PEC_8 Aproximación de funciones y regresión (II)

Alan Coila Bustinza

2022-05-15

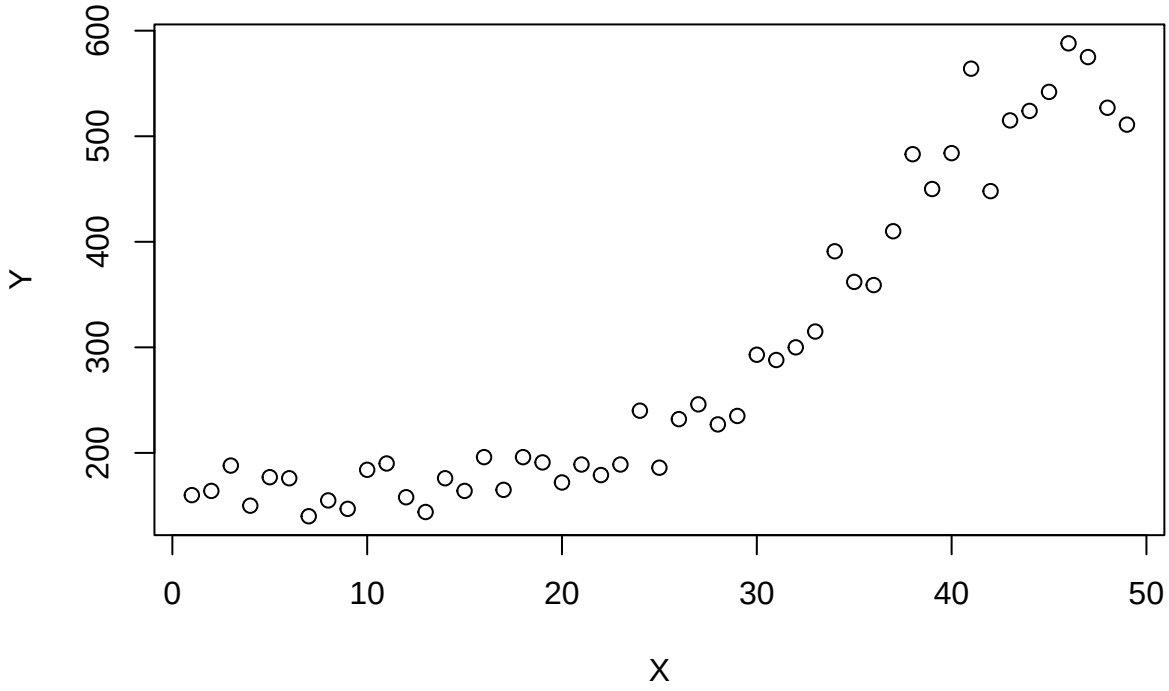
```
source("Lectura_datos_por_fecha.R")
```

1. Modelización de la Mortalidad

1.1. Modelización exponencial

Graficaremos los datos de los rangos de fecha solicitados, para valorar el tipo de distribución de los mismos.
La función que usamos proviene del archivo cargado previamente "Lectura_datos_por_fecha.R"

```
Y <- myReadData_byDate("WHO-COVID-19-global-data-SPAIN.csv", "15/12/2020", "01/02/2021", "New_deaths")
m <- length(Y)
X <- 1:m
plot(X,Y)
```



Vemos que se trata probablemente de una distribución exponencial. Para poder hacer un ajuste lineal de los datos podemos aplicar una transformación, mediante la aplicación del logaritmo a la ecuación:

$$f(x) = ae^{bx} \tag{1}$$

$$log(f(x) = log(a) + bx \tag{2}$$

Ahora crearemos la función auxiliar *Phi* que nos generará la matriz de diseño,esta nos permite crear un polinomio de ajuste, que se deduce segun la siguiente ecuación:

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 & \dots & x_1^m \\ 1 & x_2 & x_2^2 & \dots & x_2^m \\ 1 & x_3 & x_3^2 & \dots & x_3^m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \dots & x_n^m \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_m \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \vdots \\ \varepsilon_n \end{bmatrix},$$

