativos. Por tanto el *modelo lineal NO es adecuado para estos datos.*

Veamos el resumen de las estadísticas asociadas al modelo lineal simple:

```
##
## Call:
## lm(formula = amount ~ time)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -19.065 -10.029  -2.058   5.107  40.447
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  84.5534     5.0277   16.82  < 2e-16 ***
## time         -2.8272     0.2879   -9.82 9.94e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 14.34 on 29 degrees of freedom
## Multiple R-squared:  0.7688, Adjusted R-squared:  0.7608
## F-statistic: 96.44 on 1 and 29 DF,  p-value: 9.939e-11
```

- Podemos ver que los coeficientes son altamente significativos, y que el modelo explica más de 76% de la variación en la variable de respuesta (un valor muy alto de R^2).
- La moraleja es que *valores* $-$ p y R^2 nos *hablan del ajuste*, pero **no indican si el tipo of familia de modelo ajustado es el más adecuado**.

Dado que los datos son de **decaimiento radiactivo**, parece razonable pensar que una **función exponencial negativa** $y = ae^{-bx}$ pueda ajustarse mejor a los datos.

Probemos esta hipótesis, **linearizando la ecuación exponencial**, tomando logaritmo a ambos lados de la igualdad y operando así:

$$y = ae^{-bx}$$

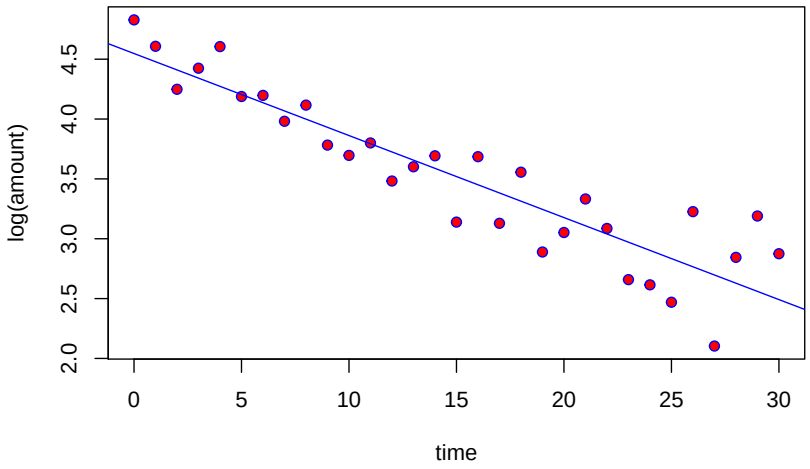
$$log(y) = log(a) - bx$$

Cambiemos $log(y)$ por Y y $log(a)$ por A, que representa un *modelo lineal*, con un punto de corte A y una pendiente b

$$Y = A - bx$$

Grafiquemos el **modelo linearizado**:

```
# transformamos solo la variable de respuesta: log(y)
plot(time,log(amount),pch=21,col="blue",bg="red")
abline(lm(log(amount)~time),col="blue")
```



El ajuste a los datos se ha mejorado de manera apreciable. Veamos el resumen estadístico para el nuevo modelo:

```
summary(lm(log(amount)~time))
```

```
##
## Call:
## lm(formula = log(amount) ~ time)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.5935 -0.2043  0.0067  0.2198  0.6297
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.547386   0.100295   45.34  < 2e-16 ***
## time        -0.068528   0.005743  -11.93 1.04e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.286 on 29 degrees of freedom
## Multiple R-squared:  0.8308, Adjusted R-squared:  0.825
## F-statistic: 142.4 on 1 and 29 DF,  p-value: 1.038e-12
```

Vemos que **el error estándar de los resíduos ha decrecido mucho, y que R^2 es mayor aún**, al igual que el nivel de significancia de los coeficientes. El *ajuste a los datos* es sin duda mejor.

Podemos formalmente **seleccionar el mejor de ambos modelos** usando el *criterio de informacón de Akaike* (**AIC**), que indica claramente que el modelo linearizado es mucho más adecuado.

```
fit1 <- lm(amount ~ time)
fit.log <- lm(log(amount) ~ time)

# elegiremos el modelo con el valor de AIC más pequeño (AICmin),
# despreciando todos los modelos con > 3 unidades AIC con respecto a AICmin
AIC(fit1, fit.log)
```

```
##         df       AIC
## fit1     3 257.00156
## fit.log  3  14.29588
```

