Entrega de ejercicios Tema 3

Blanca Cano Camarero

28 de diciembre de 2022

Indice de contenidos

Ejercicio 2 **3**
¿De cuántos países consta la muestra utilizada? . . . 4
¿Cuánto vale la suma de cuadrados que se utiliza para medir la variabilidad explicada por las tres variables regresoras? . . . 4
¿Cuánto vale la varianza muestral de la variable respuesta ? . . . 5
Contraste de la complejidad del modelo . . . 6
Contrasta a nivel $\alpha = 0.05$ la hipótesis nula $H_0 : \beta_1 = 0$ . . . 6
Contrasta a nivel $\alpha = 0.05$ la hipótesis nula $H_0 : \beta_1 = \beta_2 = \beta_3 = 0$ . . . 7
Estima la correlación entre $\hat{\beta}_1$ y $\hat{\beta}_2$. . . . 7
Calcula intervalos de confianza al nivel 90% para todos los β_i del modelo. . . . 8
Predice el valor de la esperanza de vida de los hombres en un país para el que el índice de natalidad es 29, la mortalidad infantil vale 50 y el logaritmo de su pnb vale 7. Calcula un intervalo de confianza del 95 % para el valor esperado de dicha variable. . . . 9

Ejercicio 4 **10**
Apartado primero . . . 10
Apartado 2 . . . 10
Apartado 3 . . . 11

Ejercicio 9 **12**
Selecciona el modelo óptimo . . . 14
Mejor modelo de acorde a R cuadrado ajustado . . . 16
Cp . . . 19
BIC . . . 21
Usando ahora el método iterativo . . . 24
Mejor modelo de acorde a R cuadrado ajustado con modelo iterativo hacia delante 25
Cp . . . 28

BIC . . . 31
Lasso . . . 34
Genera ahora las respuestas a partir del modelo . . . 35
Generación de los datos . . . 35
Modelo óptimo dentro de submodelos . . . 36
Submodelos óptimos . . . 37
Submodelos utilizando un método iterativo . . . 40
Lasso . . . 45
Conclusiones . . . 46

Ejercicio 12 **47**
Representación gráfica . . . 48
Ajuste de regresión de mínimos cuadrados . . . 49
Representación gráfica del error de predicción de validación cruzada generalizando en función de los grados de liberta utilizados. . . . 49
Comentario de los resultados . . . 53
Comportamiento general: Decrecemiento, mínimo y crecimiento más lento . . . 54
Gráficos desplazados . . . 54
A σ menor admita más grados de libertad antes de volver a crecer el error. . . . 54
Crecimiento mayor del error de $\sigma = 1$ frente a $\sigma = 0.5$ . . . 54

Ejercicio 2

Comenzaremos llamando a las bibliotecas que utilizaremos a lo largo de este ejercicio y cargando los datos relevantes.

```
library(magrittr)
library(dplyr)
```

```
Attaching package: 'dplyr'
```

```
The following objects are masked from 'package:stats':

    filter, lag
```

```
The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
library(purrr)
```

```
Attaching package: 'purrr'
```

```
The following object is masked from 'package:magrittr':

    set_names
```

```
natalidad <- read.table("https://verso.mat.uam.es/~joser.berrendero/datos/natalidad.txt",
  mutate(log_pnb = log(pnb))

head(natalidad)
```

```
   nat mort mortinf esph espm  pnb  log_pnb
1 24.7  5.7    30.8 69.6 75.5  600 6.396930
2 12.5 11.9    14.4 68.3 74.7 2250 7.718685
3 13.4 11.7    11.3 71.8 77.7 2980 7.999679
4 11.6 13.4    14.8 65.4 73.8 2780 7.930206
5 14.3 10.2    16.0 67.2 75.7 1690 7.432484
6 13.6 10.7    26.9 66.5 72.4 1640 7.402452
```

Se desea estudiar la esperanza de vida de los hombres como función lineal de la tasa de natalidad, la tasa de mortalidad infantil y el logaritmo del producto nacional bruto. Para ello se ajusta un modelo de regresión lineal múltiple, con los resultados siguientes:

```
reg <- lm(esph ~ nat  + mortinf + log_pnb, data = natalidad)
summary(reg)
```

```
Call:
lm(formula = esph ~ nat + mortinf + log_pnb, data = natalidad)

Residuals:
    Min      1Q  Median      3Q     Max
-8.4893 -2.1660  0.1581  2.0663  7.9084

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 65.95088    3.25642  20.253  < 2e-16 ***
nat         -0.14621    0.04762  -3.071  0.00285 **
mortinf     -0.13312    0.01537  -8.663  2.2e-13 ***
log_pnb      0.94478    0.32989   2.864  0.00524 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.128 on 87 degrees of freedom
Multiple R-squared:    0.9, Adjusted R-squared:  0.8966
F-statistic: 261.1 on 3 and 87 DF,  p-value: < 2.2e-16
```

¿De cuántos países consta la muestra utilizada?

Cada fila se corresponde con un país distinto luego habrá:

```
n <- nrow(natalidad)
n
```

```
[1] 91
```

¿Cuánto vale la suma de cuadrados que se utiliza para medir la variabilidad explicada por las tres variables regresoras?

Se busca calcular la suma de cuadrados explicada que se calcula como

$$SCE = \sum_{i=1}^{n} \left(\hat{Y}_i - \bar{Y}\right)^2.$$

Este dato no aparece directamente en el *summary* del la regresión lineal, luego habrá que calcularlo de manera manual.

```
media_vida <- mean(natalidad$esph)
p <- predict(reg, natalidad)
SCE <- Reduce('+',map(p, function(y_hat) (y_hat - media_vida)^2))
cat("La suma de cuadrados resultante es",SCE, '\n')
```

```
La suma de cuadrados resultante es 7665.063
```

¿Cuánto vale la varianza muestral de la variable respuesta ?

Un estimador de la varianza muestrar vendrá dado como

$$\text{Varianza muestrar } = \frac{1}{n-1}\sum_{i=1}^{n}(Y_i - \bar{Y})^2 = \frac{SCT}{n-1}$$

Usando la ortogonalidad de los residuos con las variables regresoras se tiene que

$$SCT = SCE + SCR$$

Donde SCE proviene del apartado anterior y

$$SCR = \sum_{i=0}^{n} e^2$$

y

$$e = Y - \hat{Y}.$$

```
# Teniendo presente que ya hemos calculado las predicciones en la variable p
e <- natalidad$esph - p
SCR <- Reduce('+',map(e, function(x) x^2))
SCT <- SCE + SCR
VMR <- SCT / (n-1)
cat('la varianza muestral de la variable respuesta vale ', VMR)
```

```
la varianza muestral de la variable respuesta vale  94.62798
```

Contraste de la complejidad del modelo

Se desea que contrastar que $H_0 : \beta_1 = 0$
esto viene determinado por el sistema matricial A que es un vector fila con un uno en 1 y el resto 0.

Debe de satisfacer que $A\beta = 0$.

Definimos el modelo reducido $M0$ que resulta de imponer las restricciones de H_0.

Bajo $H_0 : A\beta = 0$ se verifica que

$$\frac{(SCR_0 - SCR)/k}{SCR/(n-p-1)} \equiv F_{k,n-p-1},$$

donde k es el número de restricciones (rango de A ciertamente), $p+1$ el número de β (variables del modelo a ajustar) y n el número de observaciones.

La región crítica del contraste para nivel α es

$$R = \left\{ \frac{(SCR_0 - SCR)/k}{SCR/(n-p-1)} > F_{k,n-p-1} \right\}$$

Utilizaremos el comando `anova` para R.

Contrasta a nivel $\alpha = 0.05$ la hipótesis nula $H_0 : \beta_1 = 0$

Solución

Al suponer que $\beta_1 = 0$ entonces ahora el modelo predeciría de natalidad, esto sería

```
reg0 <- lm(esph ~  mortinf + log_pnb, data = natalidad)
```

```
anova(reg0, reg)
```

```
Analysis of Variance Table

Model 1: esph ~ mortinf + log_pnb
Model 2: esph ~ nat + mortinf + log_pnb
  Res.Df    RSS Df Sum of Sq      F  Pr(>F)
1     88 943.73
2     87 851.46  1    92.277 9.4286 0.00285 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

De aquí deducimos que la región crítica $F = 9.411$ y lo que buscábamos, que la probabilidad de pertenecer a la región crítica siendo la H_0 cierta es de

$$Pr(> F) = 0.00285$$

Como $ 0.00285 < 0.05$ rechazamos entonces la hipótesis nula.

Contrasta a nivel $\alpha = 0.05$ la hipótesis nula $H_0 : \beta_1 = \beta_2 = \beta_3 = 0$

Contrastaremos que solo se pueda aproximar con la media

```
reg0 <- lm(esph ~ 1, data = natalidad) # ahora sería solo con la media
anova(reg0, reg)
```

```
Analysis of Variance Table

Model 1: esph ~ 1
Model 2: esph ~ nat + mortinf + log_pnb
  Res.Df    RSS Df Sum of Sq      F    Pr(>F)
1     90 8516.5
2     87  851.5  3    7665.1 261.07 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Como podemos observar, ahora

$$Pr(> F) < 2.2e - 16$$

es decir prácticamente 0, luego se rechaza la hipótesis nula.

Ya que si la media fuera un buen estimador se podría pensar que los modelos no tienen nada que ver con el modelo.

Estima la correlación entre $\hat{\beta}_1$ y $\hat{\beta}_2$.

Esto se hará con el comando `vcoc` que devuelve la matriz de covarianza de los parámetros ajustados.

$\hat{\beta}_1$ se corresponde a `nat` y $\hat{\beta}_2$ se corresponde con `mortinf` luego mirando la respectiva fila y columna tenemos que

```
vcov(reg)
```

```
            (Intercept)           nat       mortinf      log_pnb
(Intercept) 10.60424770 -0.0636441427 -0.0154732224 -1.033908908
nat         -0.06364414  0.0022673659 -0.0004865478  0.003160908
mortinf     -0.01547322 -0.0004865478  0.0002361416  0.002230265
log_pnb     -1.03390891  0.0031609079  0.0022302652  0.108830676
```

$$cov(\hat{\beta}_1, \hat{\beta}_2) = -0.0004865478$$

Puesto que el resultado es cercano a 0 podemos pensar que el signo de una es independiente del de la otra, es decir independientes.

Por lo tanto apriori no podríamos explicar una varia con la otra con un modelo lineal.

Calcula intervalos de confianza al nivel 90% para todos los β_i del modelo.

Usaremos la función `confint`

```
confint(reg, level=0.90)
```

```
                  5 %        95 %
(Intercept) 60.5369009 71.36485887
nat         -0.2253786 -0.06704702
mortinf     -0.1586652 -0.10756848
log_pnb      0.3963072  1.49324572
```

esto nos proporciona un rango de incertidumbre en el que puede oscilar cada uno de los respectivos parámetros con un 90% de probabilidad.

$$\beta_0 \in [60.537, 71.365],$$
$$\beta_{\text{nat}} \in [-0.225 - 0.067],$$
$$\beta_{\text{mortinf}} \in [-0.158, -0.108],$$
$$\beta_{\text{log pnb}} \in [0.396, 1.493].$$

Ya

$$P(\beta_{inf} \leq \beta \leq \beta_{sup}) = 1 - \alpha.$$

Donde β_{inf} y β_{sup} son los elementos inferiores y superiores del intervalo.

Cuando mayor sea el intervalo más incertidumbre tendremos de la corrección del parámetro.

Predice el valor de la esperanza de vida de los hombres en un país para el que el índice de natalidad es 29, la mortalidad infantil vale 50 y el logaritmo de su pnb vale 7. Calcula un intervalo de confianza del 95 % para el valor esperado de dicha variable.

Para ello usaremos la función `predict`

```
# Debemos de escribir el data frame con el nombre exacto
#bde las columnas con las que se creo el modelo de regresión.
# Estas son:
# nat     mortinf     log_pnb
new_data <- data.frame(
  nat = 29,
  mortinf = 50,
  log_pnb = 7
)
# Realizamos la predicción
predict(reg, new_data, interval = "confidence", level = 0.95 )
```

```
      fit      lwr    upr
1 61.6683 60.88761 62.449
```

Por lo tanto la esperanza de visa de los hombres de ese pais es de 61.6683 años. El intervalo de confianza es de $[60.88761, 62.449]$ es decir valores sobre los que podría oscilar el valor con un 95% de confianza.

Además cabe mencionar que la predicción del intervalo se basa en que el error residual están distribuidos bajo una distribución normal y varianza constante. Luego solo deberíamos de usar estos intervalos si tuviéramos razones para pensar que se da esta condición.

Ejercicio 4

Sean Y_1, Y_2, Y_3 tres variables aleatorias independientes con distribución normal y varianza σ^2. Supongamos que μ es la media de Y_1, λ es la media de Y_2, $\lambda + \mu$ es la media de Y_3, donde $\lambda, \mu \in \mathbb{R}$.

Apartado primero

Demuestra que el vector $Y = (Y_1, Y_2, Y_3)'$ verifica el modelo de regresión múltiple

$$Y = X\beta + \epsilon.$$

Para ello, determina la matriz de diseño X, el vector de parámetros β y la distribución de las variables de error ϵ.

Solución

$$X = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}$$

$\beta = (\mu, \lambda)'$

$\epsilon \sim \mathcal{N}((0,0,0)', \sigma^2 Id_3)$.

Apartado 2

Calcula los estimadores de máxima verosimilitud (equivalentemente, de mínimos cuadrados) de λ y μ.