Creamos la matriz de diseño con el siguiente codigo:

```
# Contrucciónde la matriz de los datos evaluados en las funciones base (Phi)
myPhi <- function(x, n) {
  Phi <- matrix(1, length(x), n + 1)
  for (i in 1:n) {
    Phi[, i + 1] <- x^i # funciones base para el ajuste polinómico, segun el grado
  }
  return(Phi)
}
```

Sabemos que a partir de la ecuación anterior, los coeficientes de la funcion de aproximación polinomial se obtienen del siguiente vector, donde X s la matriz de diseño:

$$\widehat{\vec{\beta}} = (\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\,\mathbf{X}^\mathsf{T}\vec{y},$$

```
mylssolve <- function(A, y) {
  AT <- t(A)
  return((solve(AT %*% A)) %*% AT %*% y) # la función solve nos devuelve la inversa de la matriz
}

mypolyfit <- function(x, y, n) {
  Phi <- myPhi(x, n) # construimos la matriz con myPhi
  c <- mylssolve(Phi, y) # resolvemos el sistema de ecuaciones normales con la función mylssolve
  return(c)
}
```

Ahora aplicaremos las funciones a nuestros datos, donde la siguiente función nos devuelve los coficientes de la ecuación exponencial, teniendo en cuenta $log(f(x)) = log(a) + bx$

```
Clog <- mypolyfit(X, log(Y), 1) # aplicamos el logaritmo a los valores de Y
Clog
```

```
##             [,1]
## [1,] 4.77862360
## [2,] 0.03113333
```

Ahora evaluaremos el ajuste polinómico mediante la función auxiliar myeval, dados los coeficientes obtenidos previamente teniendo en cuenta que $log(a)$ -> $a = exp(log(a))$

```
myeval <- function(x, c) {
  f <- 0
  for (i in 1:length(c)) {
    f <- f + c[i] * x^(i - 1)
  }
  return(f)
}
# recordemos que segun lo explicado anteriormente tenemos que aplicar la función exp
fLog <- exp(myeval(X, Clog))
```

Planteo una funcion exponencial ya que, como sabemos en caso de los fenomenos infecciosos,a medida que el contagio crezca entre la población, esta enfermedad tendra la capacidad de llegar a más y más personas, expresando un crecimiento correspondiente con esta función.