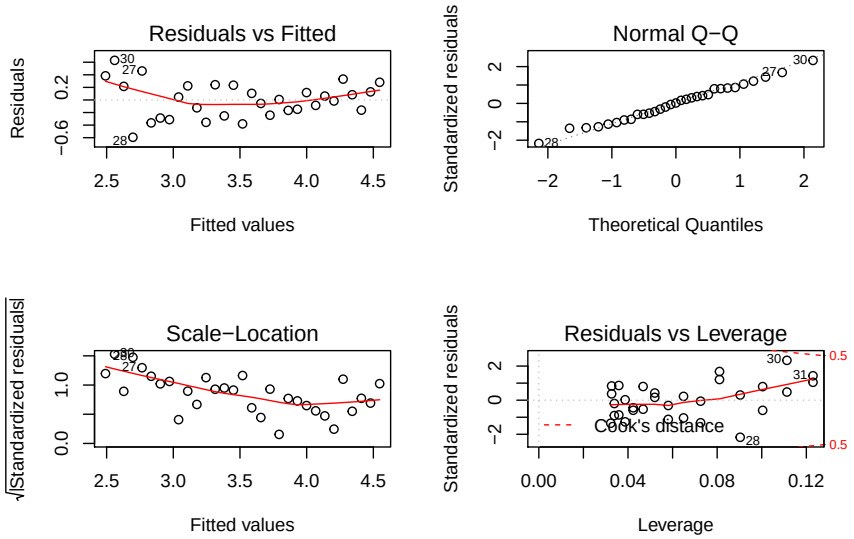
Si les gustan más los $p-valores$, podemos confirmar el resultado de $AIC()$ comparando ambos modelos mediante la función $anova(lm_0, lm_1)$

```
anova(fit1, fit.log)
```

```
## Analysis of Variance Table
##
## Response: amount
##           Df  Sum Sq Mean Sq F value    Pr(>F)
## time       1 19822.5 19822.5  96.441 9.939e-11 ***
## Residuals 29  5960.6   205.5
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Está claro por tanto que fue una decisión crítica elegir el **modelo linearizado**.

1.4.1.1 Diagnóstico del modelo linearizado



Parece no haber problemas con el *supuesto de normalidad*, pero sí parece hablerlos con respecto al *supuesto de homocedasticidad*, como muestra la forma en punta de flecha de la primera gráfica y la caída que muestran los resíduos estandarizados en función de los valores ajustados (3a. gráfica). La última gráfica muestra que los puntos 30 y 31 tienen alto *valor de palanca* ("leverage"), mientras que el 28 tiene un valor muy alto del *resíduo* asociado.

1.4.2 Ajuste de una curva exponencial negativa a datos de decaimiento radioactivo

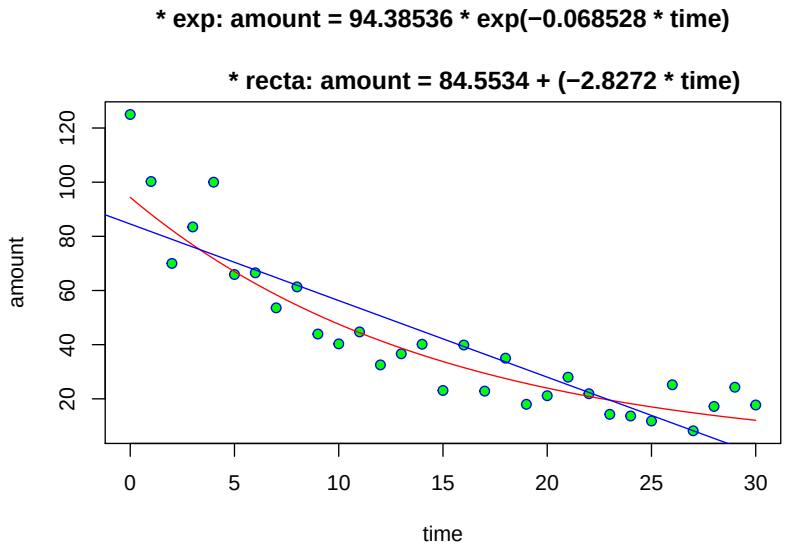
A pesar de los problemas de *heterocedasticidad*, *puntos aberrantes* y *observaciones palanca*, ajustemos la *curva exponencial negativa* a los datos no transformados de decaimiento radioactivo. Para ello necesitamos primero los coeficienes para sustituir en la función exponencial $y = ae^{bx}$. Recordemos que el modelo linearizado nos daba los coeficienes: $A = 4.547386$ y $b = -0.068528$. Necesitamos regresar A a su escala original, tomando su *antilogaritmo* usando $exp()$, es decir: $a = exp(A)$

```
exp(coef(fit.log)[1]) # ésto es exp(A)
```

```
## (Intercept)
##    94.38539
```

Además debemos saber que R grafica curvas como múltiples segmentos rectilíneos. Necesitamos > 100 de estos segmentos en el ancho de una gráfica para que la curva se vea razonablemente "lisa". Viendo el gráfico de dispersión original (no transformado), vemos que el *rango para* x es: $0 \leq x \leq 30$. Por tanto, para tener $>$ 100 segmentos queremos > 3 segmentos por unidad de tiempo, es decir, está bien usar intervalos de 0.25. Usaremos la variable xv para almacenar los "valores de x": $xv < -seq(0, 30, 0.25)$, es decir, tendremos $length(xv)$ segmentos. Veamos el código:

```
plot(time, amount, pch = 21, col = "blue", bg = "green",
     main ="* exp: amount = 94.38536 * exp(-0.068528 * time)\n
            * recta: amount = 84.5534 + (-2.8272 * time)")
xv <- seq(0,30,0.25)
yv <- 94.38536 * exp(-0.068528 * xv)
lines(xv, yv, col="red")
abline(fit1, col="blue")
```



Sin duda el ajuste de $amount = 94.38536 * exp(-0.068528 * time)$ a los datos es mucho mejor que el que muestra la recta de mínimos cuadrados: $amount = coef(fit1[1]) + coef(fit1[2]) * x) = 84.5534 + (-2.8272 * time)$. Pero todavía queda trabajo por hacer. El ajuste es bueno sólo para los valores intermedios de *time*, siendo peor cuando $0 \leq time \leq 29$.

1.5 Regresión polinomial y determinación de desviaciones de relación de linearidad entre variables

La dependencia entre la *variable de respuesta* y la *regresora* frecuentemente no es lineal. No obstante, **el principio de parsimonia** (*navaja de Occam*) requiere que ajustemos un **modelo lineal simple (el más simple, menos parametrizado) como modelo nulo**, a no ser que un modelo no lineal se demuestre significativamente superior al nulo.

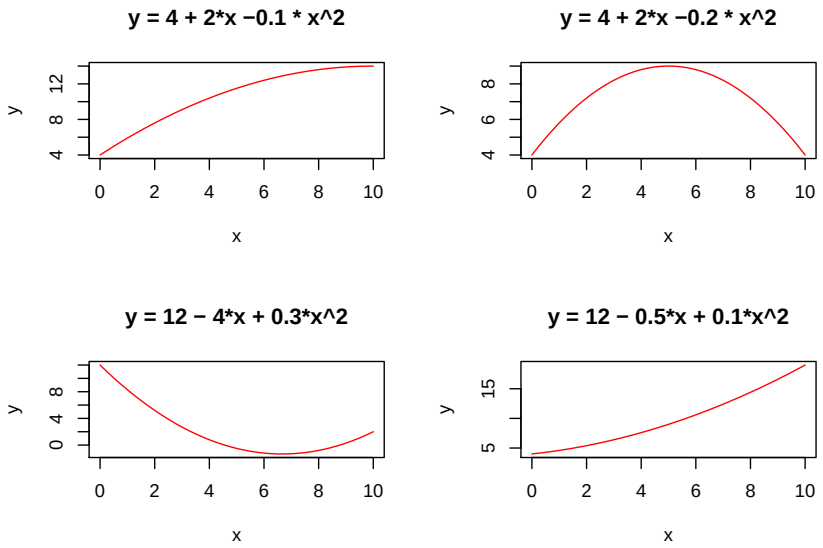
¿Cómo determinar la significancia de la desviación del supuesto de linearidad?

Una de las maneras más sencillas es usando la **regresión polinomial**, donde:

$$y = a + bx + cx^2 + dx^3 ...$$

El concepto básico es sencillo. Tenemos una sola variable explicativa contínua, x, pero podemos *ajustar potencias mayores de x, como* x^2, x^3 ... y añadirlas al modelo, junto a x, para describir diversos tipos de **curvatura** en la relación y x. Las **funciones polinomiales presentan gran flexibilidad de formas**, incluso al añadir un solo término cuadrático, dependiendo de los *signos* de los *términos lineales y cuadrático*, como se muestra seguidamente:

```
opar <- par(no.readonly = TRUE)
par(mfrow=c(2,2))
curve(4+2*x-0.1*x^2,0,10,col="red",ylab="y", main = "y = 4 + 2*x -0.1 * x^2")
curve(4+2*x-0.2*x^2,0,10,col="red",ylab="y", main = "y = 4 + 2*x -0.2 * x^2")
curve(12-4*x+0.3*x^2,0,10,col="red",ylab="y", main = "y = 12 - 4*x + 0.3*x^2")
curve(4+0.5*x+0.1*x^2,0,10,col="red",ylab="y", main = "y = 12 - 0.5*x + 0.1*x^2")
```




```
par(opar)
```

Vemos que un simple *modelo cuadrático* con sólo tres parámetros (*punto de corte, pendiente para* x *y pendiente para* x^2) puede describir un amplio rango de relaciones funcionales $y \sim x$.

Veamos cómo se ajusta un modelo cuadrático de 3 parámetros (*model*3) y cómo se compara con uno lineal de dos parámetros (*model*2).