Solución

Esto no sé justificarlo matemáticamente, pero teniendo en cuenta que el estimador máximo verosímil de la media de la normal es la media y que para estimador tenemos dos datos:

$$\lambda = \frac{Y_2 + (Y_3 - Y_1)}{2},$$

$$\mu = \frac{Y_1 + (Y_3 - Y_2)}{2}.$$

Apartado 3

Calcula la distribución del vector $(\widehat{\lambda}, \widehat{\mu})'$, formado por los estimadores calculados en el apartado anterior.

Solución

La distribución de los estimadores viene dado por

$$\hat{\beta} \sim \mathcal{N}(\beta, \sigma^2(X'X)^{-1})$$

Por lo que

$$X' = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}.$$

$$X'X = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}.$$

Calculando la inversa por adjuntos resulta que

$$(X'X)^{-1} = \frac{1}{3}\begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}.$$

Por lo que

$$\hat{\beta} \sim \mathcal{N}\left(\beta, \sigma^2\frac{1}{3}\begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}\right).$$

Ejercicio 9

Genera aleatoriamente una variable regresora X y un vector aleatorio ϵ de longitud $n = 100$, con distribución normal estándar e independientes.

Genera la variable respuesta de acuerdo con el modelo:

$$Y = X + X^2 + X^3 + \epsilon.$$

```
library(purrr)
library(ggplot2)
library(ramify)
```

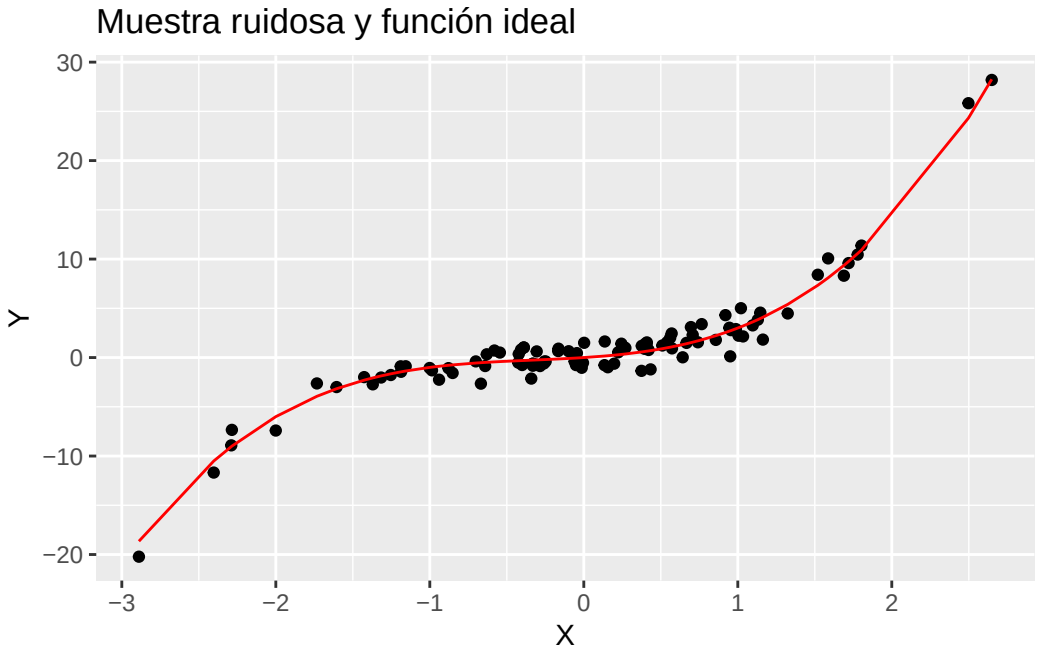
```
Attaching package: 'ramify'
```

```
The following object is masked from 'package:purrr':

    flatten
```

```
The following object is masked from 'package:graphics':

    clip
```

```
set.seed(1)
n <- 100
X_distribution <- rnorm
error_distribution <- rnorm
modelo_1 <- function (x) {
  return (x + x^2 + x^3)
}

Y_distribution <- function (n, t){
  x <- X_distribution(n)
  e <- error_distribution(n)
  return (sapply(x, t) + e)
}


Y_1 <- Y_distribution(n, modelo_1)
```

```
X <- X_distribution(n)
# Generamos las potencias
datos <- data.frame('X' = X)
#datos[1] <- X
for (i in 2:10) {
  datos[i] <- datos[1]*datos[i-1]
}

# Modelo1
Y <- unlist(as.list(datos[1]+datos[2] + datos[3] + error_distribution(n)))
datos$'Y' <- Y
datos2 <- datos
nombre_variables <- c('X1', 'X2',  'X3', 'X4', 'X5',
                      'X6',  'X7', 'X8', 'X9', 'X10', 'y')
colnames(datos) <- nombre_variables
colnames(datos2) <- nombre_variables
datos2$'Y_ideal' <- sapply(X, function(x)(x*(1 + x*(1 + x))))

p <- ggplot(datos2) +
  geom_point( aes(X, Y), colour = 'black' ) +
  geom_line(aes(X, Y_ideal), colour = 'red' ) +
  ggtitle('Muestra ruidosa y función ideal')

p
```



Selecciona el modelo óptimo

Entre todos los submodelos que contienen como variables regresoras $X, X^2, X^3, \ldots, X^{10}$. ¿Cuál es el mejor modelo de acuerdo con los criterios C_p, BIC y R_a^2?

Solución

Para poder realizar estas comparaciones utilizaremos la función `leaps::regsubsets`.

Donde los argumentos que nos interesan son:

- `datos`
- `method` donde indicaremos de si se trata de `exhaustive` o `forward`.
- `nvmax` número máximo de subconjuntos a examinar, la pondremos al máximo de parámetros que tenemos.
- `intercept` Si añadimos sesgo o no.
- `adjr2` para Adjusted r-squared.
- `cp` para Mallows's CP.
- `bic` para Schwartz's information criterion BIC.

Nodemos además que en el modelos que nos piden no hay término independiente (ya que por el contrario habría añadido un `1` a las variables regresoras), luego tendremos que hacer `intercept = FALSE`.

```r
subconjuntos_a_examinar <- 10
modelo_exhaustivo <- leaps::regsubsets(
  y ~ .,
  data=datos,
  nvmax= subconjuntos_a_examinar, # para que utilice todos las componentes posibles,
  intercept = FALSE
  )

resumen_1 <- summary(modelo_exhaustivo)
resumen_1
```

```
Subset selection object
Call: regsubsets.formula(y ~ ., data = datos, nvmax = subconjuntos_a_examinar,
    intercept = FALSE)
10 Variables
    Forced in Forced out
X2      FALSE      FALSE
X3      FALSE      FALSE
X4      FALSE      FALSE
X5      FALSE      FALSE
X6      FALSE      FALSE
X7      FALSE      FALSE
X8      FALSE      FALSE
X9      FALSE      FALSE
X10     FALSE      FALSE
1 subsets of each size up to 10
Selection Algorithm: exhaustive
          X1  X2  X3  X4  X5  X6  X7  X8  X9  X10
1  ( 1 )  " " " " "*" " " " " " " " " " " " " "
2  ( 1 )  " " "*" "*" " " " " " " " " " " " " "
3  ( 1 )  "*" "*" "*" " " " " " " " " " " " " "
4  ( 1 )  "*" "*" "*" " " "*" " " " " " " " " "
5  ( 1 )  "*" "*" "*" " " " " " " "*" " " "*" " "
6  ( 1 )  "*" "*" " " " " " " "*" "*" "*" " " " " "*"
7  ( 1 )  "*" "*" "*" " " " " " " "*" "*" " " "*" "*"
8  ( 1 )  "*" "*" "*" "*" " " " " "*" "*" "*" "*" " "
9  ( 1 )  "*" "*" "*" "*" "*" "*" "*" " " "*" "*" "*"
10  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "*"
```

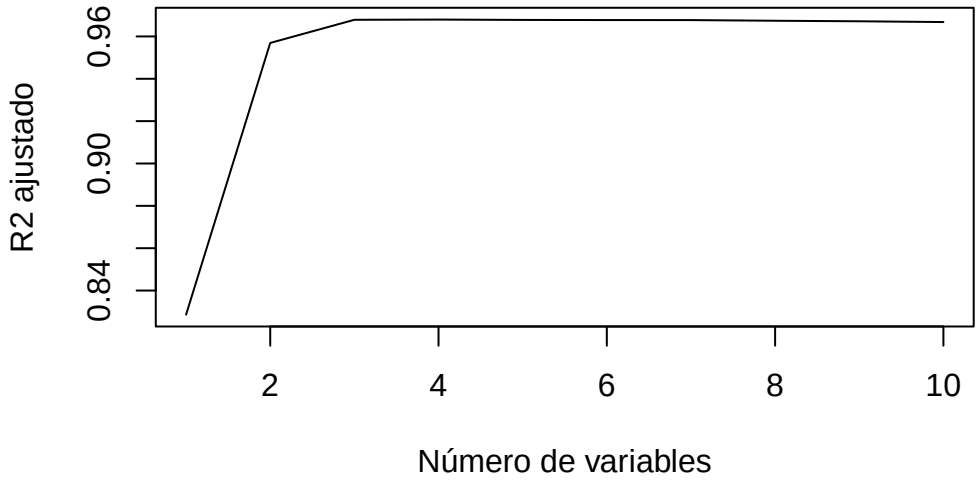
```r
# Función para pintar
pinta_ajuste <- function(y_criterio, label){
```

```
  plot_data <- data.frame(
    x = 1:subconjuntos_a_examinar,
    y = y_criterio
  )
  print(plot_data)
  y_label <- label

  plot(plot_data$x, plot_data$y,
       type = "l",
       xlab = 'Número de variables',
       ylab = y_label)
}
```

Mejor modelo de acorde a R cuadrado ajustado

```
pinta_ajuste(resumen_1$adjr2, "R2 ajustado")
```

```
    x         y
1   1 0.8286048
2   2 0.9569257
3   3 0.9678546
4   4 0.9679619
5   5 0.9677913
6   6 0.9677254
7   7 0.9677174
8   8 0.9674060
9   9 0.9671478
10 10 0.9667829
```




```
cat("Se alcanza el máximo con ", which.max(resumen_1$adjr2), "parametros")
```

```
Se alcanza el máximo con  4 parametros
```

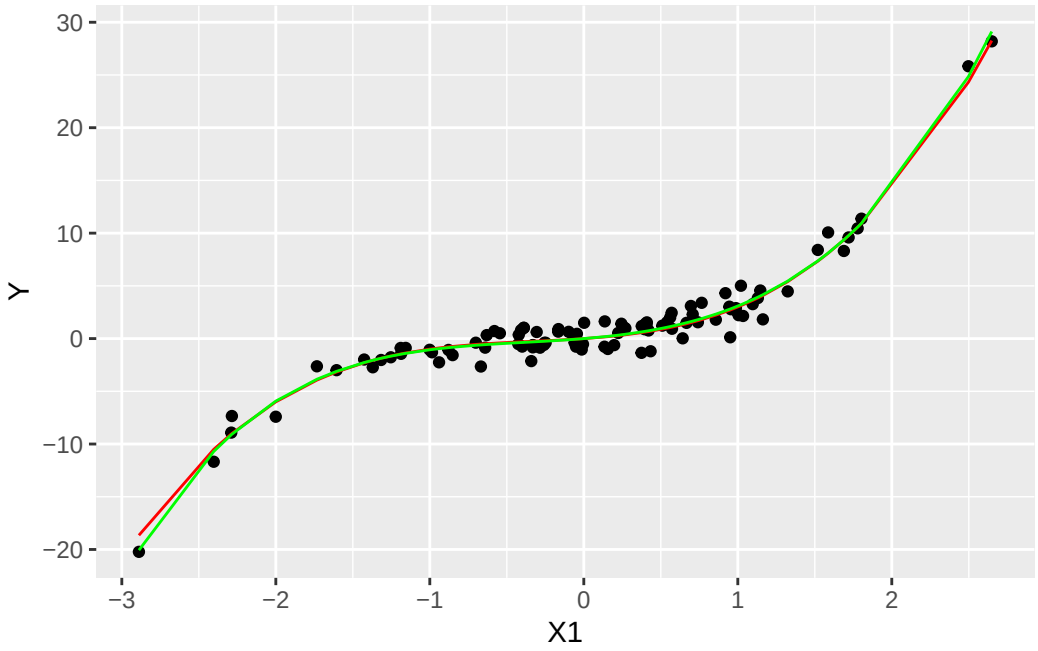
Será mejor donde alcance un máximo, esto es en $p = 4$ es decir utilizando 4 parámetros. Vamos a proceder a analizar los coeficientes.

```
# Mejores elementos
#Selection Algorithm: exhaustive
#          X1  X2  X3  X4  X5  X6  X7  X8  X9  X10
#4  ( 1 )  "*" "*" "*" " " "*" " " " " " " " " " "

modelo_r2_a <- lm(y ~ X1 + X2 + X3  + X5 -1, data=datos)
summary(modelo_r2_a)
```

```
Call:
lm(formula = y ~ X1 + X2 + X3 + X5 - 1, data = datos)

Residuals:
     Min       1Q   Median       3Q      Max
-2.68557 -0.51999 -0.00382  0.74572  1.95348

Coefficients:
   Estimate Std. Error t value Pr(>|t|)
X1  1.18463    0.24565   4.822 5.32e-06 ***
X2  1.01504    0.05303  19.140  < 2e-16 ***
```

```
X3  0.86982    0.15119   5.753 1.04e-07 ***
X5  0.02059    0.01788   1.151    0.253
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9905 on 96 degrees of freedom
Multiple R-squared:  0.9692,    Adjusted R-squared:  0.968
F-statistic: 756.3 on 4 and 96 DF,  p-value: < 2.2e-16
```

```r
y_predit_r2_a <- predict(object=modelo_r2_a, newdata=datos)

datos2$'Y_r2a' <- y_predit_r2_a

p <- ggplot(datos2) +
  geom_point( aes(X1, Y), colour = 'black' ) +
  geom_line(aes(X1, Y_ideal), colour = 'red' ) +
  geom_line(aes(X1, Y_r2a), colour = 'green' )

p
```

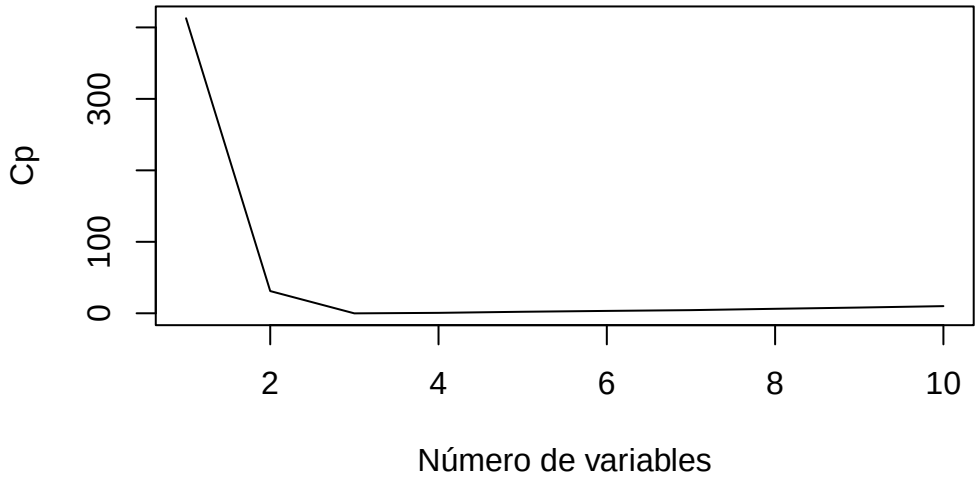


Como vemos más variables lo que hacen es acercarse al ruido.