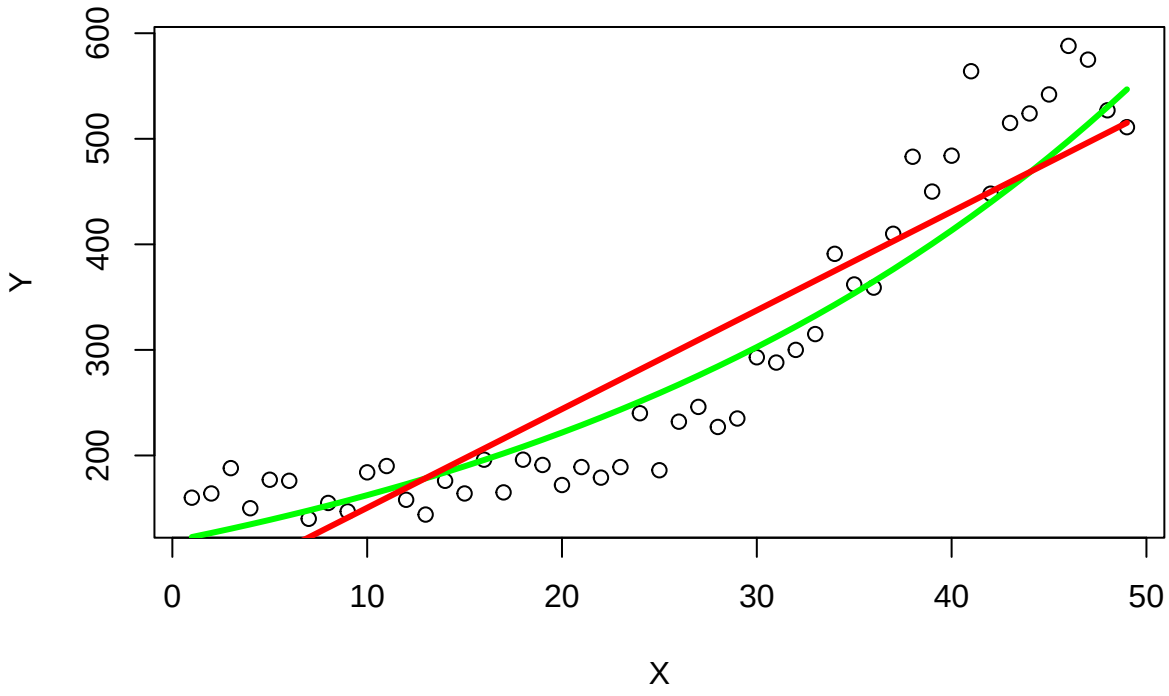
1.2. Ajuste mediante Regresión Lineal

Con nuestras funciones auxiliares anteriores podemos crear rapidamente un ajuste de regresión de grado uno

```
# Regresión lineal de grado 1
CkL <- mypolyfit(X, Y, 1) # obtenemos los coeficientes para una regresión de grado uno
fL <- myeval(X, CkL)
```

1.3. Representación grafica de ambos ajustes, el exponencial y el lineal

```
plot(X, Y)
lines(X, fLog, col = "green", lwd = 3)
lines(X, fL, col = "red", lwd = 3)
```



Como habiamos planetado anteriormente el ajuste exponencial, para predecir mejor que el lineal para el tipo de distribucion de nuestros datos.

1.4. Cálculo del coeficiente de correlación r

El calculo del coeficiente de correlación se realiza segun la ecuación:

$$r = \sqrt{\frac{\sum_{i=1}^{n} (y_i - \bar{y})^2 - \sum_{i=1}^{n} (\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2}}$$

```
# Cálculo de  discrepancias de los datos con respecto a la media
St <- sum((Y - mean(Y))^2)
# Sr_L = # sumatorio de los residuos para el ajuste exponencial
Sr_L <- sum((Y - exp(myeval(X, Clog)))^2)
# Sr_P = # sumatorio de los residuos para el ajuste lineal
Sr_P <- sum((Y - myeval(X, CkL))^2)
# r_L = # coeficiente de correlación para el ajuste exponencial
r_L <- (St - Sr_L) / St
```

```r
# r_P = # coeficiente de correlación para el ajuste lineal
r_P <- (St - Sr_P) / St
print(r_L)
```

```
## [1] 0.9006119
```

```r
print(r_P)
```

```
## [1] 0.8293
```

Vemos que el coeficiente de correlación para el ajute exponenecial es de 0.9006119 , mientras que para el ajuste lineal es de 0.8293, por lo que corroboramos que el model exponencial explica mejor el comportamiento de nuestros datos.

1.5. Predecir el numero de muertes

Simplemente tenemos que reemplazar el valor deseado en nuestro modelo.

```r
x15 <- length(X)+15
y15_L <- exp(myeval(x15, Clog))
y15_P <- myeval(x15, CkL)
print(y15_L)
```

```
## [1] 872.3206
```

```r
print(y15_P)
```

```
## [1] 655.4598
```

Segun el modelo exponencial se alcanzarian un numero de 872.3205559 muertes al dia 45, mientras que con el modelo lineal 655.4597959. Si siguiesemos este ultimo modelo, veriamos que podriamos estar subestimando la enfemerdad en cuestión y no tomar las medidas adecuadas para controlar la velocidad de contagio o prepararse para la atención de enfermos.

