Instituto Jorge Sabato, 25 años.
Comisión Nacional de Energía atómica.

Modelización de Materiales 2019

Resumen de la Guía 1

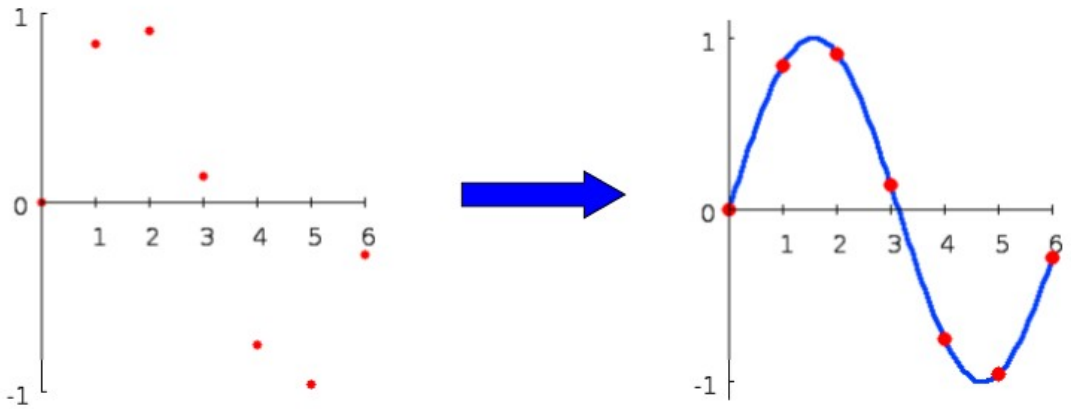
Mariano Forti - Ruben Weht

marianodforti@gmail.com - ruweht@cnea.gov.ar

www.tandar.cnea.gov.ar/~weht/Modelización

https://mdforti.github.io/Modelizacion/

Ejercicio 1: Interpolación



El ejercicio nos pide interpolar con splines a un Conjunto de puntos dados ...

Ejercicio 1: Cuentas

En cada intervalo se aproxima por un polinomio de orden 3

$$f_i(x) = d_i + c_i(x - x_i) + b_i(x - x_i)^2 + a_i(x - x_i)^3 \quad ; \quad x \in [x_i, x_{i+1}]$$

4(N-1) incognitas

Que debe pasar por los puntos

$$f_i(x_i) = y_i \Rightarrow d_i = y_i \qquad \text{N ecuaciones}$$

Que deben formar una curva continua

$$f_i(x_{i+1}) = f_{i+1}(x_{i+1}) \qquad \text{N-2 ecuaciones}$$

Que deben formar una curva suave

$$f_i'(x_{i+1}) = f_{i+1}'(x_{i+1}) \qquad \text{N-2 ecuaciones}$$

La derivada segunda debe ser continua

$$f_i''(x_{i+1}) = f_{i+1}''(x_{i+1}) \qquad \text{N-2 ecuaciones}$$

Los splines eran polinomios a que se ensamblan de a tramos para construir una función que debe 1) pasar por los puntos , 2) ser continua, 3) ser dos veces deribable.

Splines Cúbicos: Sistema Lineal

Resolviendo para los b :

$$3\left(\frac{y_{i+1}-y_i}{h_i}-\frac{y_i-y_{i-1}}{h_{i-1}}\right)=b_{i-1}h_{i-1}+2b_i\left(h_i+h_{i-1}\right)+b_{i+1}h_i$$

Es un sistema de N – 2 ecuaciones con N - 1 incognitas b_1 … b_{N-1}

Condiciónes extra: Derivadas segundas en los extremos

$b_1 = 0$ → La condición de contorno fija el valor del b_1, reescribiendo la ecuación para i = 1

$b_N = 0$ → Es el coeficiente de un intervalo extra a la derecha del último punto. No nos interesan los otros coeficientes en ese intervalo.

El desarrollo está en el “apunte de interpolación”, pero la idea es que si uno despeja las ecuaciones puede sacar los coeficientes b, siempre que se impongan algunas condiciones de controno. Por ejemplo estaban estas, que significan que las derivadas segundas se tienen que anular en los extremos.