Cp

```
pinta_ajuste(resumen_1$cp, "Cp")
```

```
    x           y
1   1 412.8247879
2   2  31.0815100
3   3  -0.1295806
4   4   0.5924715
5   5   2.1159807
6   6   3.3327617
7   7   4.3835952
8   8   6.2740547
9   9   8.0003139
10 10  10.0000000
```




```
cat("Con el criterio CP, el mejor modelo tiene ", which.min(resumen_1$cp), "parametros\n")
```

```
Con el criterio CP, el mejor modelo tiene  3 parametros
```

Vemos que con este criterio alcanza un mínimo en tres variables. es decir que el mejor de los resultados sería

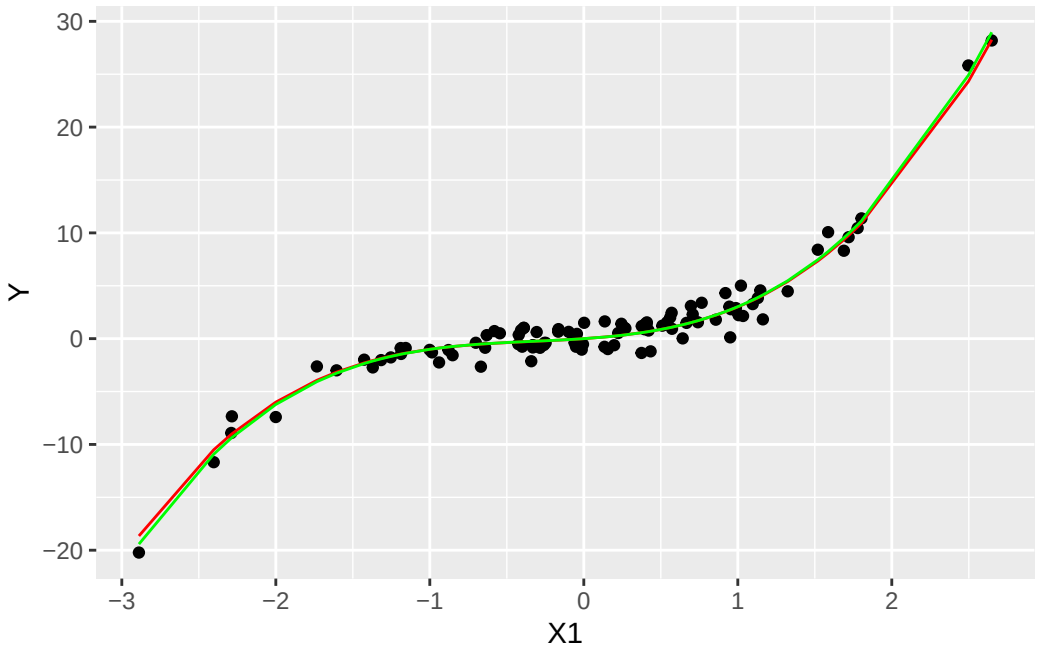
```
Selection Algorithm: exhaustive
         X1  X2  X3  X4  X5  X6  X7  X8  X9  X10
3  ( 1 )  "*" "*" "*" " " " " " " " " " " " " " "
```

que si entrenamos obtendríamos

```
modelo_cp<- lm(y ~ X1 + X2 + X3  -1, data=datos)
summary(modelo_cp)
```

```
Call:
lm(formula = y ~ X1 + X2 + X3 - 1, data = datos)

Residuals:
     Min       1Q   Median       3Q      Max
-2.61090 -0.54234  0.02662  0.74564  2.00549

Coefficients:
   Estimate Std. Error t value Pr(>|t|)
X1  0.97666    0.16672   5.858 6.41e-08 ***
X2  1.00906    0.05287  19.087  < 2e-16 ***
X3  1.03788    0.03935  26.377  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9922 on 97 degrees of freedom
Multiple R-squared:  0.9688,    Adjusted R-squared:  0.9679
F-statistic:  1005 on 3 and 97 DF,  p-value: < 2.2e-16
```

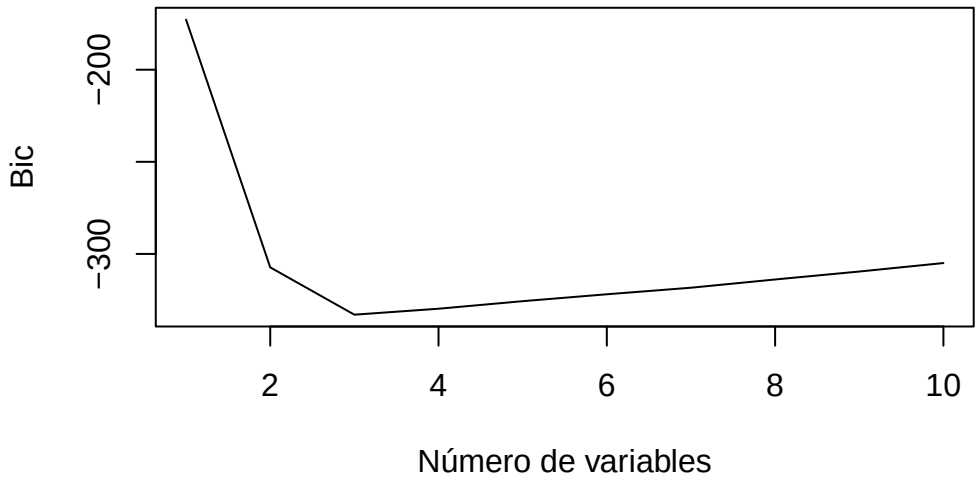
```
y_predit_cp <- predict(object=modelo_cp, newdata=datos)

datos2$'Y_cp' <- y_predit_cp

p <- ggplot(datos2) +
  geom_point( aes(X1, Y), colour = 'black' ) +
  geom_line(aes(X1, Y_ideal), colour = 'red' ) +
  geom_line(aes(X1, Y_cp), colour = 'green' )

p
```



BIC

```
pinta_ajuste(resumen_1$bic, "Bic")
```

```
    x         y
1   1 -172.7782
2   2 -307.2928
3   3 -332.9790
4   4 -329.7446
5   5 -325.6553
6   6 -321.9040
7   7 -318.3436
8   8 -313.8596
9   9 -309.5582
10 10 -304.9534
```




```
cat("Con el criterio BIC, el mejor modelo tiene ", which.min(resumen_1$bic), "parametros\n
```

```
Con el criterio BIC, el mejor modelo tiene  3 parametros
```

```
summary(modelo_exhaustivo)
```

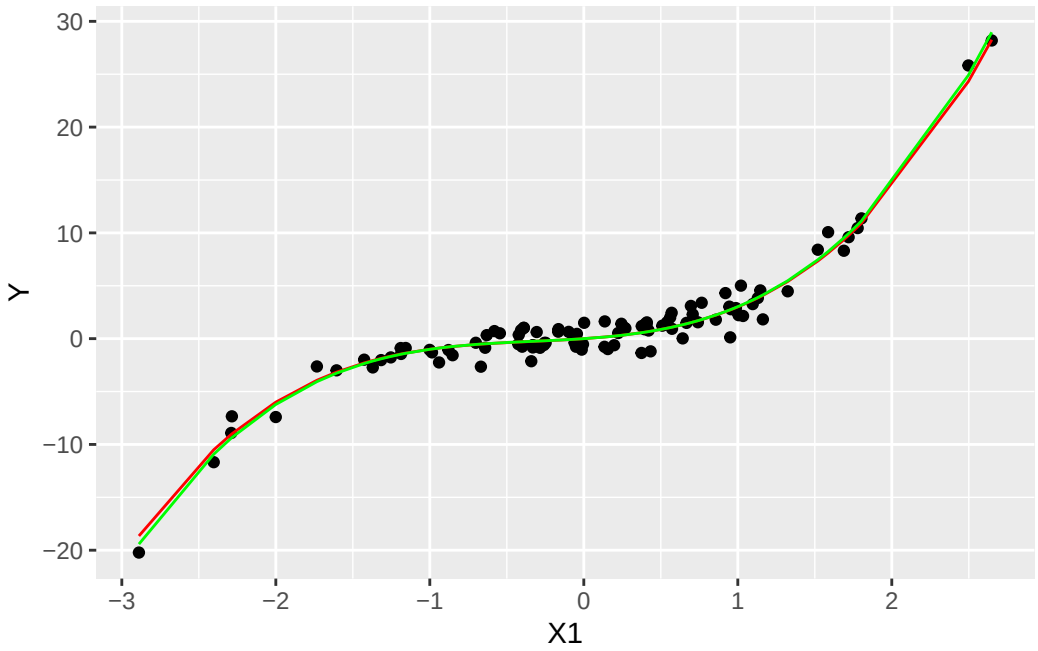
```
Subset selection object
Call: regsubsets.formula(y ~ ., data = datos, nvmax = subconjuntos_a_examinar,
    intercept = FALSE)
10 Variables
    Forced in Forced out
X2      FALSE      FALSE
X3      FALSE      FALSE
X4      FALSE      FALSE
X5      FALSE      FALSE
X6      FALSE      FALSE
X7      FALSE      FALSE
X8      FALSE      FALSE
X9      FALSE      FALSE
X10     FALSE      FALSE
1 subsets of each size up to 10
Selection Algorithm: exhaustive
          X1  X2  X3  X4  X5  X6  X7  X8  X9  X10
1  ( 1 )  " " " " "*" " " " " " " " " " " " " "
2  ( 1 )  " " "*" "*" " " " " " " " " " " " " "
3  ( 1 )  "*" "*" "*" " " " " " " " " " " " " "
```

```
4  ( 1 )  "*" "*" "*" " " "*" " " " " " " " " " " " "
5  ( 1 )  "*" "*" "*" " " " " " " "*" " " "*" " " " " "
6  ( 1 )  "*" "*" " " " " " " "*" "*" "*" " " " " " " "*"
7  ( 1 )  "*" "*" "*" " " " " " " "*" "*" " " " " "*" "*"
8  ( 1 )  "*" "*" "*" "*" " " " " "*" "*" "*" "*" " " "
9  ( 1 )  "*" "*" "*" "*" "*" "*" " " " " "*" "*" "*"
10  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" "*" "*" "*"
```

```
modelo_bic<- lm(y ~ X1 + X2 + X3 -1, data=datos)
summary(modelo_bic)
```

```
Call:
lm(formula = y ~ X1 + X2 + X3 - 1, data = datos)

Residuals:
     Min       1Q   Median       3Q      Max
-2.61090 -0.54234  0.02662  0.74564  2.00549

Coefficients:
   Estimate Std. Error t value Pr(>|t|)
X1  0.97666    0.16672   5.858 6.41e-08 ***
X2  1.00906    0.05287  19.087  < 2e-16 ***
X3  1.03788    0.03935  26.377  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9922 on 97 degrees of freedom
Multiple R-squared:  0.9688,    Adjusted R-squared:  0.9679
F-statistic:  1005 on 3 and 97 DF,  p-value: < 2.2e-16
```

```
y_predit_bic <- predict(object=modelo_bic, newdata=datos)

datos2$'Y_bic' <- y_predit_bic

p <- ggplot(datos2) +
  geom_point( aes(X1, Y), colour = 'black' ) +
  geom_line(aes(X1, Y_ideal), colour = 'red' ) +
  geom_line(aes(X1, Y_bic), colour = 'green' )

p
```