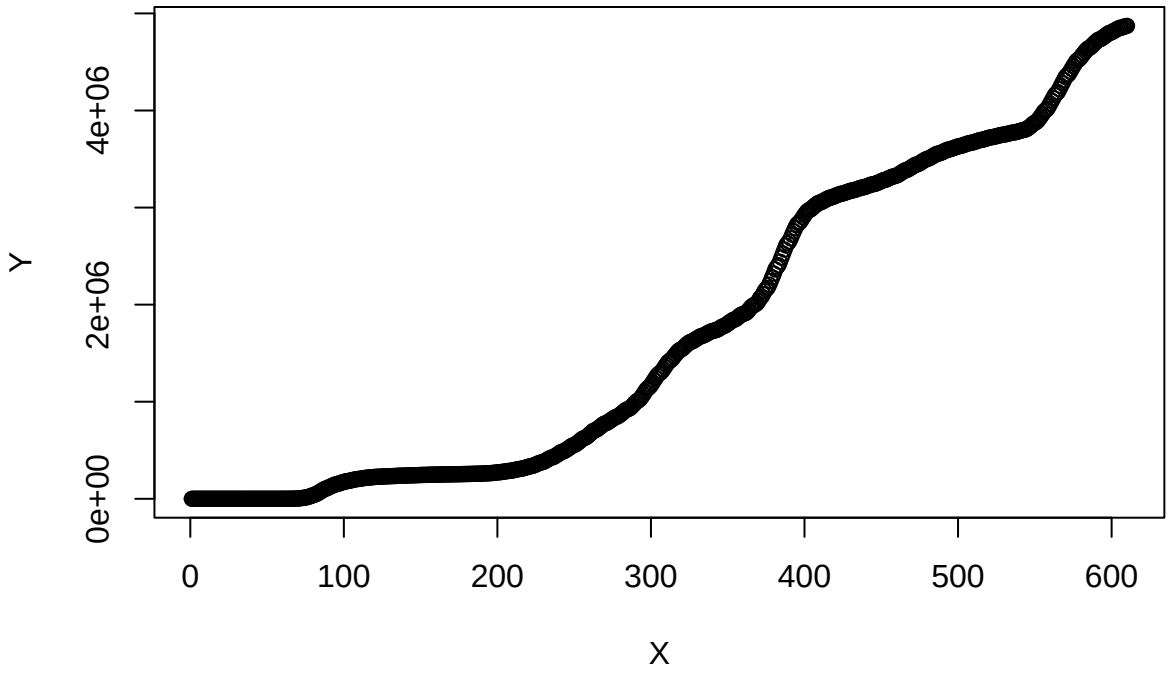
2. Detectar los cambios de tendencia

2.1 Regresión Segmentada

Mediante este metodo, la variable independiente se divide en intervalos y se ajusta un segmento de línea separado a cada intervalo. De tal forma:

$$
\begin{aligned}
Ys_1 = a_1 + b_1 x \quad para \quad x_1 < punto \quad de \quad corte \quad 1 \\
Ys_2 = a_2 + b_2 x \quad para \quad x_1 > punto \quad de \quad corte \quad 1 \\
\vdots \\
Ys_n = a_n + b_n x \quad para \quad x_n < punto \quad de \quad corte \quad n \\
Ys_{n+1} = a_{n+1} + b_{n+1} x \quad para \quad x_n > punto \quad de \quad corte \quad n
\end{aligned}
$$

```r
# Comenzamos con  la lectura de datos para la fecha solicitada
Y <- myReadData_byDate("WHO-COVID-19-global-data-SPAIN.csv", "03/01/2020", "03/09/2021", "Cumulative_cas
m <- length(Y)
X <- 1:m # definir la variable que representa a los das
plot(X, Y)
```