Ejercicio 1: Sistema Lineal

$$\begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ h_1 & 2(h_2+h_1) & h_2 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & h_2 & 2(h_3+h_2) & h_3 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & h_{N-3} & 2(h_{N-2}+h_{N-3}) & h_{N-2} & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & h_{N-2} & 2(h_{N-1}+h_{N-2}) & h_{N-1} \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_{N-2} \\ b_{N-1} \\ b_N \end{pmatrix} =$$

$$3 \begin{pmatrix} 0 \\ \frac{y_3-y_2}{h_2} - \frac{y_2-y_1}{h_1} \\ \vdots \\ \frac{y_N-y_{N-1}}{h_{N-1}} - \frac{y_{N-1}-y_{N-2}}{h_{N-2}} \\ 0 \end{pmatrix}$$

5

El sistemita de ecuaciones sobre las b puede leerse como una ecuación matricial que se resuelve por métodos de álgebra lineal, por ejemplo pasar la matriz inversa para el otro lado.

Splines Cúbicos: Polinomios por tramos

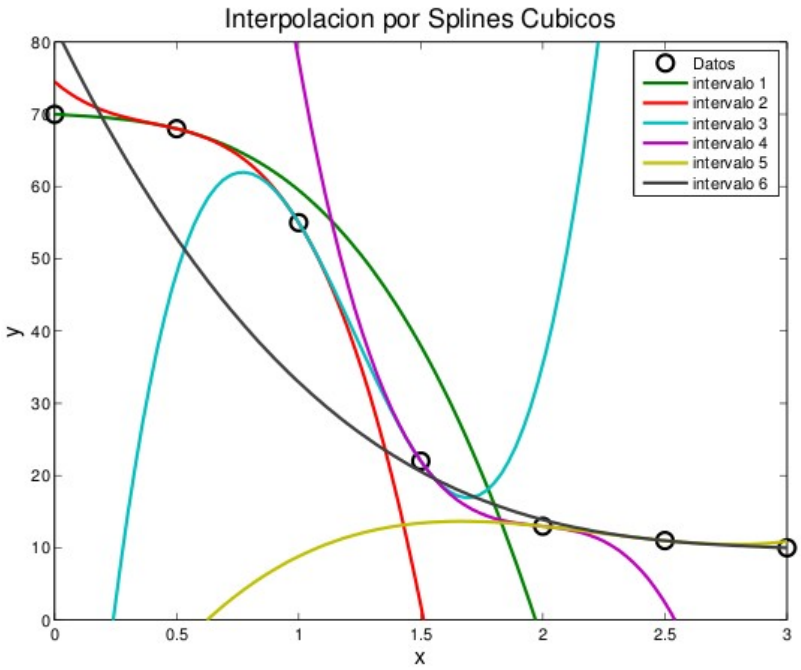
Los polinomios solo valen en los tramos en que se definen

$$f_i(x) = d_i + c_i(x - x_i) + b_i(x - x_i)^2 + a_i(x - x_i)^3 \quad ; \quad x \in [x_i, x_{i+1}]$$

$$d_i = y_i$$

$$c_i = \frac{(y_{i+1} - y_i)}{h_i} - b_i h_i - a_i h_i^2$$

$$a_i = \frac{1}{3} \frac{(b_{i+1} - b_i)}{h_i}$$



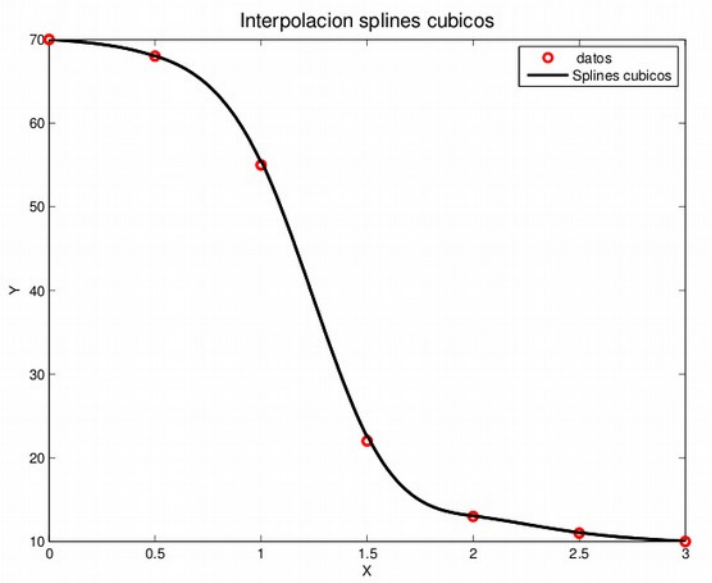
Los otros coeficientes, a c y d salen por relaciones de recursividad (una vez que saco los b puedo calcular los a y los c). entonces uno puede graficar los polinomios como quedan definidos.