Usando ahora el método iterativo

```
modelo_iterativo_delante <- leaps::regsubsets(
  y ~ .,
  data=datos,
  method = "forward",
  nvmax = subconjuntos_a_examinar,
  intercept = FALSE
  )
resumen_2 <- summary(modelo_iterativo_delante)
resumen_2
```

```
Subset selection object
Call: regsubsets.formula(y ~ ., data = datos, method = "forward", nvmax = subconjuntos_a_exar
    intercept = FALSE)
10 Variables
    Forced in Forced out
X2      FALSE      FALSE
X3      FALSE      FALSE
X4      FALSE      FALSE
X5      FALSE      FALSE
X6      FALSE      FALSE
```

```
X7      FALSE      FALSE
X8      FALSE      FALSE
X9      FALSE      FALSE
X10     FALSE      FALSE
1 subsets of each size up to 10
Selection Algorithm: forward
          X1  X2  X3  X4  X5  X6  X7  X8  X9  X10
1  ( 1 )  " " " " "*" " " " " " " " " " " " " "
2  ( 1 )  " " "*" "*" " " " " " " " " " " " " "
3  ( 1 )  "*" "*" "*" " " " " " " " " " " " " "
4  ( 1 )  "*" "*" "*" " " "*" " " " " " " " " "
5  ( 1 )  "*" "*" "*" " " "*" " " " " " " " " "*"
6  ( 1 )  "*" "*" "*" " " "*" " " "*" " " " " "*"
7  ( 1 )  "*" "*" "*" " " "*" "*" "*" " " " " "*"
8  ( 1 )  "*" "*" "*" " " "*" "*" "*" " " "*" "*"
9  ( 1 )  "*" "*" "*" "*" "*" "*" "*" " " "*" "*"
10  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" "*" "*"
```

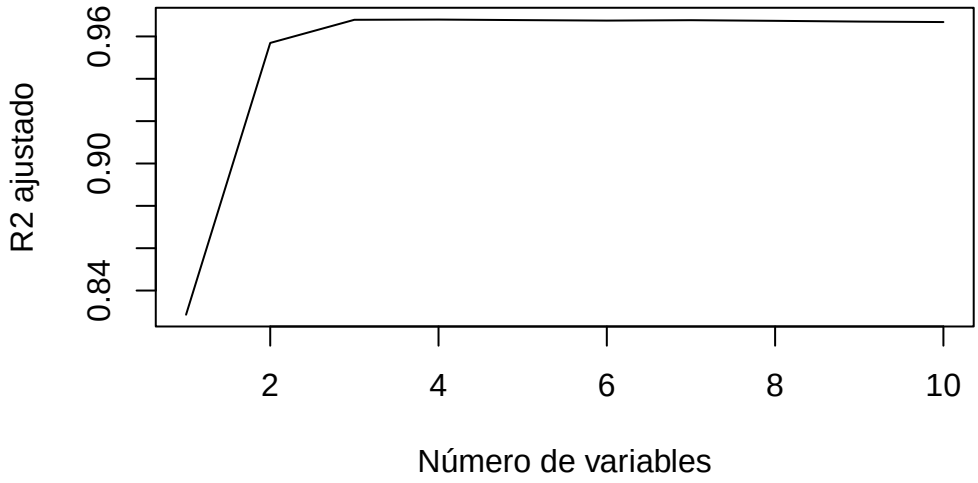
Observaciones:

Como vemos ahora las variables seleccionadas para p parámetros están incluidas en el modelo con $p+1$ parámetros. Esto es lo esperado ya que es el comportamiento de forward con el fin de mejorar el rendimiento.

Mejor modelo de acorde a R cuadrado ajustado con modelo iterativo hacia delante

```r
pinta_ajuste(resumen_2$adjr2, "R2 ajustado")
```

```
   x         y
1   1 0.8286048
2   2 0.9569257
3   3 0.9678546
4   4 0.9679619
5   5 0.9677077
6   6 0.9674817
7   7 0.9676667
8   8 0.9673671
9   9 0.9670145
10 10 0.9667829
```




```
cat("Se alcanza el máximo con ", which.max(resumen_2$adjr2), "parametros")
```

```
Se alcanza el máximo con  4 parametros
```

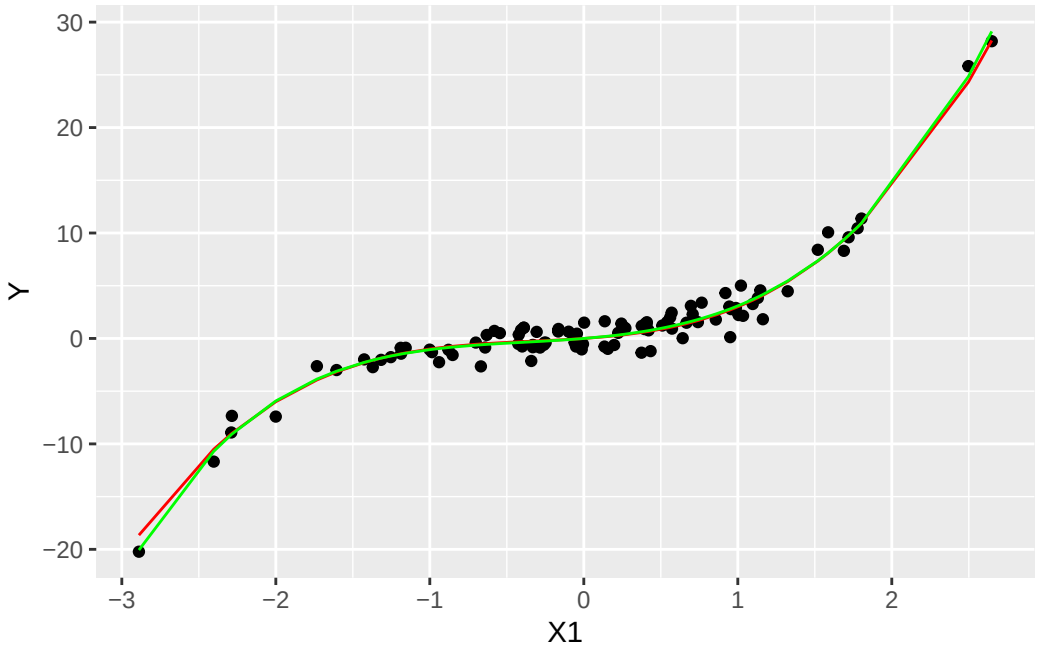
Será mejor donde alcance un máximo, esto es en $p = 8$ es decir utilizando 8 parámetros. Vamos a proceder a analizar los coeficientes.

```
#          X1  X2  X3  X4  X5  X6  X7  X8  X9  X10
#4  ( 1 )  "*" "*" "*" " " "*" " " " " " " " " " "


modelo_r2_a <- lm(y ~ X1 + X2 + X3 +  X5 +-1, data=datos)
summary(modelo_r2_a)
```

```
Call:
lm(formula = y ~ X1 + X2 + X3 + X5 + -1, data = datos)

Residuals:
     Min       1Q   Median       3Q      Max
-2.68557 -0.51999 -0.00382  0.74572  1.95348

Coefficients:
   Estimate Std. Error t value Pr(>|t|)
X1  1.18463    0.24565   4.822 5.32e-06 ***
X2  1.01504    0.05303  19.140  < 2e-16 ***
X3  0.86982    0.15119   5.753 1.04e-07 ***
```

```
X5  0.02059    0.01788   1.151    0.253
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9905 on 96 degrees of freedom
Multiple R-squared:  0.9692,    Adjusted R-squared:  0.968
F-statistic: 756.3 on 4 and 96 DF,  p-value: < 2.2e-16
```

```r
y_predit_r2_a <- predict(object=modelo_r2_a, newdata=datos)

datos2$'Y_r2a' <- y_predit_r2_a

p <- ggplot(datos2) +
  geom_point( aes(X1, Y), colour = 'black' ) +
  geom_line(aes(X1, Y_ideal), colour = 'red' ) +
  geom_line(aes(X1, Y_r2a), colour = 'green' )

p
```

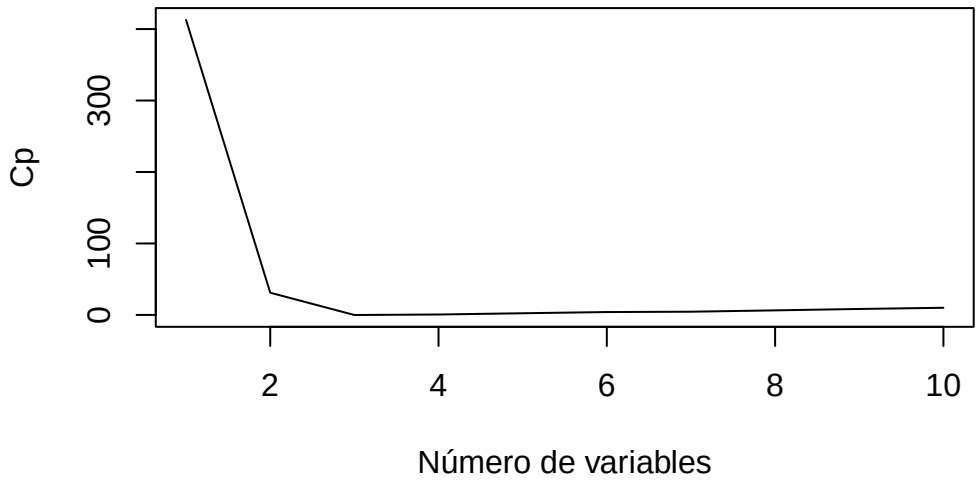


Como vemos más variables lo que hacen es acercarse al ruido. El error `djusted R-squared: 0.968` coincide con el del criterio.

Cp

```
pinta_ajuste(resumen_2$cp, "Cp")
```

```
    x           y
1   1 412.8247879
2   2  31.0815100
3   3  -0.1295806
4   4   0.5924715
5   5   2.3549212
6   6   4.0223717
7   7   4.5256530
8   8   6.3819508
9   9   8.3654007
10 10  10.0000000
```




```
cat("Con el criterio CP, el mejor modelo tiene ", which.min(resumen_2$cp), "parametros\n")
```

```
Con el criterio CP, el mejor modelo tiene  3 parametros
```

```
summary(modelo_iterativo_delante)
```

```
Subset selection object
Call: regsubsets.formula(y ~ ., data = datos, method = "forward", nvmax = subconjuntos_a_exa
```

```
    intercept = FALSE)
10 Variables
    Forced in Forced out
X2      FALSE      FALSE
X3      FALSE      FALSE
X4      FALSE      FALSE
X5      FALSE      FALSE
X6      FALSE      FALSE
X7      FALSE      FALSE
X8      FALSE      FALSE
X9      FALSE      FALSE
X10     FALSE      FALSE
1 subsets of each size up to 10
Selection Algorithm: forward
          X1  X2  X3  X4  X5  X6  X7  X8  X9  X10
1  ( 1 )  " " " " "*" " " " " " " " " " " " " "
2  ( 1 )  " " "*" "*" " " " " " " " " " " " " "
3  ( 1 )  "*" "*" "*" " " " " " " " " " " " " "
4  ( 1 )  "*" "*" "*" " " "*" " " " " " " " " "
5  ( 1 )  "*" "*" "*" " " "*" " " " " " " " "*"
6  ( 1 )  "*" "*" "*" " " "*" " " "*" " " " " "*"
7  ( 1 )  "*" "*" "*" " " "*" "*" "*" " " " " "*"
8  ( 1 )  "*" "*" "*" " " "*" "*" "*" " " "*" "*"
9  ( 1 )  "*" "*" "*" "*" "*" "*" "*" " " "*" "*"
10  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" "*" "*"
```

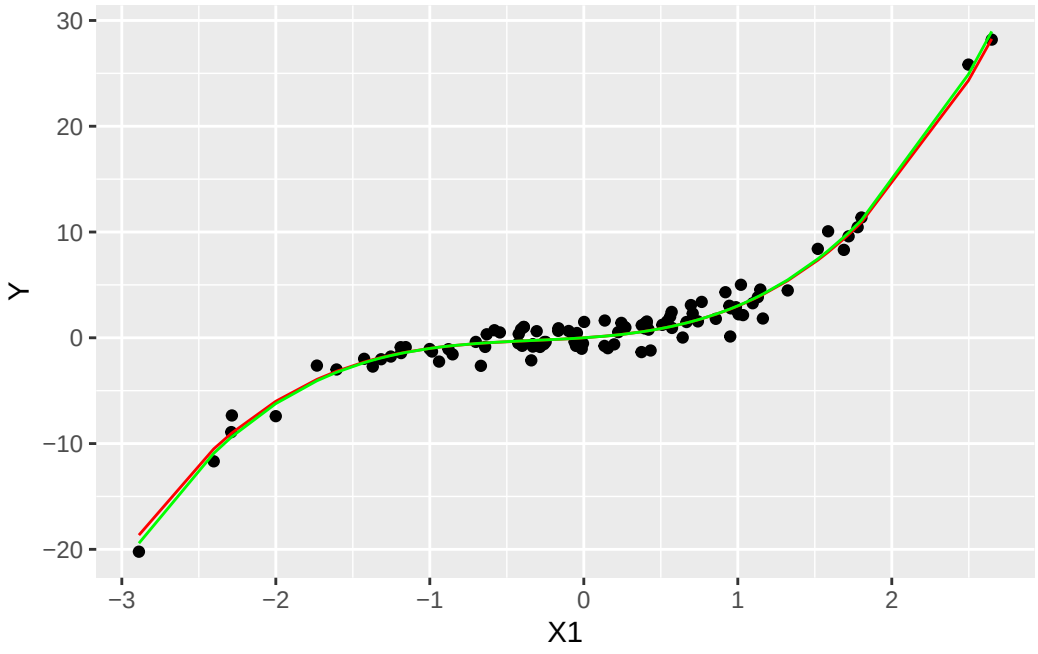
Vemos que con este criterio alcanza un mínimo en tres variables.

```
modelo_cp<- lm(y ~ X1 + X2 + X3  -1, data=datos)
summary(modelo_cp)
```

```
Call:
lm(formula = y ~ X1 + X2 + X3 - 1, data = datos)

Residuals:
     Min       1Q   Median       3Q      Max
-2.61090 -0.54234  0.02662  0.74564  2.00549

Coefficients:
   Estimate Std. Error t value Pr(>|t|)
X1  0.97666    0.16672   5.858 6.41e-08 ***
X2  1.00906    0.05287  19.087  < 2e-16 ***
```

```
X3  1.03788    0.03935  26.377  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9922 on 97 degrees of freedom
Multiple R-squared:  0.9688,    Adjusted R-squared:  0.9679
F-statistic:  1005 on 3 and 97 DF,  p-value: < 2.2e-16
```