```
model2 <- lm(amount~time)
model3 <- lm(amount~time+I(time^2))
model3
```

```
##
## Call:
## lm(formula = amount ~ time + I(time^2))
##
## Coefficients:
## (Intercept)         time    I(time^2)
##    106.3888      -7.3448       0.1506
```

```
summary(model3)
```

```
##
## Call:
## lm(formula = amount ~ time + I(time^2))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -22.302  -6.044  -1.603   4.224  20.581
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 106.38880    4.65627  22.849  < 2e-16 ***
## time         -7.34485    0.71844 -10.223 5.90e-11 ***
## I(time^2)     0.15059    0.02314   6.507 4.73e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.205 on 28 degrees of freedom
```

```
## Multiple R-squared:  0.908,  Adjusted R-squared:  0.9014
## F-statistic: 138.1 on 2 and 28 DF,  p-value: 3.122e-15
```

```
AIC(model2,model3)
```

```
##        df      AIC
## model2  3 257.0016
## model3  4 230.4445
```

```
anova(model2,model3)
```

```
## Analysis of Variance Table
##
## Model 1: amount ~ time
## Model 2: amount ~ time + I(time^2)
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1     29 5960.6
## 2     28 2372.6  1    3588.1 42.344 4.727e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Vemos que la *pendiente para el término cuadrático* (0.15059) es altamente significativa, indicando la existencia de una *fuerte curvatura* en los datos. Por tanto el AIC y el test de $ANOVA$ indican una clara superioridad del modelo cuadrático sobre el lineal, rechazándolo.

Finalmente comparemos los tres **modelos en competición**: *lineal, cuadrático y exponencial linearizado* que hemos ajustado a los datos de decaimiento, para ver cuál de ellos es el mejor de los que hemos evaluado hasta ahora:

```
model2 <- lm(amount~time)
model2.lin <- lm(log(amount)~time)

# nótese uso de I(): "as is"; permite pasar operadores aritméticos en fórmulas
model3 <- lm(amount~time+I(time^2))
AIC(model2, model2.lin, model3)
```

```
##            df       AIC
## model2      3 257.00156
## model2.lin  3  14.29588
## model3      4 230.44447
```

```
#anova(model2, model2.lin, model3)
```

- *Seleccionamos* definitivamente el *modelo exponencial linearizado model2.lin*.
- Usaremos éste para hacer **predicciones**.

1.6 *AIC*: Selección de modelos mediante el criterio de información de Akaike

El criterio de información de Akaike es un *estimador no sesgado del parámetro de contenido de información de Kullback-Leibler*, el cual es una **medida de la información perdida al usar un modelo para aproximar la realidad**.

Por tanto, **a menor valor de AIC mejor ajuste del modelo a los datos**. Al penalizar por cada parámetro adicional, considera tanto la bondad de ajuste como la varianza asociada a la estima de los parámetros.

Hay que recordar que **añadir parámetros a un modelo siempre contribuirá a explicar algo más de la variación estocástica o no explicada**. Por tanto, a mayor parametrización, mejor ajuste. De hecho

los **modelos saturados**, con un parámetro por cada dato puntual, presentan un *ajuste perfecto* a los datos. El problema es que *carecen de cualquier fuerza explicadora.*

Por tanto hay que buscar siempre el **balance óptimo entre el grado de parametrización (ajuste) del modelo e incremento asociado en la varianza de las predicciones**, resultado de estimar más parámetros del mismo conjunto finito de datos. Tenemos que evaluar críticamente el aporte que cada nuevo parámetro hace a la fuerza explicativa del modelo. Muchos parámetros presentan alta *colinearidad* entre ellos, no aprotando fuerza explicativa al modelo, pero sí inflando su error.

1.6.1 Fórmulas para cálculo del AIC_i para un $modelo_i$

- El AIC_i para un $modelo_i$ en particular se calcula así:

$$AIC_i = -2lnLi + 2 * (N_i + 1)$$

- $N_i = no.\ de\ parámetros\ libres\ en\ el\ modelo_i$
- $L_i = verosimilitud\ bajo\ el\ modelo_i$

También se puede describir el AIC como:

$$AICi = deviancia_i + 2 * N_i$$

Cuando la $deviancia < 2$ entre un $modelo_0$ y otro alternativo $modelo_1$ con un parámetro adicional, no se justifica la inclusión del mismo, ya que infla más la varianza en la estima que lo que aporta en fuerza explicativa.

1.6.1.1 AIC: ejemplos numéricos

$$AIC_i = -2 * ln(L_i) + 2 * (N_i + 1)$$

- Por ejemplo, para $model2$, calculamos su AIC así: $AIC_{model2} = -2 * logLik(model2) + 2 * (2 + 1)$

```
AIC_model2 = -2 * logLik(model2) + 2 * (2 + 1)
AIC_model2
```

```
## 'log Lik.' 257.0016 (df=3)
```

- Calculemos el AIC de model2, model2.lin y model3:

```
AIC_model2 = -2 * logLik(model2) + 2 * (2 + 1)
AIC_model2.lin = -2 * logLik(model2.lin) + 2 * (2 + 1)
AIC_model3 = -2 * logLik(model3) + 2 * (3 + 1)

cat(" * AIC_model2 = ", AIC_model2, "\n", "* AIC_model2.lin = ", AIC_model2.lin, "\n",
    "* AIC_model3 = ", AIC_model3, "\n")
```

```
##  * AIC_model2 =  257.0016
##  * AIC_model2.lin =  14.29588
##  * AIC_model3 =  230.4445
```

1.6.2 Estadísticos de diferencias en AIC (ΔAIC_i) y ponderaciones de Akaike

Se pueden usar los estadísticos de **diferencias en AIC** (ΔAIC_i) y **ponderaciones de Akaike** para cuantificar el nivel de incertidumbre en la selección del modelo.