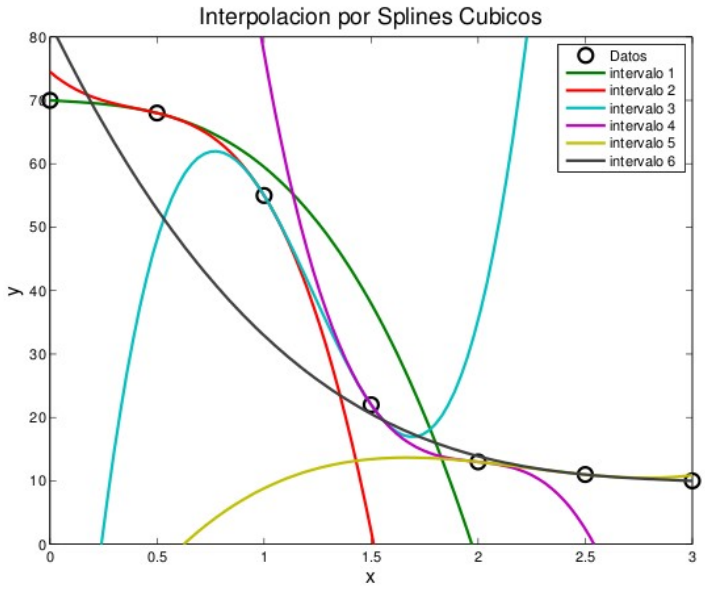
Lo que se observa es que cada polinomio pasa solo por los dos puntos que determinan su intervalo de definición, y se alejan de los demas!

Splines cúbicos: Resultado Final

Resolución de todos los coeficientes.

$$f_i(x) = d_i + c_i(x - x_i) + b_i(x - x_i)^2 + a_i(x - x_i)^3 \quad ; \quad x \in [x_i, x_{i+1}]$$



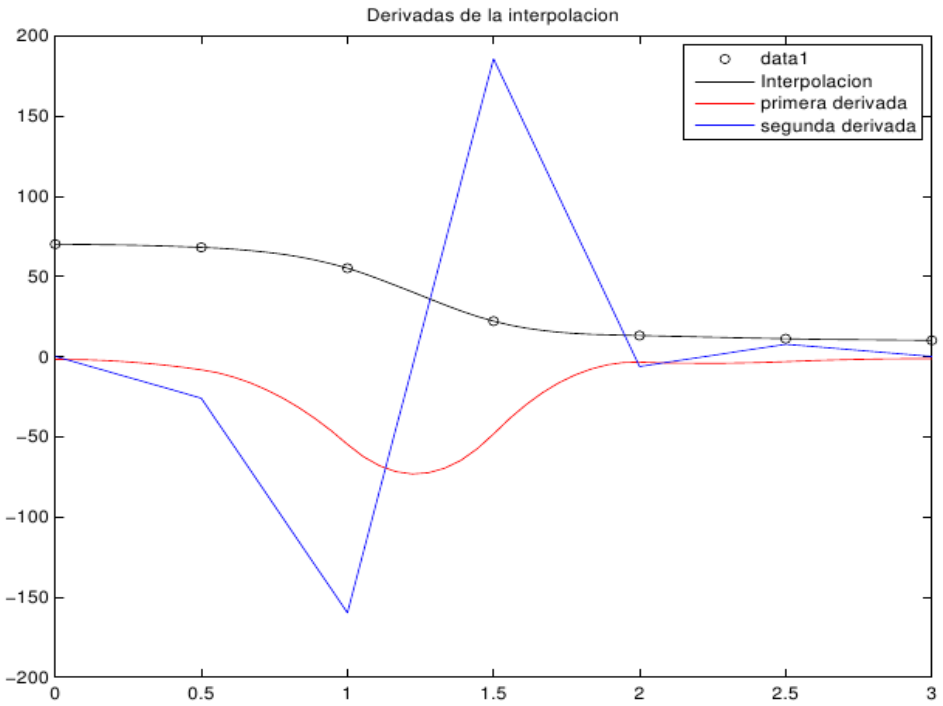


Al recortar los polinomios al pedacito entre los puntos que tocan, nos queda la función que buscábamos.