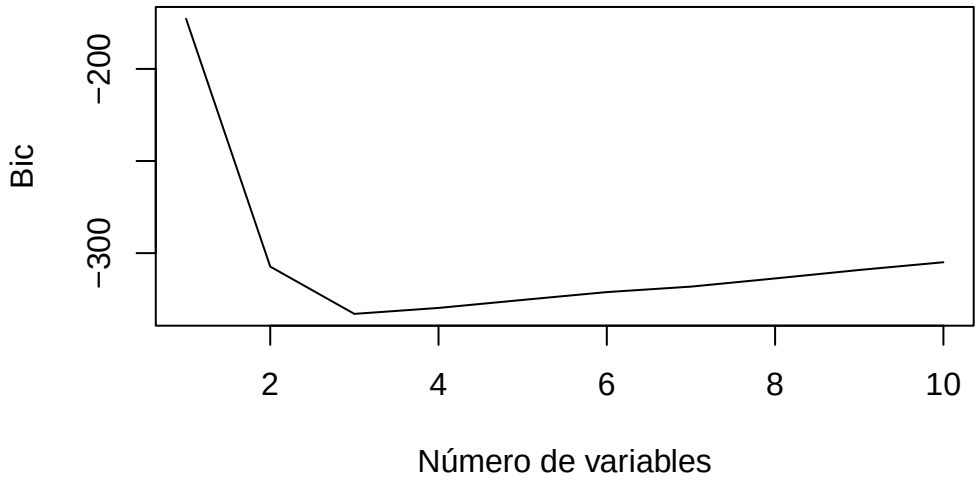
```r
y_predit_cp <- predict(object=modelo_cp, newdata=datos)

datos2$'Y_cp' <- y_predit_cp

p <- ggplot(datos2) +
  geom_point( aes(X1, Y), colour = 'black' ) +
  geom_line(aes(X1, Y_ideal), colour = 'red' ) +
  geom_line(aes(X1, Y_cp), colour = 'green' )

p
```



BIC

```
pinta_ajuste(resumen_2$bic, "Bic")
```

```
    x         y
1   1 -172.7782
2   2 -307.2928
3   3 -332.9790
4   4 -329.7446
5   5 -325.3963
6   6 -321.1518
7   7 -318.1865
8   8 -313.7402
9   9 -309.1533
10 10 -304.9534
```




```
cat("Con el criterio BIC, el mejor modelo tiene ", which.min(resumen_1$bic), "parametros\n
```

```
Con el criterio BIC, el mejor modelo tiene  3 parametros
```

```
summary(modelo_iterativo_delante)
```

```
Subset selection object
Call: regsubsets.formula(y ~ ., data = datos, method = "forward", nvmax = subconjuntos_a_exar
```

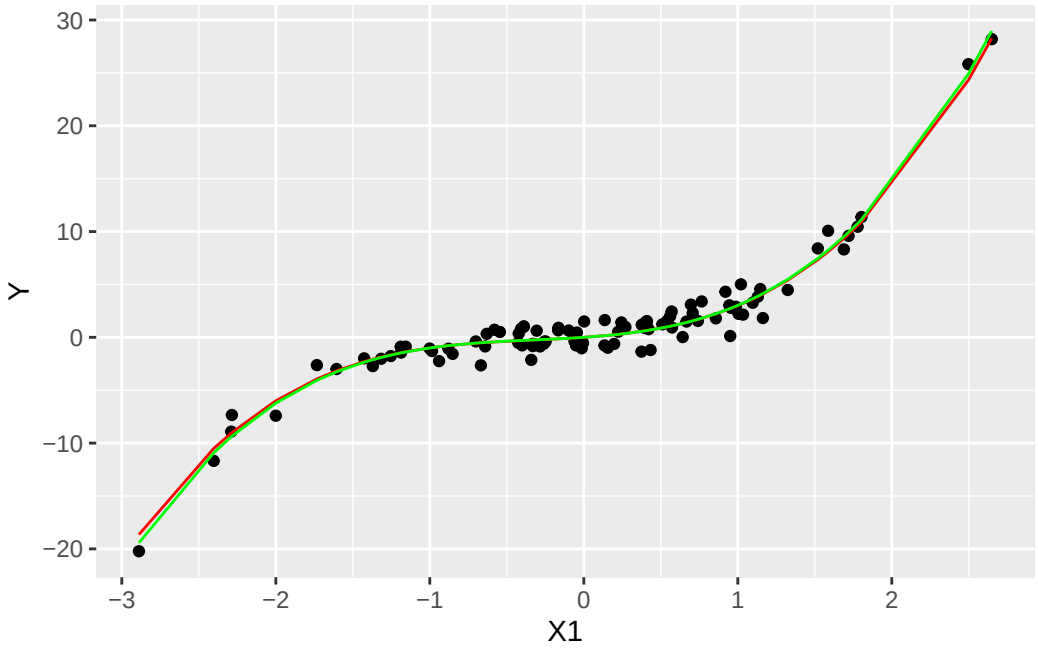
```
    intercept = FALSE)
10 Variables
    Forced in Forced out
X2      FALSE      FALSE
X3      FALSE      FALSE
X4      FALSE      FALSE
X5      FALSE      FALSE
X6      FALSE      FALSE
X7      FALSE      FALSE
X8      FALSE      FALSE
X9      FALSE      FALSE
X10     FALSE      FALSE
1 subsets of each size up to 10
Selection Algorithm: forward
          X1  X2  X3  X4  X5  X6  X7  X8  X9  X10
1  ( 1 )  " " " " "*" " " " " " " " " " " " " "
2  ( 1 )  " " "*" "*" " " " " " " " " " " " " "
3  ( 1 )  "*" "*" "*" " " " " " " " " " " " " "
4  ( 1 )  "*" "*" "*" " " "*" " " " " " " " " "
5  ( 1 )  "*" "*" "*" " " "*" " " " " " " " "*"
6  ( 1 )  "*" "*" "*" " " "*" " " "*" " " " " "*"
7  ( 1 )  "*" "*" "*" " " "*" "*" "*" " " " " "*"
8  ( 1 )  "*" "*" "*" " " "*" "*" "*" " " "*" "*"
9  ( 1 )  "*" "*" "*" "*" "*" "*" "*" " " "*" "*"
10  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" "*" "*"
```

```r
modelo_bic<- lm(y ~ X1 + X2 +  X3  -1, data=datos)
summary(modelo_bic)
```

```
Call:
lm(formula = y ~ X1 + X2 + X3 - 1, data = datos)

Residuals:
     Min       1Q   Median       3Q      Max
-2.61090 -0.54234  0.02662  0.74564  2.00549

Coefficients:
   Estimate Std. Error t value Pr(>|t|)
X1  0.97666    0.16672   5.858 6.41e-08 ***
X2  1.00906    0.05287  19.087  < 2e-16 ***
X3  1.03788    0.03935  26.377  < 2e-16 ***
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9922 on 97 degrees of freedom
Multiple R-squared:  0.9688,    Adjusted R-squared:  0.9679
F-statistic:  1005 on 3 and 97 DF,  p-value: < 2.2e-16
```

```r
y_predit_bic <- predict(object=modelo_bic, newdata=datos)

datos2$'Y_bic' <- y_predit_bic

p <- ggplot(datos2) +
  geom_point( aes(X1, Y), colour = 'black' ) +
  geom_line(aes(X1, Y_ideal), colour = 'red' ) +
  geom_line(aes(X1, Y_bic), colour = 'green' )

p
```




```r
print('Los coefficientes del modelo son')
```

```
[1] "Los coefficientes del modelo son"
```

```r
coef(modelo_bic)
```

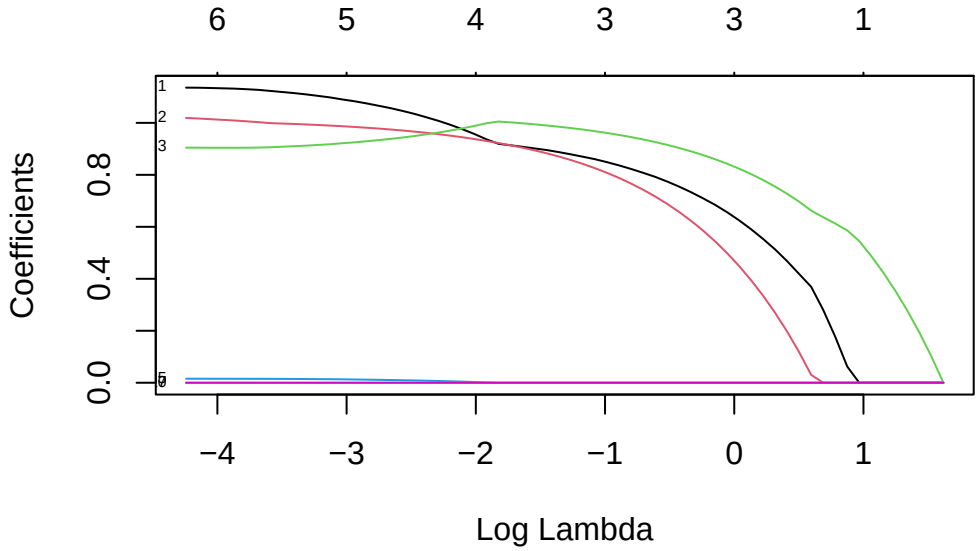
```
       X1        X2        X3
0.9766564 1.0090610 1.0378754
```

Lasso

```r
library(glmnet)
```

```
Loading required package: Matrix

Attaching package: 'Matrix'

The following objects are masked from 'package:ramify':

    tril, triu

Loaded glmnet 4.1-6
```

```r
x <- datos[c(1:10)]
y <- datos$y
modelo_lasso <- glmnet(x, y, alpha = 1, intercept=FALSE)   # alpha = 1 (lasso); alpha = 0
plot(modelo_lasso, xvar='lambda', label=TRUE)
```



```r
coef(modelo_lasso, s = 0.1)
```

```
11 x 1 sparse Matrix of class "dgCMatrix"
                    s1
(Intercept) .
X1          1.009813169
X2          0.957280286
X3          0.961825895
X4          .
X5          0.006891391
X6          .
X7          .
X8          .
X9          .
X10         .
```

```r
#help(coef)
```

La ventaja que tiene lasso es que permite eliminar ciertos coeficientes esto queda de manifiesto en que la mayoría son nulos.

Genera ahora las respuestas a partir del modelo

$$Y = X^7 + \epsilon.$$

Generación de los datos

```r
# Generamos las potencias
datos <- data.frame('1'=X)
for (i in 2:10) {
  datos[i] <- datos[1]*datos[i-1]
}
# Modelo1
Y <- unlist(
  as.list(
    datos[7] + error_distribution(n)
  )
)
datos$'y' <- Y
```

```r
colnames(datos) <- nombre_variables
```

Modelo óptimo dentro de submodelos

```r
modelo_exhaustivo <- leaps::regsubsets(
  y ~ .,
  data=datos,
  nvmax = subconjuntos_a_examinar,
  intercept = FALSE
  )
resumen_1 <- summary(modelo_exhaustivo)
resumen_1
```

```
Subset selection object
Call: regsubsets.formula(y ~ ., data = datos, nvmax = subconjuntos_a_examinar,
    intercept = FALSE)
10 Variables
    Forced in Forced out
X2      FALSE      FALSE
X3      FALSE      FALSE
X4      FALSE      FALSE
X5      FALSE      FALSE
X6      FALSE      FALSE
X7      FALSE      FALSE
X8      FALSE      FALSE
X9      FALSE      FALSE
X10     FALSE      FALSE
1 subsets of each size up to 10
Selection Algorithm: exhaustive
          X1  X2  X3  X4  X5  X6  X7  X8  X9  X10
1  ( 1 )  " " " " " " " " " " " " "*" " " " " " "
2  ( 1 )  " " " " " " " " "*" " " "*" " " " " " "
3  ( 1 )  "*" " " "*" " " " " " " " "*" " " " " " "
4  ( 1 )  " " " " "*" " " "*" " " "*" " " "*" " "
5  ( 1 )  " " " " "*" " " "*" "*" "*" " " "*" " "
6  ( 1 )  " " " " "*" "*" "*" " " "*" " " "*" "*"
7  ( 1 )  " " " " "*" " " "*" "*" "*" "*" "*" "*"
8  ( 1 )  " " "*" "*" "*" "*" "*" "*" "*" "*" " "
9  ( 1 )  "*" "*" "*" "*" "*" "*" "*" "*" "*" " "
10  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" "*" "*"
```

Vemos que ahora a pesar de estar en un método exhaustivo, X^7 (el que sabemos que forma parte del modelo ideal) se encuentra en todos. Esto puede entenderse como que al ser solo una *es más simple que la encuentre* y que no se compense con nada.