Las ΔAIC_i son $AICs$ re-escalados con respecto modelo con el AIC más bajo ($AICmin$), de modo que:

$$\Delta AIC_i = AIC_i - AICmin$$

Las ΔAIC_i son fáciles de interpretar y permiten ordenar los modelos candidatos:

- rango de ΔAIC_i entre modelo $AICmin$ y competidores
- 1-2 con respecto al modelo ganador tienen un soporte sustancial y deben de ser considerados como modelos alternativos.
- 3-7 con respecto al modelo ganador tienen un soporte significativamente inferior
- 10+ carecen de soporte

1.7 Predicciones usando el modelo seleccionado

Una vez *seleccionado el mejor modelo* y habiéndolo *ajustado a los datos*, típicamente querremos usarlo para **hacer predicciones** de la variable de respuesta a valores de nuestro interés de la variable regresora. Existen dos tipos de predicciones, sujetas a niveles muy diferentes de incertidumbre:

- Las **interpolaciones** son predicciones hechas *dentro* del rango de los datos usados para construir el modelo
- Las **extrapolaciones** son predicciones hechas fuera de este rango y son mucho más problemáticas, siendo la selección del modelo aún más crítica y difrentes modelos frecuentemente generan predicciones dispares.

La fución genérica $predict()$ del paquete de base *stats* para hacer predicciones a partir de los objetos obtenidos de aplicar diversas funciones de ajuste de modelos, como $lm()$, y que llamaremos genéricamente $objeto.modelo$.

La sintaxis básica es:

$predict(objeto.modelo, ...)$

Si lo llamamos así, sin argumentos opcionales, nos regresa los valores ajustados como vimos en ejemplos anteriores.

Si queremos calcular los valores predichos por el modelo para nuevos valores de la variable predictora, le pasamos una lista o dataframe como argumento a $newdata =$.

$predict(objeto.modelo, newdata = data.frame(varPredictora = c(x1, x2...)))$ Es importante notar que el nombre de $varPredictora$ debe ser el de la que hemos usado para construir el modelo ($time$ en el caso que nos ocupa)

Además de las estimas puntuales, podemos pedirle a $predict()$ que regrese también el **intervalos de predicción para la respuesta** del nivel deseado ($level = .95$ por defecto), pasándole el valor *"prediction"* al parámetro $interval$: $interval =$ "$prediction$". Este **intervalo de la predicción es el rango de la distribución de la respuesta predicha**. Si le pasamos el valor *"interval"*, obtendremos los **intervalos de confianza para la respuesta media**.

$predict(objeto.modelo, newdata = data.frame(varPredictora = c(x1, x2...)), interval =$ "$prediction$", $level = .99)$

Esto quedará más claro con los ejemplos de código que se muestran seguidamente, para estimar los valores de la variable de respuesta *"amount"* de radiación en función de valores de la variable regresora *"time"* en los que estamos interesados.

1.7.1 Interpolaciones

```r
# los valores de interés de la var. regresora se pasan como un vector con el nombre de
# la variable regresora, almacenados en una lista o dataframe
# En este caso los valores de x están dentro del rango de las observaciones usadas
# para construir el modelo: vamos a "interpolar"
preds <- data.frame(time=c(3, 12.5, 22.2, 29))

# La función predict() nos calcula los valores ajustados al modelo.
# Podemos pedirle que regrese sólo las predicciones puntuales, pero si queremos
# también el intervalo de confiaza del 95% pasamos el valor "prediction" al
# parámetro interval para obtenerlo.
prediction.interpol.pi <- predict(model2.lin, newdata=preds, interval="prediction",
                                  level=.95)
str(prediction.interpol.pi)
```

```
##  num [1:4, 1:3] 4.34 3.69 3.03 2.56 3.73 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:4] "1" "2" "3" "4"
##   ..$ : chr [1:3] "fit" "lwr" "upr"
```

```r
# Si cambiamos interval="confidence", obtendremos los intervalos de confianza para
# la respuesta media estimada de radiación (amount)
prediction.interpol.ci <- predict(model2.lin, newdata=preds, interval="confidence",
                                  level=.95)

# recordemos que en model2.lin, tomamos log(amount); de ahí el exp()
# noten como el valor del ci en torno a la media es más estrecho que el ci
# para las predicciones de observaciones.
# Noten también cómo el rango para *.ci son mucho más estrechos que para
# las predicciones *.pi
exp(prediction.interpol.ci)
```

```
##        fit      lwr      upr
## 1 76.84581 64.45778 91.61468
## 2 40.07612 35.93455 44.69503
## 3 20.61574 18.01475 23.59226
## 4 12.93661 10.64330 15.72406
```

```r
exp(prediction.interpol.pi)
```

```
##        fit       lwr       upr
## 1 76.84581 41.722272 141.53781
## 2 40.07612 22.104370  72.65965
## 3 20.61574 11.311133  37.57437
## 4 12.93661  6.982805  23.96686
```