Ejercicio 1: Varias soluciones Posibles

$$f'_i(x) = c_i + 2b_i(x - x_i) + 3a_i(x - x_i)^2 \quad ; \quad x \in [x_i, x_{i+1}]$$

$$f''_i(x) = 2 * b_i + 6 * a_i(x - x_i) \quad ; \quad x \in [x_i, x_{i+1}]$$



El ejercico no termina ahí, sino que tenemos que encontrar la posición donde la derivada es máxima.

La primer opción para eso es derivar los pòlinomios a mano. El probmela de eso es que las derivadas de los polinomios son de orden finito, por lo que perdemos resolución al derivar una y otra vez.

Ejercicio 1: Varias Soluciones Posibles

$$xo \in [x_i, x_{i+1}] \Rightarrow F(x_i) \cdot F(x_{i+1}) \leq 0$$

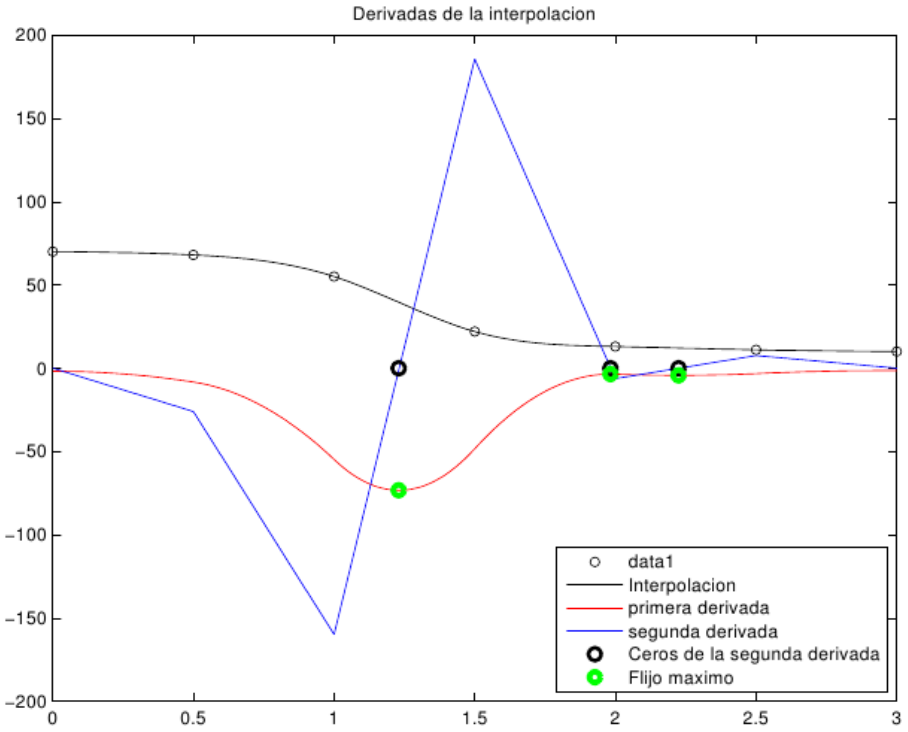
Este gráfico está bien para ilustrar el cumplimiento de las condiciones impuestas a los polinomios, pero la deri vada de los polinomios no necesariamente es una buena aproximación a las derivadas de la función. Por ejemplo, fijate que esa derivada segun da tiene varios ceros aunque la función tenga un único punto de inflección.