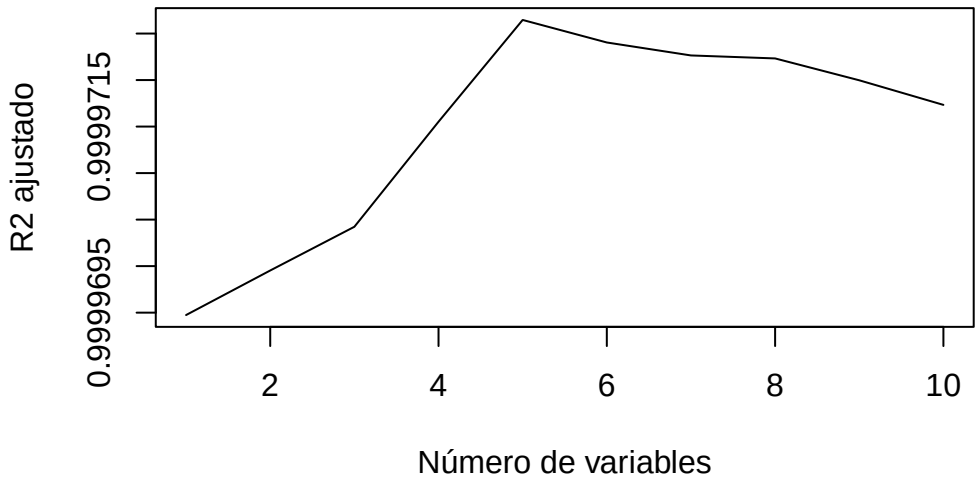
Submodelos óptimos

```
print("Submodelos óptimos")
```

```
[1] "Submodelos óptimos"
```

```
pinta_ajuste(resumen_1$adjr2, "R2 ajustado")
```

```
    x         y
1   1 0.9999695
2   2 0.9999700
3   3 0.9999704
4   4 0.9999716
5   5 0.9999726
6   6 0.9999724
7   7 0.9999723
8   8 0.9999722
9   9 0.9999720
10 10 0.9999717
```

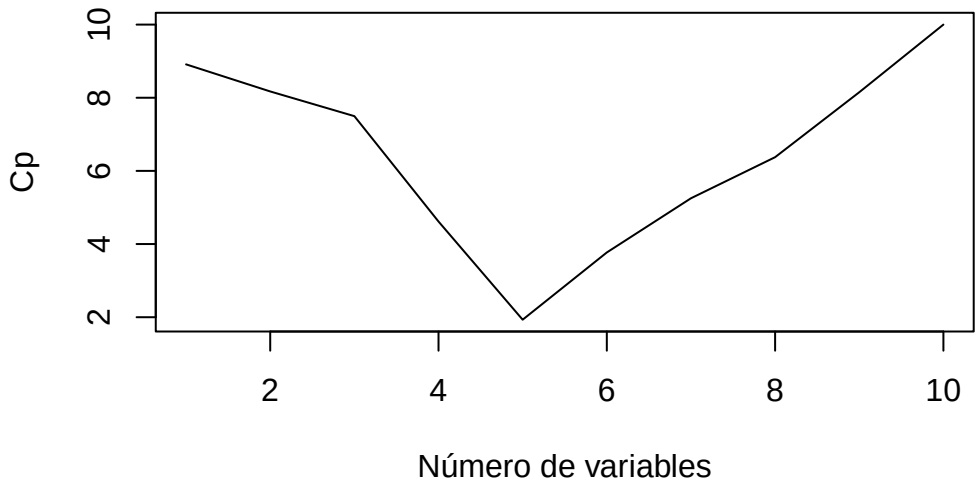


```
cat("Se alcanza el máximo con ", which.max(resumen_1$adjr2), "parametros")
```

```
Se alcanza el máximo con  5 parametros
```

```
pinta_ajuste(resumen_1$cp, "Cp")
```

```
    x         y
1   1  8.912823
2   2  8.172334
3   3  7.496707
4   4  4.615398
5   5  1.931619
6   6  3.770076
7   7  5.251341
8   8  6.375017
9   9  8.149811
10 10 10.000000
```

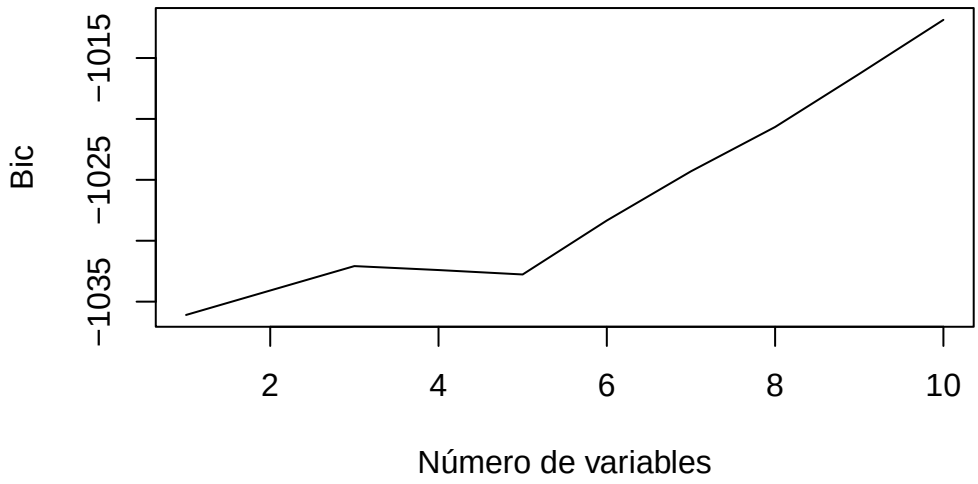



```
cat("Con el criterio CP, el mejor modelo tiene ", which.min(resumen_1$cp), "parametros\n")
```

```
Con el criterio CP, el mejor modelo tiene  5 parametros
```

```
pinta_ajuste(resumen_1$bic, "Bic")
```

```
    x         y
1   1 -1036.093
2   2 -1034.084
3   3 -1032.081
4   4 -1032.405
5   5 -1032.769
6   6 -1028.340
7   7 -1024.302
8   8 -1020.661
9   9 -1016.306
10 10 -1011.867
```




```
cat("Con el criterio BIC, el mejor modelo tiene ", which.min(resumen_1$bic), "parametros\n
```

```
Con el criterio BIC, el mejor modelo tiene  1 parametros
```

```
summary(modelo_exhaustivo)
```

```
Subset selection object
Call: regsubsets.formula(y ~ ., data = datos, nvmax = subconjuntos_a_examinar,
    intercept = FALSE)
10 Variables
    Forced in Forced out
X2      FALSE      FALSE
X3      FALSE      FALSE
```

```
X4      FALSE      FALSE
X5      FALSE      FALSE
X6      FALSE      FALSE
X7      FALSE      FALSE
X8      FALSE      FALSE
X9      FALSE      FALSE
X10     FALSE      FALSE
1 subsets of each size up to 10
Selection Algorithm: exhaustive
          X1  X2  X3  X4  X5  X6  X7  X8  X9  X10
1  ( 1 )  " " " " " " " " " " " " "*" " " " " " "
2  ( 1 )  " " " " " " " " "*" " " "*" " " " " " "
3  ( 1 )  "*" " " "*" " " " " " " " "*" " " " " " "
4  ( 1 )  " " " " "*" " " "*" " " "*" " " "*" " "
5  ( 1 )  " " " " "*" " " "*" "*" "*" " " "*" " "
6  ( 1 )  " " " " "*" "*" "*" " " "*" " " "*" "*"
7  ( 1 )  " " " " "*" " " "*" "*" "*" "*" "*" "*"
8  ( 1 )  " " "*" "*" "*" "*" "*" "*" "*" "*" " "
9  ( 1 )  "*" "*" "*" "*" "*" "*" "*" "*" "*" " "
10  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" "*" "*"
```

Submodelos utilizando un método iterativo

```
modelo_iterativo_delante <- leaps::regsubsets(
  y ~ .,
  data=datos,
  method = "forward",
  nvmax = subconjuntos_a_examinar,
  intercept = FALSE
  )
resumen_2 <- summary(modelo_iterativo_delante)
resumen_2
```

```
Subset selection object
Call: regsubsets.formula(y ~ ., data = datos, method = "forward", nvmax = subconjuntos_a_exar
    intercept = FALSE)
10 Variables
    Forced in Forced out
X2      FALSE      FALSE
X3      FALSE      FALSE
X4      FALSE      FALSE
```

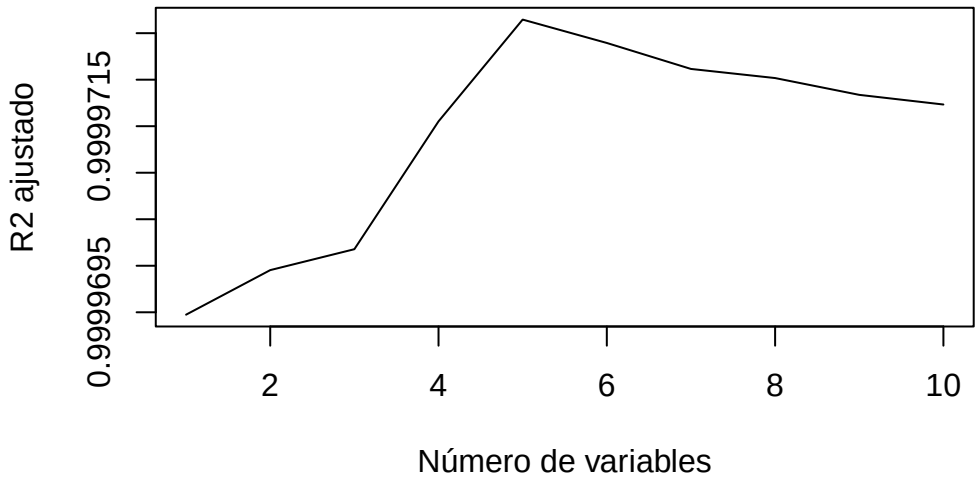
```
X5      FALSE      FALSE
X6      FALSE      FALSE
X7      FALSE      FALSE
X8      FALSE      FALSE
X9      FALSE      FALSE
X10     FALSE      FALSE
1 subsets of each size up to 10
Selection Algorithm: forward
          X1  X2  X3  X4  X5  X6  X7  X8  X9  X10
1  ( 1 )  " " " " " " " " " " " " "*" " " " " " "
2  ( 1 )  " " " " " " " " "*" " " "*" " " " " " "
3  ( 1 )  " " " " " " " " "*" " " "*" " " "*" " "
4  ( 1 )  " " " " "*" " " "*" " " "*" " " "*" " "
5  ( 1 )  " " " " "*" " " "*" "*" "*" " " "*" " "
6  ( 1 )  "*" " " "*" " " "*" "*" "*" " " "*" " "
7  ( 1 )  "*" " " "*" " " "*" "*" "*" " " "*" "*"
8  ( 1 )  "*" " " "*" " " "*" "*" "*" "*" "*" "*"
9  ( 1 )  "*" "*" "*" " " "*" "*" "*" "*" "*" "*"
10  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" "*" "*"
```

```
print("Submodelos utilizando el método iterativo \n")
```

```
[1] "Submodelos utilizando el método iterativo \n"
```

```
pinta_ajuste(resumen_2$adjr2, "R2 ajustado")
```

```
    x         y
1   1 0.9999695
2   2 0.9999700
3   3 0.9999702
4   4 0.9999716
5   5 0.9999726
6   6 0.9999724
7   7 0.9999721
8   8 0.9999720
9   9 0.9999718
10 10 0.9999717
```

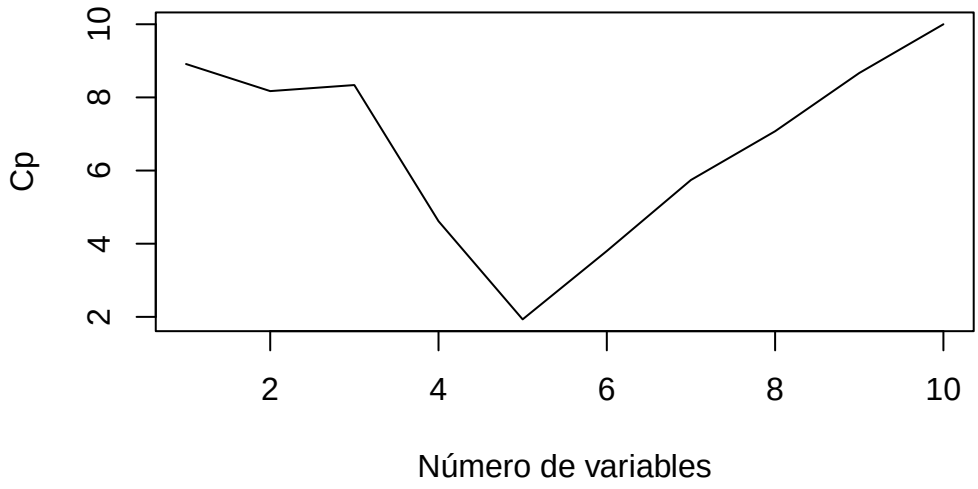



```
cat("Se alcanza el máximo con ", which.max(resumen_2$adjr2), "parametros")
```

```
Se alcanza el máximo con  5 parametros
```

```
pinta_ajuste(resumen_2$cp, "Cp")
```

```
    x         y
1   1  8.912823
2   2  8.172334
3   3  8.337723
4   4  4.615398
5   5  1.931619
6   6  3.800172
7   7  5.741825
8   8  7.075010
9   9  8.666075
10 10 10.000000
```