1.7.2 extrapolaciones

```r
# estos valores de la variable regersora time están fuera del rango de las observaciones.
# por tanto estamos EXTRAPOLANDO:
preds <- data.frame(time=c(35, 50,  65, 95, 125))
prediction.extrapol.ci <- predict(model2.lin, newdata=preds, interval="confidence",
                                  level=.95)
prediction.extrapol.pi <- predict(model2.lin, newdata=preds, interval="prediction",
```

```
                                level=.95)
exp(prediction.extrapol.ci)
```

```
##          fit         lwr        upr
## 1 8.57534033 6.629701032 11.0919725
## 2 3.06781008 2.007038793  4.6892261
## 3 1.09750265 0.604375227  1.9929871
## 4 0.14046230 0.054567845  0.3615620
## 5 0.01797687 0.004917582  0.0657168
```

```
exp(prediction.extrapol.pi)
```

```
##          fit         lwr         upr
## 1 8.57534033 4.526126493 16.24710709
## 2 3.06781008 1.489370114  6.31908658
## 3 1.09750265 0.475940138  2.53080579
## 4 0.14046230 0.046207508  0.42697951
## 5 0.01797687 0.004336027  0.07453083
```

1.7.3 Graficado de bandas e intervalos de confianza

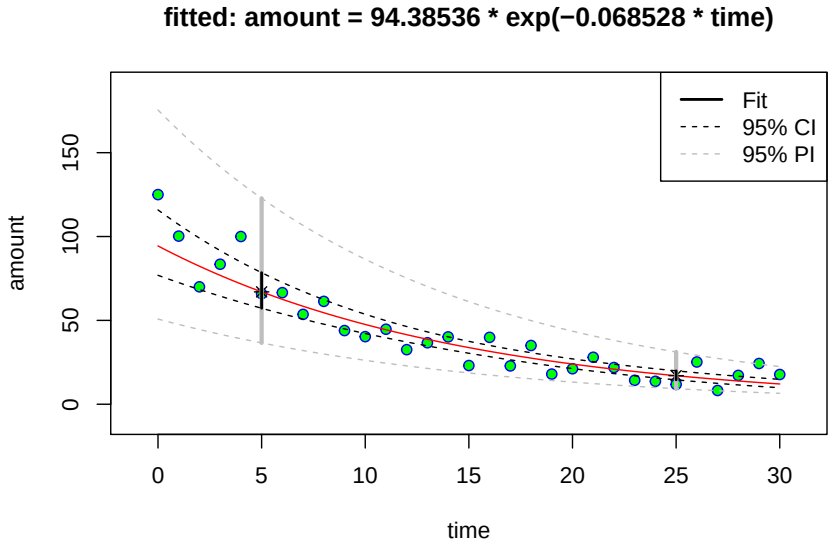
Para concluir este capítulo de introducción a la regresión, veamos cómo graficar los intervalos de confianza asociados al modelo de regresión.

El siguiente código muestra como graficar **intervalos de confianza para la respuesta media** (ci) e **intervalos de predicción para la respuesta** (pi).