Ceros malos:

```
>> zomx

zomx =

    1.2298
    1.9823
    2.2247

>>
```

Flujo

```
>> DYom

DYom =

  -73.3144   -3.5336   -4.2959
```

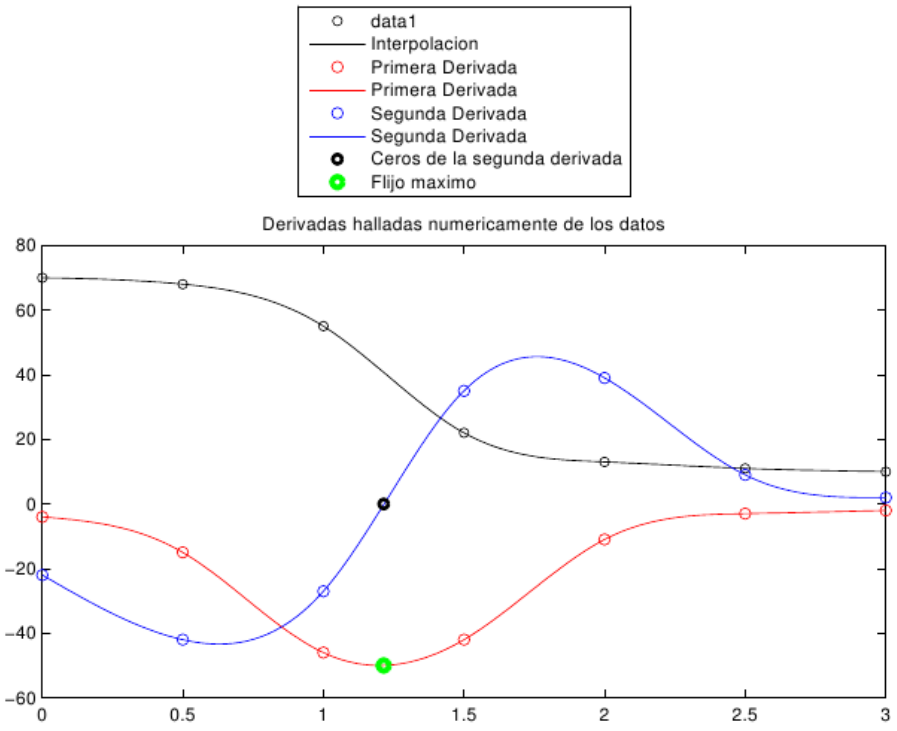


Entonces por ejemplo si buscamos el lugar donde la derivada segunda se anula, encontramos que el resultado así obtenido nos da varios lugares, básicamente porque la derivada segunda de un ensamble de polinomios es un “serrucho”

Ejercicio 1: Varias Soluciones posibles

$$f'(x_i) = \frac{f(x_{i+1}) - f(x_{i-1})}{x_{i+1} - x_{i-1}} + interpolación$$

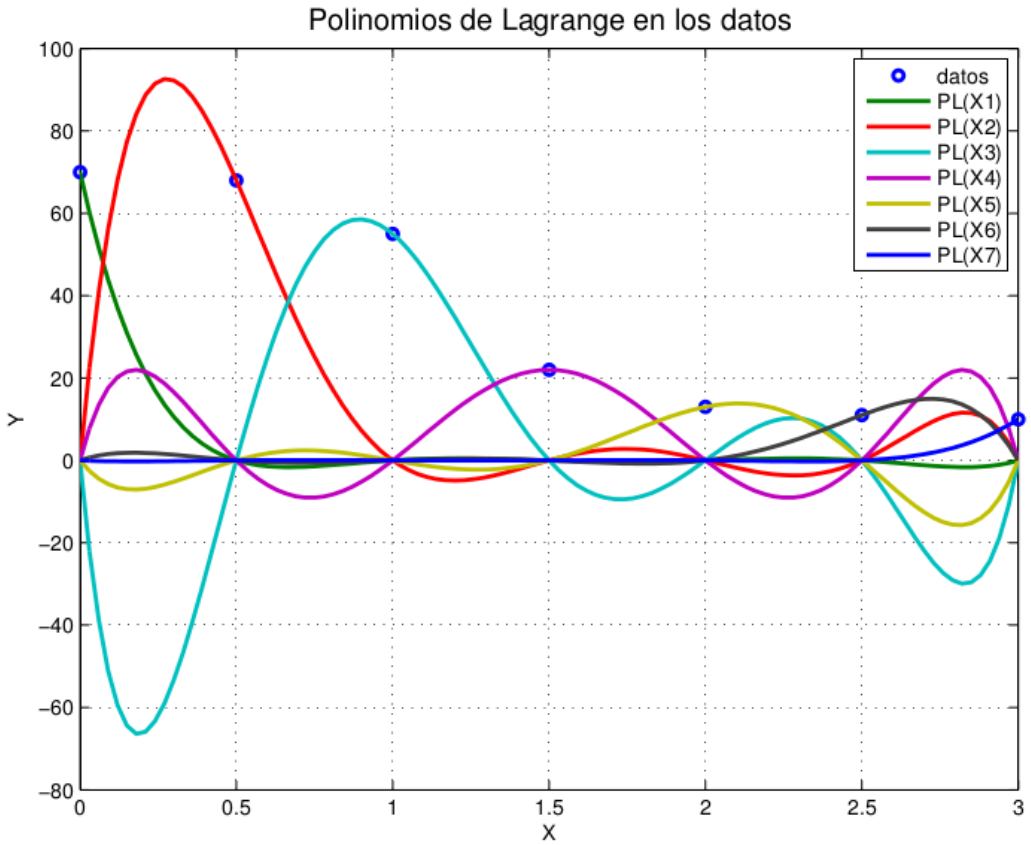
```
zox =

    1.2146

DYo =

   -49.9587
```