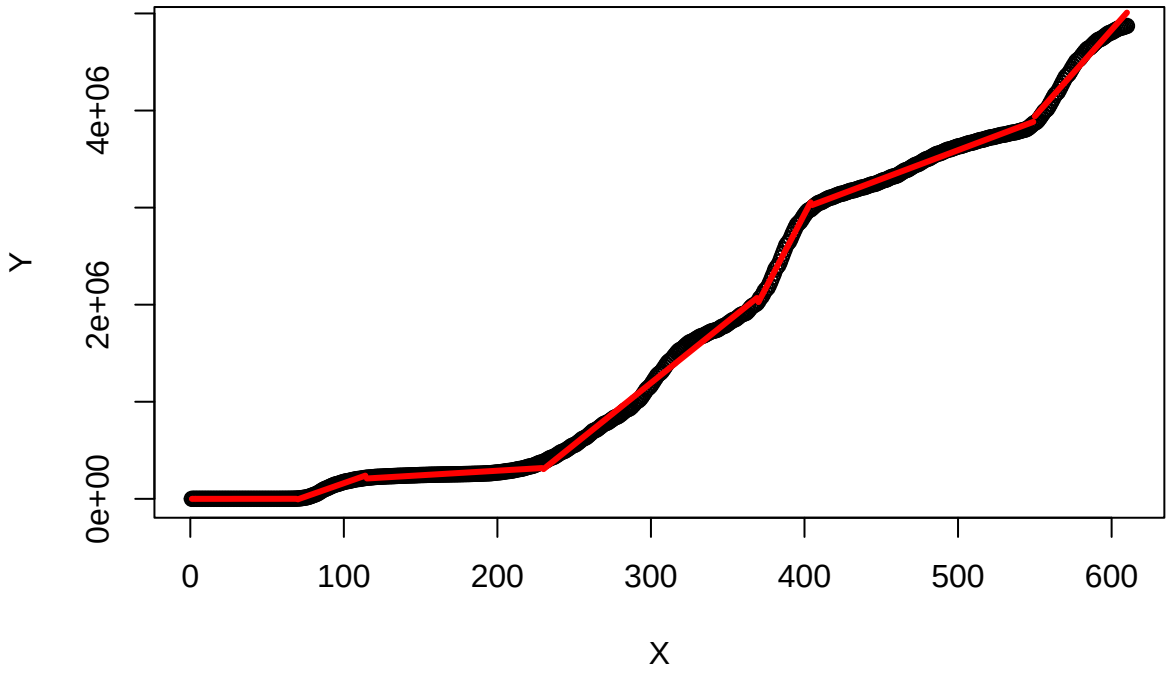

```r
# Obtención y representación de cada uno de los segmentos (en este caso se utilizan 7 segmentos)
seg <- 7 # numero de segmentos empleados
# designamos lo valores que pueden constituir el cambio de tendencia, de forma visual
ini_segmentos <- c(1, 70, 115, 230, 370, 405, 550)
# obtenemos los puntos donde finalizaran los segmentos
fin_segmentos <- c(ini_segmentos[2:length(ini_segmentos)] - 1, 610)
```

2.2 Representar gráficamente la regresión segmentada

Usaremos un bucle for para realizar un ajuste lineal de cada segmento de puntos que obtuvimos previamente, que almacenaremos dentro de la matriz Cks. Posterior a ello evaluaremos cada una de las funciones mediante la función auxiliar myeval(), que correspondera a los valores predichos por el ajuste.

```r
# creamos la matrix de 2 x nro de segmentos para almacenar los coeficientes que obtendremos en cada ite
Cks <- matrix(0, 2, seg)
plot(X, Y)
for (i in 1:seg) { # bucle en segmentos
  s <- ini_segmentos[i]:fin_segmentos[i]
```

```
  # dias de cada segmento
  Xs <- s
  # datos de cada segmento segun los dias
  Ys <- Y[Xs]
  # realizamos el ajuste lineal y almacenamos los coeficientes en cada columna de la matriz Cks
  Cks[, i] <- mypolyfit(Xs, Ys, 1)
  # graficamos las lineas por intervalos
  lines(Xs, myeval(Xs, matrix(Cks[, i])), col = "red", lwd = 3)
}
```



Vemos que la regresión segementada propuesta se aproxima bastante a los cambios de tendencia de la curva de casos acumulados por COVID 19. Las ecuaciones que rigen estos segmentos lineales son :