Graficaremos tanto **intervalos** de ci como de pi entorno a dos puntos selectos, como **bandas** *de CIs y PIs* en torno a las predicciones correspondientes para un rango de valores de x.

```
# I. Graficado de intervalos de credibilidad en torno a la media (ci) y a las
#    predicciones (pi) para para dos valors específicos de "time"
xvals <- data.frame(time=c(5, 25))

# Recuerden: hemos ajustado un modelo exponencial linearizado, por tanto tenemos
# que tomar el antilogaritmo para regresar los valores de y estimados a  su
# escala original, para añadirlos sobre el gráfico de dispersión
mypred.ci <- exp(predict(model2.lin, newdata=xvals, interval="confidence", level=.95))
mypred.pi <- exp(predict(model2.lin, newdata=xvals, interval="prediction", level=.95))

# Generamos el gráfico de dispersión y añadimos la curva corresondiente al ajuste
# exponencial negativo, dando un poco más de márgenes al rango de x e y con xlim e ylim
plot(time, amount, pch = 21, col = "blue", bg = "green",
     main = "fitted: amount = 94.38536 * exp(-0.068528 * time)",
     xlim = c(-1, 31), ylim = c(-10, 190))
xv <- seq(0,30,0.25)
yv <- 94.38536 * exp(-0.068528 * xv)
lines(xv, yv, col="red")

# añadimos los puntos de la predicción y barras en negro y gris para los cis y pis
points(xvals[,1],mypred.ci[,1],pch=8)
segments(x0=c(5,25), x1=c(5,25), y0=c(mypred.pi[1,2],mypred.pi[2,2]),
         y1=c(mypred.pi[1,3],mypred.pi[2,3]),col="gray",lwd=3)
segments(x0=c(5,25), x1=c(5,25), y0=c(mypred.ci[1,2],mypred.ci[2,2]),
         y1=c(mypred.ci[1,3],mypred.ci[2,3]),lwd=2)
```

```r
# II Graficado de bandas de intervalo de confianza
# 1. con seq() generamos una secuencia de 10 valores equidistantes de x entre 0 y 30
xseq <- data.frame(time=seq(0,30,length=100))

# 2 predecimos los valores puntuales, junto a los ci y pi asociados, para
#   cada uno de los 100 valores de x almacenados en xseq
ci.band <- exp(predict(model2.lin,newdata=xseq,interval="confidence",level=0.95))
pi.band <- exp(predict(model2.lin,newdata=xseq,interval="prediction",level=0.95))

# 3. Añadimos líneas en el rango de x (0:30), para cada una de las 100 predicciones
#    puntuales en este intervalo, almacenados en ci.band y pi.band
#    También incluimos una leyenda con legend()
lines(xseq[,1],ci.band[,2],lty=2)
lines(xseq[,1],ci.band[,3],lty=2)
lines(xseq[,1],pi.band[,2],lty=2,col="gray")
lines(xseq[,1],pi.band[,3],lty=2,col="gray")
legend("topright",legend=c("Fit","95% CI","95% PI"),lty=c(1,2,2),
       col=c("black","black","gray"),lwd=c(2,1,1))
```



- Tal y como ya habíamos diagnosticado, la varianza no es homogénea. Decrece marcadamente con el incremento de x, tal y como muestran las bandas de confiaza del 95% en torno al *ajuste* y a las *predicciones* de las observaciones.
- Nótese también que los errores son asimétricos arriba y abajo de la estima.

Como nota final, hay que tener en cuenta que estos intervalos de predicción son extremadamente sensibles a las desviaciones del supuesto de normalidad. Si hay claros indicios de que la variable de respuesta no presenta una distribución normal, considera usar *técnicas no-paramétricas*, **robustas a dichas violaciones**, como el *bootstrap*.

2 Regresión lineal simple - ejercicios de tarea

2.1 Datos

Usaremos el set de datos *states.x77* del paquete base *datasets* que R carga por defecto, el cual ya conocemos del *tema 8: correlación*.

2.2 Objetivo

Estudiaremos el efecto de las variables regresoras o predictoras "*Population*" e "*Illiteracy*" sobre la variable de respuesta "$Murder$"

Deben de:

1. Genera gráficos de dispersión para los pares de variables Murder ~ Population, Murder ~ Illiteracy del dataframe $states.x77$
2. Cálcula los coeficientes de correlación para las asociaciones entre Murder ~ Population, Murder ~ Illiteracy
3. Cálcula y representa gráficamente sobre los gráficos de dispersión correspondientes, las rectas de mínimos cuadrados para Murder ~ Population, Murder ~ Illiteracy
4. Evalúa gráficamente los supuestos del modelo lineal y comentalos brevemente
5. Evalúa si los modelos lineales son los más adecuados en ambos casos mediante el ajsute de una función cuadrática
6. Estima las tasas de homicidio para niveles de *Illiteracy* de 0.25, 1.2, 2.1, 3.0, 4.0; ¿Con qué valores estamos interpolando y con cuales extrapolando?

Los resultados los deben de subir al *sitio moodle* como un solo archivo *html* generado con *RStudio*.

3 Bibliografía selecta

3.1 Libros

3.1.1 R y estadística

- Andy Field, Discovering statistics using R (A. P. Field, Miles, and Field 2012)
- Michael J. Crawley, Statistics - An introduction using R (Crawley 2015)
- Tilman M. Davies - The book of R - a first course in programming and statistics (Davies 2016)

3.1.2 R - aprendiendo el lenguaje base

- Michael J. Crawley, The R book, 2nd edition (Crawley 2012)
- Paul Teetor, The R cookbook (Teetor and Loukides 2011)
- Richard Cotton, Learning R (Cotton 2013)
- Paul Murrell, R graphics (Murrell 2009)

3.1.3 R - manipulación, limpieza y visualización avanzada de datos con tidyr, dplyr y ggplot2

- Bradley C Boehmke, Data wrangling with R (Boehmke 2016)
- Hadley Wickham, ggplot2 - elegant graphics for data analysis (Wickham 2016)