En cambio si derivamos los datos originales y calculamos una iinterpolación a esa derivada, ahora el ensamble de polinomios para la derivada nos da una derivada suave.

Si vuelvo a derivar otra vez con el mismo método, obtengo una derivada segunda también suave, con algun resultado para la posición del máximo.

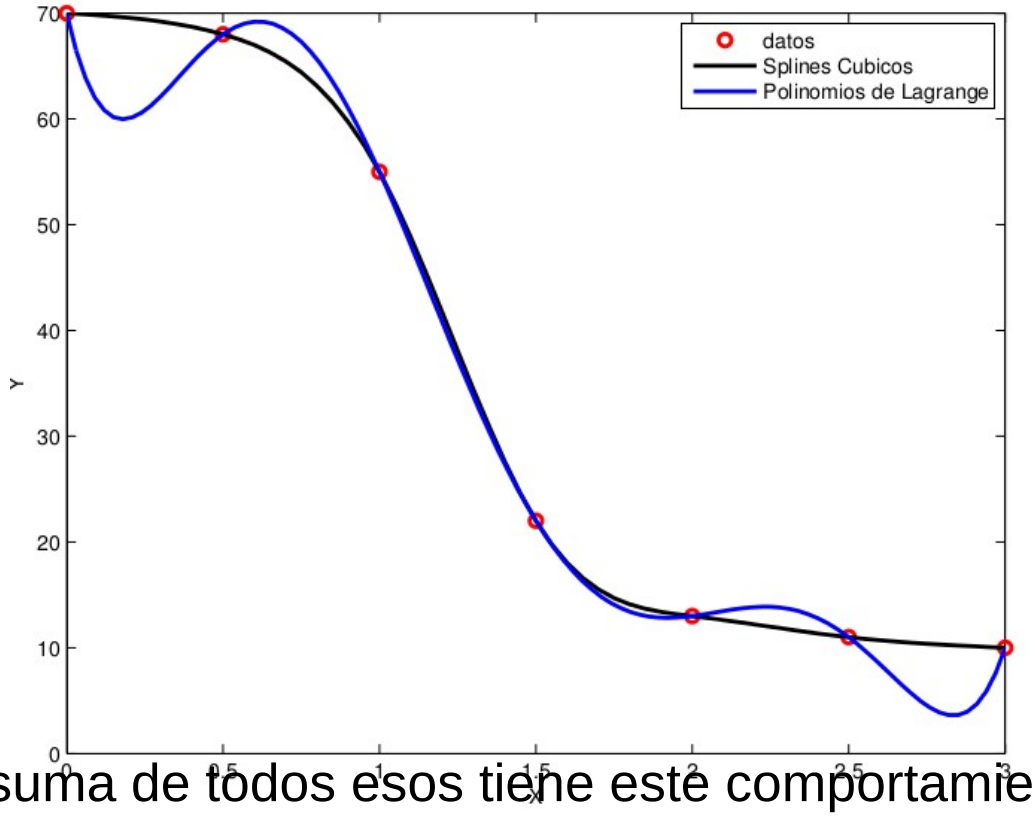
Polinomios de Lagange



Esto es simplemente anecdótico, cuando uno busca ajustar con polinomios de lagrange pide condiciones distintas.

Aca se ve que cada polinomio de lagrange pasa por uno de los puntos pero se anula en todos los demas,

Polinomios de Lagrange



Y la suma de todos esos tiene este comportamiento, donde se ve que al no haber condiciones sobre las derivadas en los extremos el “ensamble” hace cosas raras...