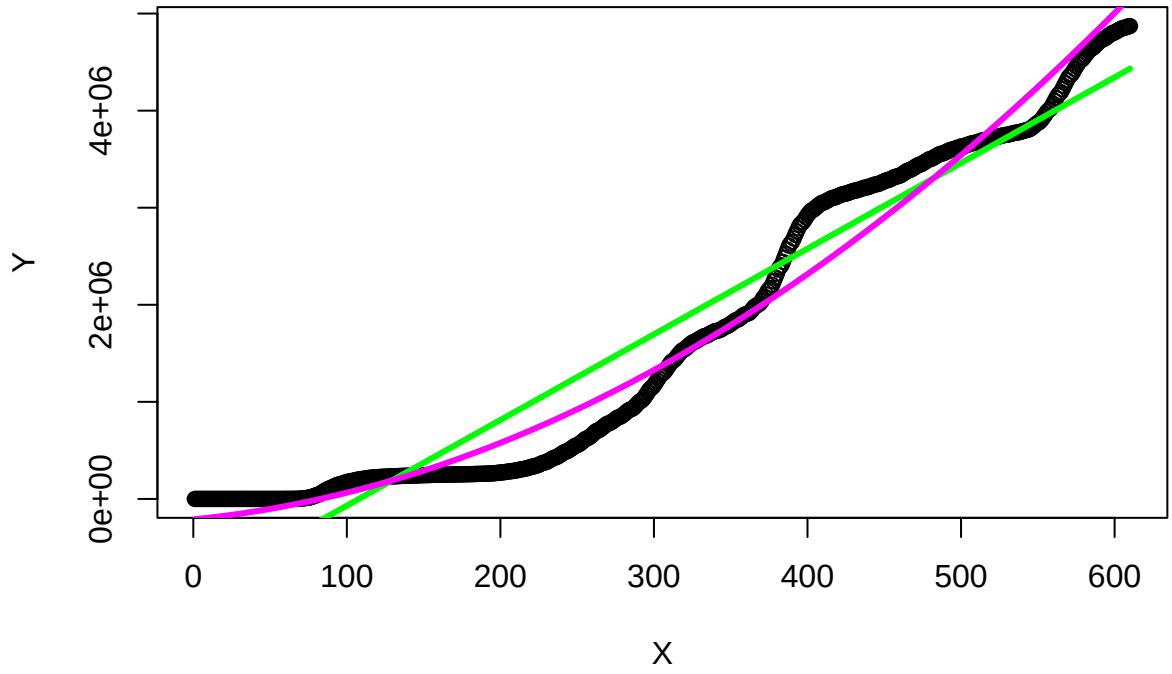
$$y_1 = -192.9501279 + 8.6234198x_1$$
$$y_2 = -3.9390348 \times 10^5 + 5552.4641634x_2$$
$$y_3 = 1.0142812 \times 10^5 + 941.1375602x_3$$
$$y_4 = -2.6118297 \times 10^6 + 1.2684961 \times 10^4 x_4$$
$$y_5 = -9.1560221 \times 10^6 + 3.0222859 \times 10^4 x_5$$
$$y_6 = 6.1692055 \times 10^5 + 5949.4810896x_6$$
$$y_7 = -5.8550233 \times 10^6 + 1.7809776 \times 10^4 x_7$$

2.3 Regresión lineal y regresión polinómica de grado 2

Realizamos el ajuste para una regresión lineal (de grado 1) y una regresión polinómica (de grado 2), mediante el uso de la funciones auxiliares previas.

```
plot(X, Y)
# Ajuste por regresin lineal (n=1) y representación
C_lin <- mypolyfit(X, Y, 1)
y_lin <- myeval(X, C_lin)
lines(X, y_lin, col = "green", lwd = 3)

# Ajuste por regresin lineal polinmica de grado 2 (n=2) y representación
C_pol2 <- mypolyfit(X, Y, 2)
y_pol2 <- myeval(X, C_pol2)
lines(X, y_pol2, col = "magenta", lwd = 3)
```



Para los ajustes solicitados las ecuaciones obtenidas son las siguientes: Para el ajuste lineal $y = -9.5101976 \times 10^5 + 8825.6602674x$ Mientras que para el ajuste polinómico de segundo grado $y = -2.1026391 \times 10^5 + 1563.3480847x + 11.8859447x^2$. Como vemos al contrario de la regresión segmentaria, la regresión lineal y polinómica de grado 2, contienen mas errores