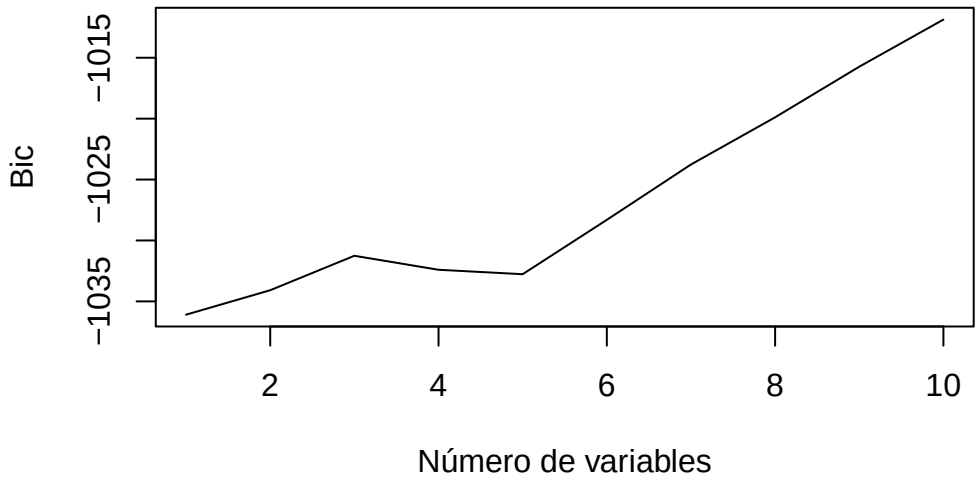



```
cat("Con el criterio CP, el mejor modelo tiene ", which.min(resumen_2$cp), "parametros\n")
```

```
Con el criterio CP, el mejor modelo tiene  5 parametros
```

```
pinta_ajuste(resumen_2$bic, "Bic")
```

```
    x         y
1   1 -1036.093
2   2 -1034.084
3   3 -1031.256
4   4 -1032.405
5   5 -1032.769
6   6 -1028.307
7   7 -1023.765
8   8 -1019.890
9   9 -1015.735
10 10 -1011.867
```




```
cat("Con el criterio BIC, el mejor modelo tiene ", which.min(resumen_1$bic), "parametros\n
```

```
Con el criterio BIC, el mejor modelo tiene  1 parametros
```

```
summary(modelo_iterativo_delante)
```

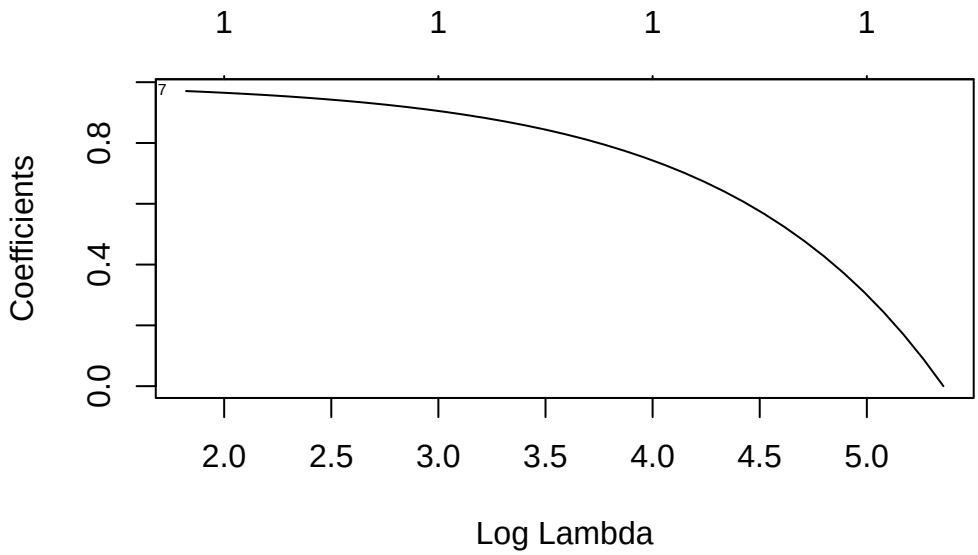
```
Subset selection object
Call: regsubsets.formula(y ~ ., data = datos, method = "forward", nvmax = subconjuntos_a_exa
    intercept = FALSE)
10 Variables
    Forced in Forced out
X2      FALSE      FALSE
X3      FALSE      FALSE
X4      FALSE      FALSE
X5      FALSE      FALSE
X6      FALSE      FALSE
X7      FALSE      FALSE
X8      FALSE      FALSE
X9      FALSE      FALSE
X10     FALSE      FALSE
1 subsets of each size up to 10
Selection Algorithm: forward
          X1  X2  X3  X4  X5  X6  X7  X8  X9  X10
1  ( 1 )  " " " " " " " " " " " " "*" " " " " " "
2  ( 1 )  " " " " " " " " "*" " " "*" " " " " " "
3  ( 1 )  " " " " " " " " "*" " " "*" " " "*" " "
```

```
4  ( 1 )  " " " " "*" " " "*" " " "*" " " "*" " "
5  ( 1 )  " " " " "*" " " "*" "*" "*" " " "*" " "
6  ( 1 )  "*" " " "*" " " "*" "*" "*" " " "*" " "
7  ( 1 )  "*" " " "*" " " "*" "*" "*" " " "*" "*"
8  ( 1 )  "*" " " "*" " " "*" "*" "*" "*" "*" "*"
9  ( 1 )  "*" "*" "*" " " "*" "*" "*" "*" "*" "*"
10  ( 1 ) "*" "*" "*" "*" "*" "*" "*" "*" "*" "*"
```

Lasso

```
x <- datos[c(1:10)]

y <- datos$y
modelo_lasso <- glmnet(x, y, alpha = 1, intercept=FALSE)   # alpha = 1 (lasso); alpha = 0
plot(modelo_lasso, xvar='lambda', label=TRUE)
```

```
Warning in plotCoef(x$beta, lambda = x$lambda, df = x$df, dev = x$dev.ratio, : 1
or less nonzero coefficients; glmnet plot is not meaningful
```




```
coef(modelo_lasso, s = 0.1)
```

```
11 x 1 sparse Matrix of class "dgCMatrix"
                    s1
```

```
(Intercept) .
X1          .
X2          .
X3          .
X4          .
X5          .
X6          .
X7          0.9708107
X8          .
X9          .
X10         .
```

```
#help(coef)
```

Conclusiones

A la vista de los resultados el modelo BIC es el más robusto a la hora de determinar el número de parámetros correcto. Lasso es una alternativa razonable para reducir el número de variables y minimizar el error.

Ejercicio 12

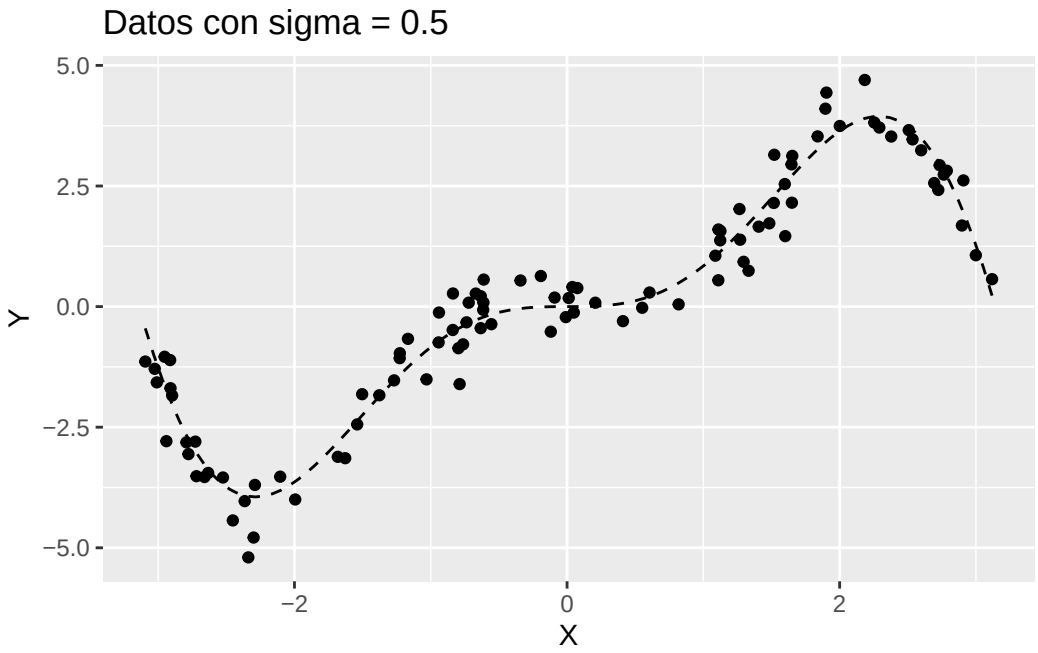
Se generan los siguientes dos conjuntos de datos:

```
library(purrr)
library(ggplot2)
```

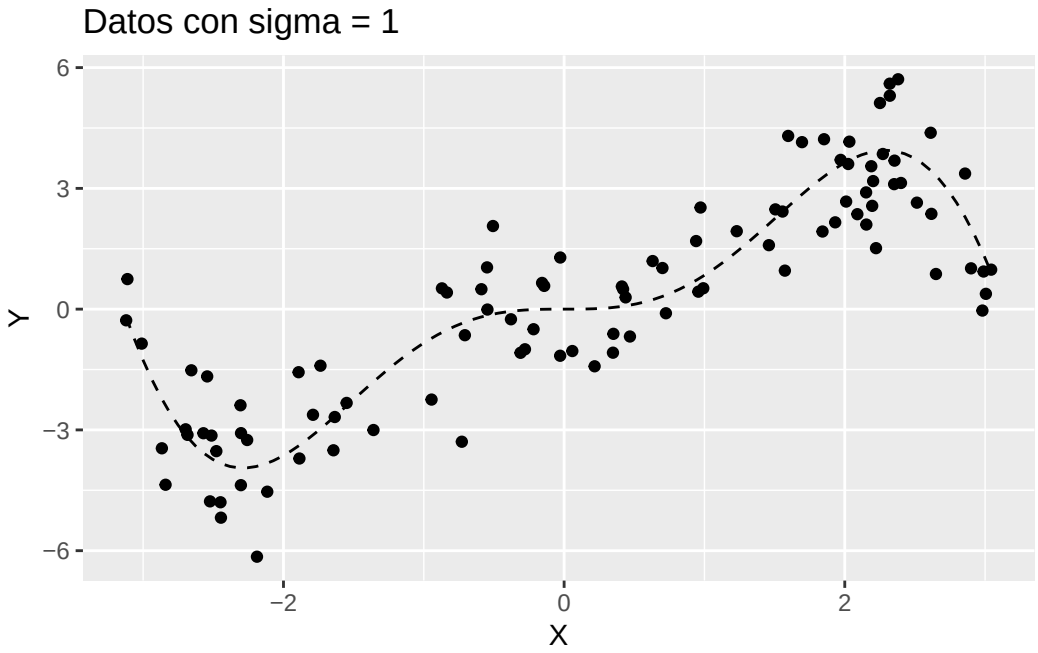
```
n <- 100
sigma_1 <- 0.5
sigma_2 <- 1

fun_reg <- function (x) (x^2*sin(x))

generator <- function (n, sigma) {
  error <- rnorm(n)
  unif <- runif(n, min = -pi, max = pi)
  Y <- (
    unif*unif
    *
    sapply(unif, sin)
    +
    sigma * error
  )

  return (
    data.frame(
      X= unif,
      Y = Y
    )
  )

}
X_1 <- generator(n, sigma_1)
X_2 <- generator(n, sigma_2)
```

Representación gráfica

```
p <- ggplot(X_1) +
  geom_point( aes(X, Y), colour = 'black' )+
  ggtitle('Datos con sigma = 0.5') +
  geom_function(fun = 'fun_reg', linetype = 2)

q <- ggplot(X_2) +
  geom_point( aes(X, Y), colour = 'black' )+
  ggtitle('Datos con sigma = 1') +
  geom_function(fun = 'fun_reg', linetype = 2)
p
```




```
q
```



Puede verse como el sigma aumenta la dispersión respecto al eje Y.

Ajuste de regresión de mínimos cuadrados

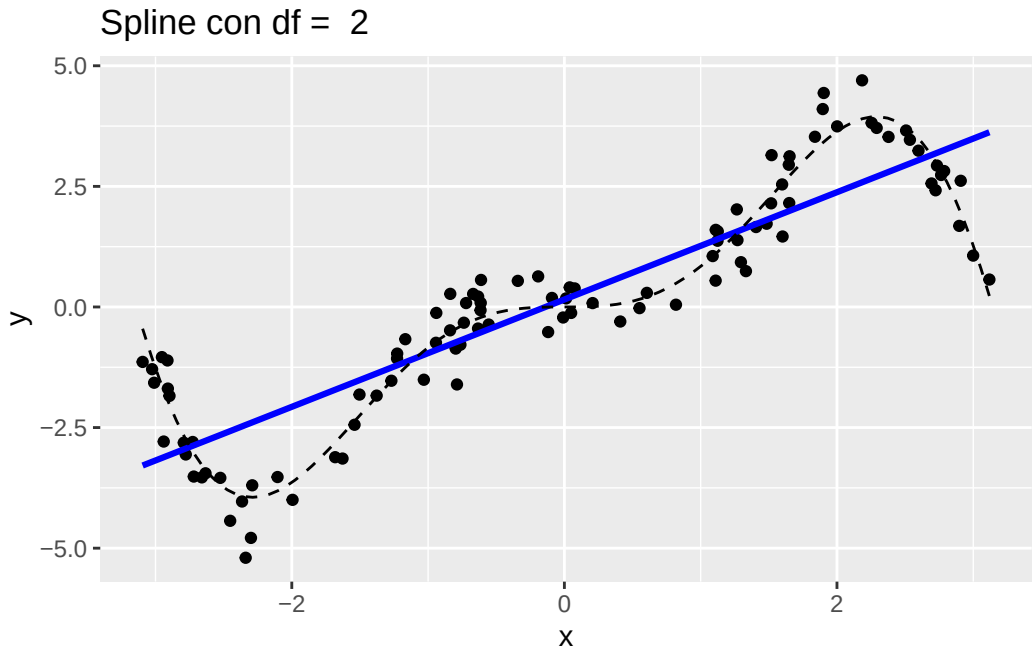
Para cada conjunto de datos se pretende ajustar una regresión de mínimos cuadrados penalizada prefijando uno de los grados de libertad efectivos.