Ejercicio 2: Integración

Integración de una función distribución:

$$dF = 200\left(\frac{z}{(5+z)}\right)\exp(-2z/30)\,dz$$

$$F = \int_0^{30} dF$$

$$d = \left(\frac{1}{F}\right)\int_0^{30} z\,dF$$

Ejercicio 2: Integración

$$I^{Trapecios}_{N\ intervalos} = \frac{1}{2}\left(\frac{b-a}{N}\right)\left(f(a)+2\sum_{j=2}^{N} f(x_j)+f(b)\right)$$

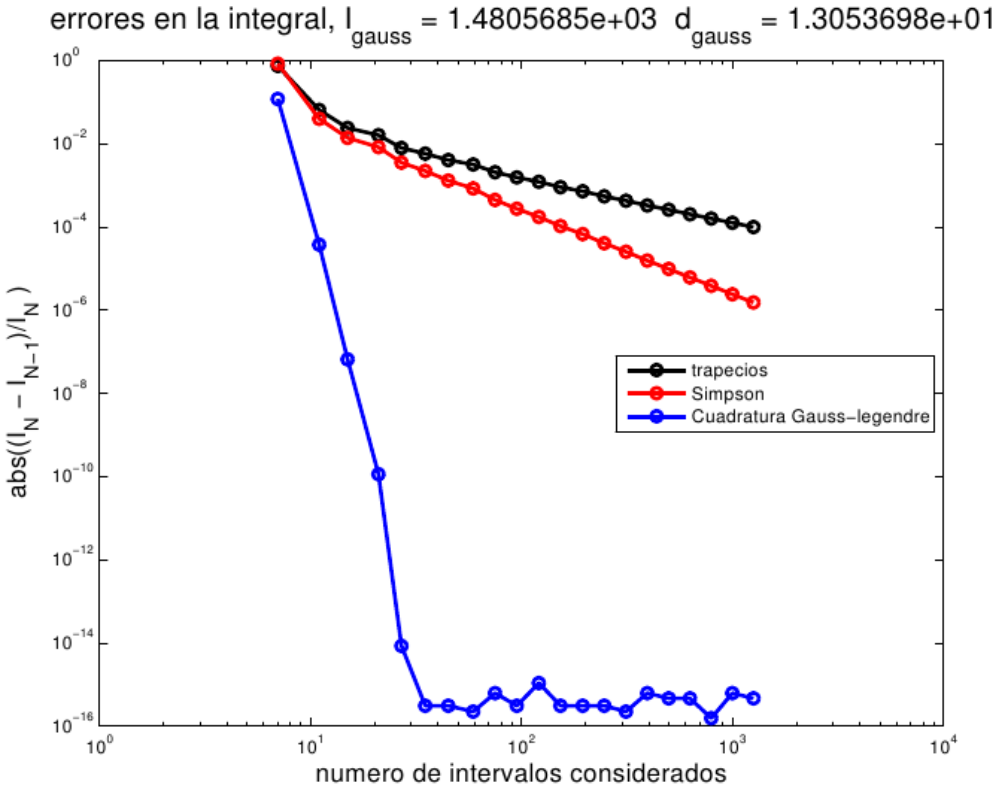
$$I^{Simpson}_{N\ intervalos} = \frac{1}{3}\left(\frac{b-a}{N}\right)\left(f(a)+4\sum_{j=3,\,impar}^{N} f(x_j) \;+\; 2\sum_{j=2,\,pares}^{N-1} f(x_j)+f(b)\right)$$

$$I^{Gauss}_{N\ intervalos} = \sum_{j=1}^{N+1} W_j f(t_j)$$

$$\begin{cases} \sum_{j=1}^{N+1} W_j = \int_{-1}^{1} dt \\ \sum_{j=1}^{N+1} W_j t_j = \int_{-1}^{1} t\,dt \\ \sum_{j=1}^{N+1} W_j t_j^2 = \int_{-1}^{1} t^2\,dt \\ \vdots \\ \sum_{j=1}^{N+1} W_j t_j^{2N+1} = \int_{-1}^{1} t^{2N+1}\,dt \end{cases}$$

Subrutina lgwt de matlabcentral

Ejercicio 2: Resultado y Error



Ejercicio 3: Ecuaciones Diferenciales Ordinarias

- Ecuación Diferencial: $\frac{dy}{dx} = 4e^{0.8x} - 0.5y$

$$\frac{dy}{dt} = \tilde{F}(x, y) = 4e^{0.8x} - 0.5y$$

- Solución Teórica: $y(x) = \frac{4}{1.3}\left(e^{0.8x} - e^{-0.5x}\right) + 2e^{-0.5x}$