2.4 Determinar los cambios de tendencia.

Los cambios de tendencia en la regresión segmentaria se representan en los puntos en los cuales las lineas se entrecruzan, por lo que para hallar estos valores (x,y) podemos realizar un sistema de ecuaciones y resolverlo

de forma matricial:

$$y_1 = b_1 x + a_1$$

$$y_2 = b_2 x + a_2$$

$$[A]\begin{bmatrix} y \\ x \end{bmatrix} = [B]$$

$$\begin{bmatrix} 1 & -b_1 \\ 1 & -b_2 \end{bmatrix}\begin{bmatrix} y \\ x \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$$

```
for (i in 1:(seg-1)) {
  # creamos la matriz que A
  A <- matrix(1, 2, 2)
  # recordamos que en la matriz CKs tenemos almacenados cada par de coeficientes de las funciones
  # lineales segmentarias
  A[2,] <- c(Cks[,i][2],Cks[,i+1][2])*-1
  B <- c(Cks[,i][1],Cks[,i+1][1])
  # transponemos la matrix para realizar el calculo segun nuestra ecuación
  AT <-  t(A)
  # Resolvermos el sistema con la función solve(),y rescatamos el segundo elemento
  # que corresponde con el valor de x
  xs <-solve(AT,B)[2]
  # De esta forma obtenemos los valores de x donde sucede la intersección de las lineas o el
  # cambio de tendencia los cuales son:
  print(xs)
}
```

```
## [1] 71.01765
## [1] 107.4163
## [1] 231.037
## [1] 373.1458
## [1] 402.6198
## [1] 545.6815
```

Vemos que se aproximan bastante a los puntos X que escogimos visualmente

```
print(ini_segmentos)
```

```
## [1]   1  70 115 230 370 405 550
```

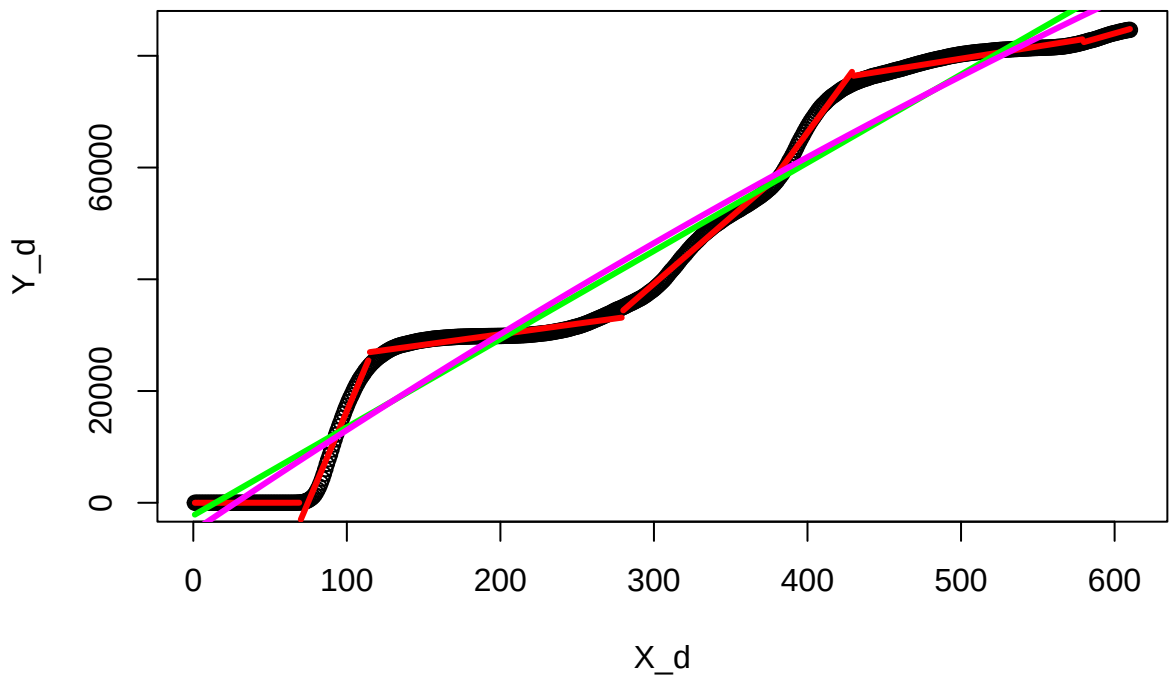
2.5 Repetir los apartados anteriores empleando los datos muertes acumuladas (etiqueta Cumulative_deaths)

```
Y_d <- myReadData_byDate("WHO-COVID-19-global-data-SPAIN.csv", "03/01/2020", "03/09/2021", "Cumulative_
m_d <- length(Y)
X_d <- 1:m
plot(X_d,Y_d)
seg_d <- 7
ini_segmentos_d <- c(1, 70, 115, 280, 380, 430, 580)
fin_segmentos_d <- c(ini_segmentos_d[2:length(ini_segmentos_d)] - 1, 610)
Cks_d <- matrix(0, 2, seg_d)
```

```r
for (i in 1:seg_d) {

  s_d <- ini_segmentos_d[i]:fin_segmentos_d[i]
  Xs_d <- s_d
  Ys_d <- Y_d[Xs_d]
  Cks_d[, i] <- mypolyfit(Xs_d, Ys_d, 1)
  lines(Xs_d, myeval(Xs_d, matrix(Cks_d[, i])), col = "red", lwd = 3)
}

C_lin_d <- mypolyfit(X_d,Y_d,1)
y_lin_d <- myeval(X_d, C_lin_d)
lines(X_d, y_lin_d, col = "green", lwd = 3)
C_pol2_d <-  mypolyfit(X_d,Y_d,2)
y_pol2_d <- myeval(X, C_pol2_d)
lines(X_d, y_pol2_d, col = "magenta", lwd = 3)
```




```r
for (i in 1:(seg_d-1)) {
  A_d <- matrix(1, 2, 2)
  A_d[2,] <- c(Cks_d[,i][2],Cks_d[,i+1][2])*-1
  AT_d <-  t(A_d)
  B_d <- c(Cks_d[,i][1],Cks_d[,i+1][1])
  xs_d <-solve(AT_d, B_d)[2]
  print(xs_d)
}
```

```
## [1] 74.65814
## [1] 116.3798
## [1] 273.6832
## [1] 375.9896
## [1] 426.7368
## [1] 595.75
```