3.1.4 R - aplicaciones en análisis de datos usando multiples paquetes

- Robert Kabacoff, R in action (Kabacoff 2015)
- Jared Lander, R for everyone (Lander 2014)

3.1.5 R - programación

- Garrett Grolemund, Hands on programming with R (Grolemund 2014)
- Norman Matloff, The art of R programming (Matloff 2011)

3.1.6 Investigación reproducible con R y RStudio

- Christopher Gandrud, Reproducible research with R and RStudio (Gandrud 2015)

4 Funciones y paquetes de R usados para este documento

4.1 Funciones de paquetes base (R Core Team 2018)

abline() AIC() anova() arrows() cat() class() coef() colnames() cor() curve() data.frame() exp() lm() head() I() legend() length() library() logLik() mean() lines() log() numeric() par() pairs() pf() plot() points() predict() pt() qf() qt() segments() seq() sqrt() str() subset() sum() summary() summary.aov() tail() which()

4.2 Datos del paquetes base

- datasets [R-datasets]

4.3 Paquetes especializados

- car (Fox, Weisberg, and Price 2018)

4.4 Paquetes y software para investigación reproducible y generación de documentos en múltiples formatos

- knitr (Xie 2018)
- pandoc (MacFerlane 2016)
- rmarkdown (Allaire et al. 2018)
- xtable (Dahl 2016)

5 Recursos en línea

5.1 The comprehensive R archive network (CRAN)

- CRAN

5.2 Cursos

- RStudio - online learning
- datacamp - learning R
- swirl - learn R, in R

5.3 Consulta

- R cookbook
- QuickR
- downloadable books o R and stats
- Use R!
- Official CRAN documentation
- [r], stackoverflow

5.4 Manipulación y graficado de datos con paquetes especializados

- plotly and gglplot2 user guide
- Data wrangling with R and RStudio
- Data wrangling with R and RStudio - cheatsheet
- Data wrangling with R and RStudio - webinar
- Bradley C Boehmke - Data wrangling with R

Referencias

Allaire, JJ, Yihui Xie, Jonathan McPherson, Javier Luraschi, Kevin Ushey, Aron Atkins, Hadley Wickham, Joe Cheng, and Winston Chang. 2018. *Rmarkdown: Dynamic Documents for R*. https://CRAN.R-project.org/package=rmarkdown.

Boehmke, Bradley. 2016. *Data Wrangling with R*. 1st ed. Springer. http://www.springer.com/la/book/9783319455983.

Cotton, Richard. 2013. *Learning R*. 1st ed. O'Reilly.

Crawley, Michael J. 2012. *The R book*. 2nd ed. Wiley.

———. 2015. *Statistics : an introduction using R*. 2nd ed. Wiley.

Dahl, David B. 2016. *Xtable: Export Tables to Latex or Html*. https://CRAN.R-project.org/package=xtable.

Davies, Tilman M. 2016. *The book of R - a first course in programming and statistics*. 1st ed. No Starch Press.

Field, Andy P., Jeremy Miles, and Zoe. Field. 2012. *Discovering statistics using R*. 1st ed. London: Sage.

Fox, John, Sanford Weisberg, and Brad Price. 2018. *Car: Companion to Applied Regression*. https://CRAN.R-project.org/package=car.

Gandrud, Christopher. 2015. *Reproducible research with R and RStudio*. 2nd ed. Chapman & Hall. https://github.com/christophergandrud/Rep-Res-Book.

Grolemund, Garrett. 2014. *Hands-on programming with R*. O'Reilly.

Kabacoff, Robert. 2015. *R in action : data analysis and graphics with R*. 2nd ed. Manning. https://github.com/kabacoff/RiA2 http://www.statmethods.net/.

Lander, Jared P. 2014. *R for everyone : advanced analytics and graphics*. New York, N.Y.: Addison-Wesley.

MacFerlane, John. 2016. "Pandoc - a universal document converter." http://pandoc.org/.

Matloff, Norman S. 2011. *The art of R programming : tour of statistical software design*. No Starch Press. http://heather.cs.ucdavis.edu/{~}matloff/132/NSPpart.pdf.

Murrell, Paul. 2009. "R Graphics." In *Wiley Interdisciplinary Reviews: Computational Statistics*, 1:216–20.

doi:10.1002/wics.22.

R Core Team. 2018. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

RStudio Team. 2016. *RStudio: Integrated Development Environment for R*. Boston, MA: RStudio, Inc. http://www.rstudio.com/.

Teetor, Paul, and Michael Kosta. Loukides. 2011. *R cookbook*. O'Reilly. http://www.cookbook-r.com/.

Wickham, Hadley. 2016. *Ggplot2*. 2nd ed. Use R! Cham: Springer International Publishing. doi:10.1007/978-3-319-24277-4.

Xie, Yihui. 2018. *Knitr: A General-Purpose Package for Dynamic Report Generation in R*. https://CRAN.R-project.org/package=knitr.