Ejercicio 3: Integración de la ecuación

$$\frac{y_i - y_{i-1}}{dt} = \tilde{F}\left(t_{i-1}, y_{i-1}\right)$$

Euler $$Y_i = Y_{i-1} + dt\ \tilde{F}\left(t_{i-1}, v_{i-1}\right)$$

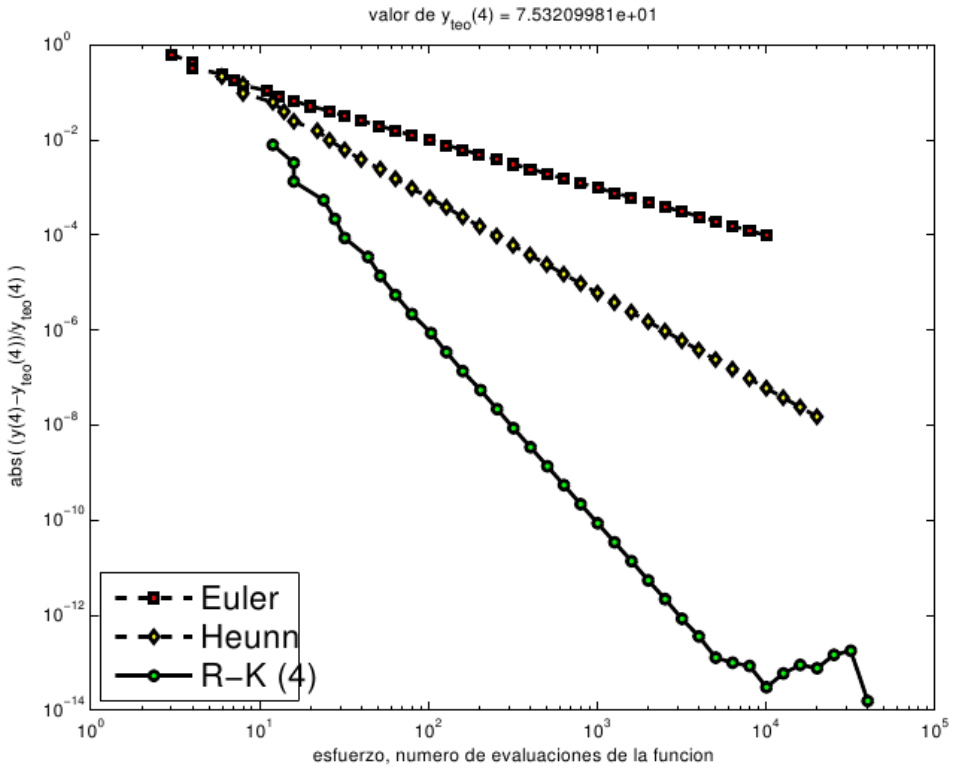
Heun
$$k_1 = \tilde{F}\left(t_{i-1}, y_{i-1}\right)$$
$$k_2 = \tilde{F}\left(t_{i-1} + dt, y_{i-1} + k_1\, dt\right)$$
$$y_i = y_{i-1} + \frac{1}{2} dt\left(k_1 + k_2\right)$$

R-K
$$k_1 = \tilde{F}\left(t_{i-1}, Y_{i-1}\right)$$
$$k_2 = \tilde{F}\left(t_{i-1} + \frac{1}{2} dt, Y_{i-1} + \frac{1}{2} k_1\, dt\right)$$
$$k_3 = \tilde{F}\left(t_{i-1} + \frac{1}{2} dt, Y_{i-1} + \frac{1}{2} k_2\, dt\right)$$
$$k_4 = \tilde{F}\left(t_{i-1} + dt, Y_{i-1} + k_3\, dt\right)$$
$$Y_i = Y_{i-1} + \frac{1}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right) dt$$

Problema de Ecuaciones Diferenciales.

Todos saben ya las cuentas para programar RK de orden 4,

Ejercicio 3: Solución y error



Lo que me interesaba mostrarles es esto.

Si calculan algun error, en este caso respecto del valor teórico de la solución, en función del número de veces que se evalúa la función, tienen una cosa así.

Lo que se ve es que para el caso de runge kutta, a pesar de que la función se ejecuta cuatro veces en cada iteración, el error es varios ordenes de magntitud menor para igual número de evaluaciones que para otros métodos.

Ahí está la ventaja de refinar el método!