Similar al caso anterior, los puntos x visuales para la regresión propuesta se aproxima bastante a los cambios de tendencia de la curva de MUERTES acumuladas por COVID 19. Las ecuaciones que rigen estos segmentos lineales son :

$$y_1 = -2.8064791 + 0.1290464x_1$$

$$y_2 = -4.831742 \times 10^4 + 647.2736495x_2$$

$$y_3 = 2.263109 \times 10^4 + 37.644581x_3$$

$$y_4 = -3.2215097 \times 10^4 + 238.0448785x_4$$

$$y_5 = -8.3480564 \times 10^4 + 374.3929892x_5$$

$$y_6 = 5.761948 \times 10^4 + 43.7441291x_6$$

$$y_7 = 3.7187524 \times 10^4 + 78.0403226x_7$$

Para los ajustes solicitados posteriormente las ecuaciones obtenidas son las siguientes: Para el ajuste lineal y =-2276.28 + 157.8573558x Mientras que para el ajuste polinómico de segundo grado y =-5090.4959396 + 185.4477446x + -0.0451561x^2.Observamos que en este caso el ajuste lineal y polinómico, son mas pobres en acierto, que en el caso anterior.

Los valores obtenidos del entrcruzamiento, guardan mucha relación con los valores propuestos visualmente para la confección de los segementos lineales.

```
print(ini_segmentos_d)
```

```
## [1]   1  70 115 280 380 430 580
```

```
for (i in 1:(seg_d-1)) {
  A_d <- matrix(1, 2, 2)
  A_d[2,] <- c(Cks_d[,i][2],Cks_d[,i+1][2])*-1
  AT_d <-  t(A_d)
  B_d <- c(Cks_d[,i][1],Cks_d[,i+1][1])
  xs_d <-solve(AT_d, B_d)[2]
  print(xs_d)
}
```

```
## [1] 74.65814
## [1] 116.3798
## [1] 273.6832
## [1] 375.9896
## [1] 426.7368
## [1] 595.75
```