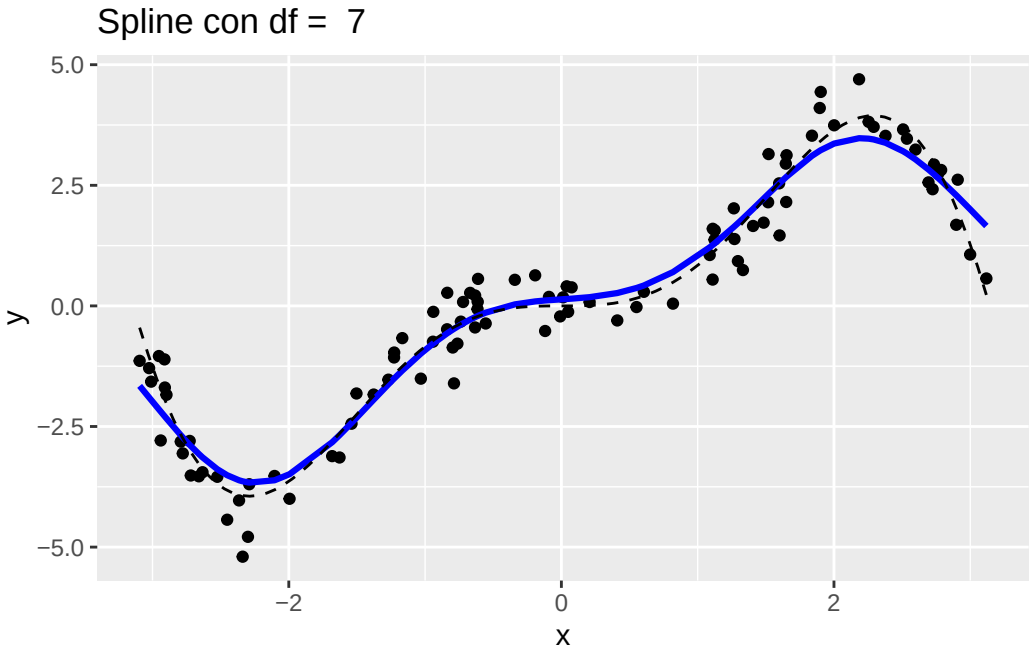
Representación gráfica del error de predicción de validación cruzada generalizando en función de los grados de liberta utilizados.

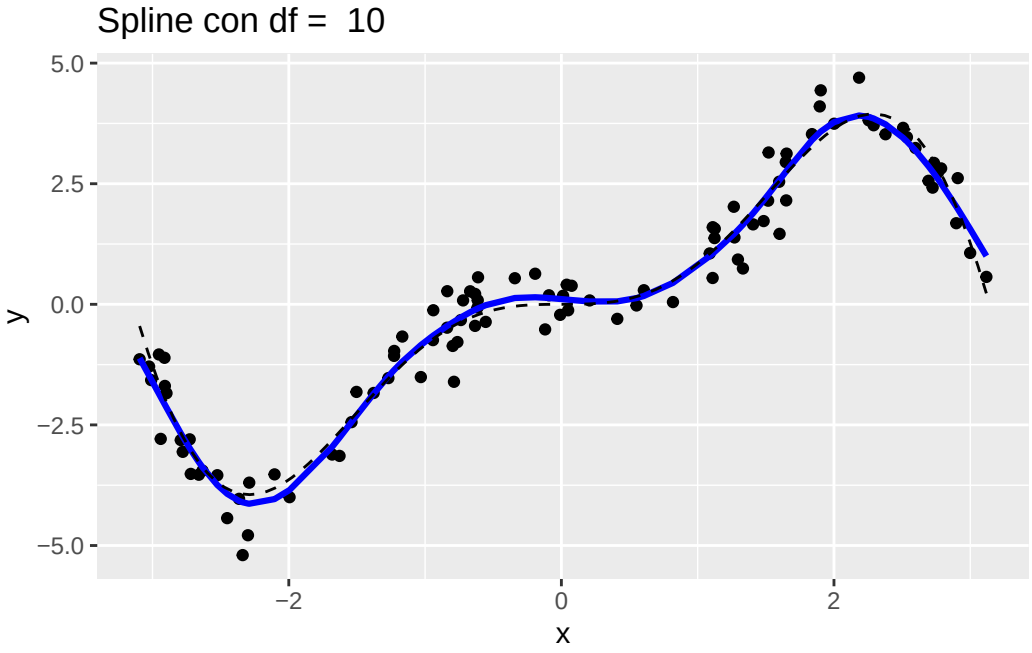
Imprimimos primero algunos ajustes:

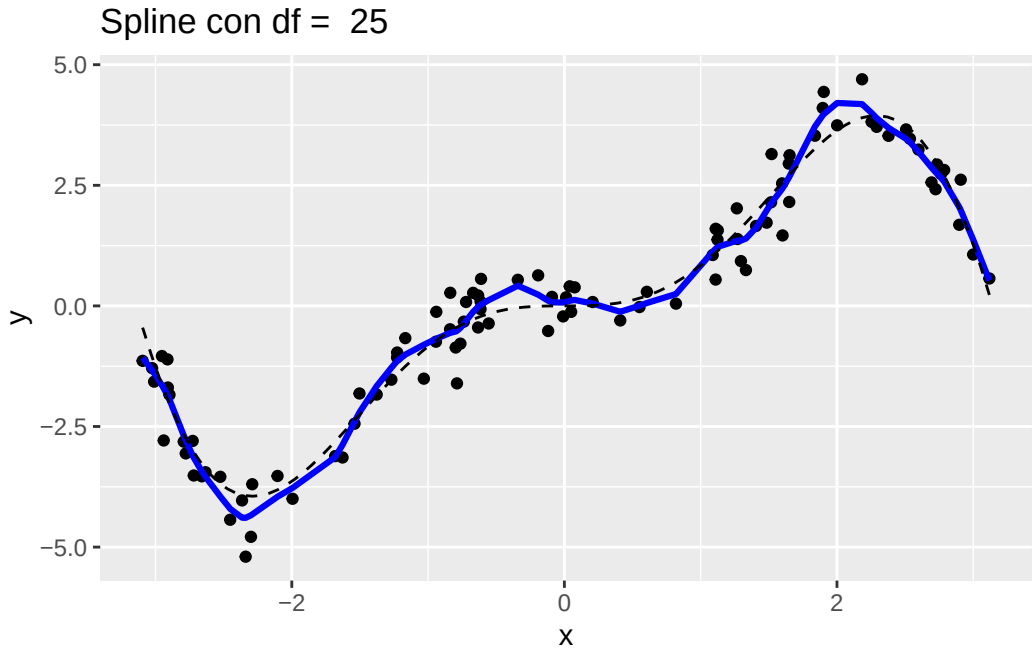
```
pinta_spline <- function (X, grados_libertad){
spline_1 <- smooth.spline(
  X$X, X$Y, df = grados_libertad
)
datos <- data.frame(
  x = X$X,
  y = X$Y,
  xfit = spline_1$x,
  yfit = spline_1$y
```

```
)

titulo <- paste('Spline con df = ', grados_libertad)
ggplot(datos) +
  geom_point(aes(x, y)) +
  geom_line(aes(xfit, yfit), color="blue", size = 1.1) +
  geom_function(fun = 'fun_reg', linetype = 2) +
  ggtitle(titulo)
}

for( df in c(2,7,10, 25) ){
  print(pinta_spline(X_1, df))
}
```



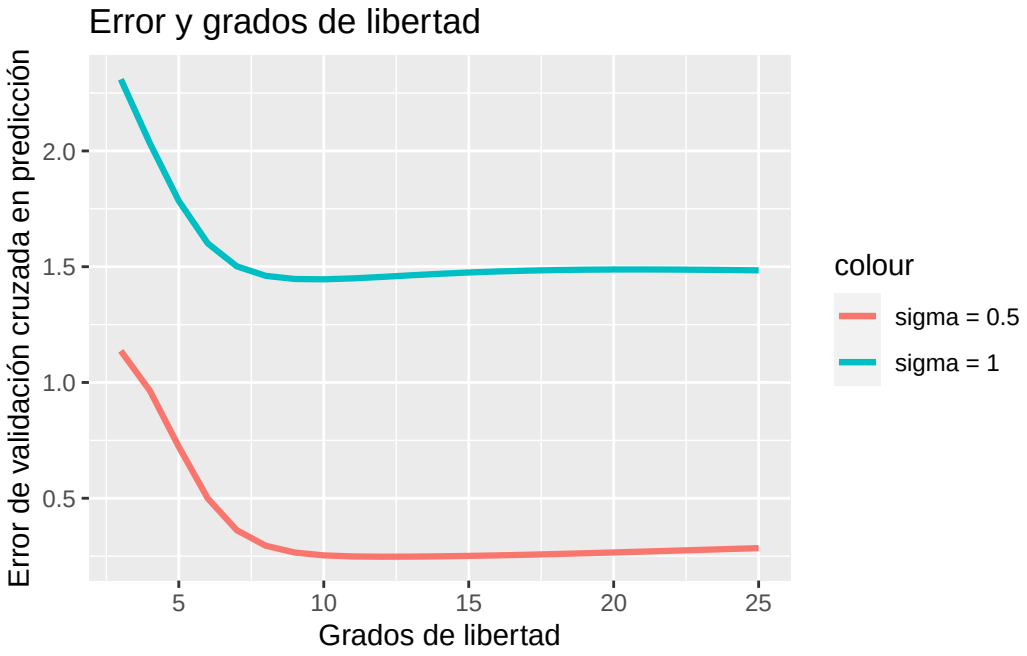

Spline con df = 7
5.0
2.5
0.0
−2.5
−5.0
y
−2
0
2
x


Spline con df = 10
5.0
2.5
0.0
−2.5
−5.0
y
−2
0
2
x



Bajo una inspección visual puede verse que a mayor número de grados de libertad mejor es el ajuste a los datos utilizados en regresión c, procedamos a constatarlo

```
cv_error <- function (X, grados_libertad){
  spline_1 <- smooth.spline(
    X$X, X$Y, df = grados_libertad
  )
  return (spline_1$cv.crit)
}

df <- c(3:25)
datos <- data.frame(
  x = df,
  y1 = sapply(df, function(d)(cv_error(X_1, d))),
  y2 = sapply(df, function(d)(cv_error(X_2, d)))
)

ggplot(datos) +
  geom_line(aes(x, y1, color="sigma = 0.5"), size = 1.1) +
  geom_line(aes(x, y2, color="sigma = 1") , size = 1.1) +
  labs(title = 'Error y grados de libertad',
       x = 'Grados de libertad',
       y = 'Error de validación cruzada en predicción')
```




```
cat("Pasa sigma = 0.5 alcanza un mínimo en ", datos$x[which.min(datos$y1)], " grados de li
```

```
Pasa sigma = 0.5 alcanza un mínimo en  12  grados de libertad.
```

```
cat("Pasa sigma = 1 alcanza un mínimo en ", datos$x[which.min(datos$y2)], " grados de libe
```

```
Pasa sigma = 1 alcanza un mínimo en  10  grados de libertad.
```

Comentario de los resultados

Hay varios fenómenos llamativos al observa esta gráfica: 1. Comportamiento general: Decrece, alcanza un mínimo y vuelve a crecer en menor medida. 2. El que los dos gráficos parezcan más o menos desplazados. 3. Que σ menor admita más grados de libertad antes de volver a crecer el error. 4. Que la gráfica de $\sigma = 1$ comience a crecer más que la $\sigma = 0.5$.

Todos estos fenómenos se pueden explicar con la relación entre los errores vista en teoría:

$$\text{Error}_{Test} = \text{Error}_{Training} + \frac{2\sigma^2}{n}\text{grados de libertad.}$$

(Notemos que grados de libertad se corresponde a la traza de la matriz M, que en nuestro caso particular se trata de los mínimos cuadrados)

Comportamiento general: Decrecemiento, mínimo y crecimiento más lento

Como rasgo general podemos observar como añadir más grados de libertad mejora el ajuste a los datos de *aprendizaje*, el error de *training* está decreciendo en mayor medida que el aumento de grados de libertad. A partir de 10 (o 13 si $\sigma = 1$) grados puede verse cómo el error de test comienza a crecer de nuevo, este fenómeno conocido como *sobreajuste a los datos de entrenamiento* no es más que el error de training se disminuye en menor proporción que el peso que suma el término $\frac{2\sigma^2}{n}$grados de libertad.

Gráficos desplazados

Esto está motivado por que los *datos training* de uno es mayor que la de otro. Este fenómenos es natural, ya que por la propia naturaleza de los datos sabemos que los datos de $\sigma = 1$ (la gráfica con mayor error) posee un ruido mayor (concretamente el de $\mathcal{N}(0, 1)$)

A σ menor admita más grados de libertad antes de volver a crecer el error.

Relacionado con lo anterior, si el *ruido* es menor será más similar a su desarrollo de Taylor admitiendo por tanto desarrollar más términos de la serie de Taylor (grados de libertad en nuestro caso).

Crecimiento mayor del error de $\sigma = 1$ frente a $\sigma = 0.5$

Esto es claro resultado del segundo sumando de la relación mostrada:

$$\frac{2\sigma^2}{n}\text{grados de libertad}$$

ya que en ambos caso n y *grados de libertad* son iguales y solo difiere el valor de σ.