UNIVERSIDAD TECNOLÓGICA NACIONAL

Facultad Regional Santa Fe

Matemática Superior

Ingeniería en Sistemas de Información

Trabajo Práctico 3

Señales biológicas: ECG

Alumno	Correo electrónico
Izaguirre, Ezequiel	ezequielmizaguirre@gmail.com
Pacheco Pilan, Federico Ignacio	fedepacheco2112@gmail.com
Rodríguez, Alejandro	rodriguezalejandro_@hotmail.com

ISI A • 2021

1. Herramientas utilizadas

Se hizo uso de las siguientes librerías del lenguaje de programación python:

- *numpy*
- *sympy*
- *scipy*
- *pandas*
- *matplotlib*

2. ECG

2.1. Modelado de la señal:

Preliminarmente, dado que el enunciado indica que la duración del muestreo es de un total de 60 segundos, se normalizó el dominio multiplicando cada valor del mismo por $\frac{6}{10}$ en tanto que la señal provista por la cátedra se encuentra entre $[0, 100]$.

Para efectuar el modelado de la señal, se la dividió en tramos y se emplearon polinomios del mismo grado para cada uno, cuyos coeficientes se calcularon por el método de mínimos cuadrados. Se decidió esto puesto que el muestreo de la señal original presenta ruido, con lo que realizar una interpolación no es recomendable.

Para cumplir el requerimiento de continuidad y suavidad, antes aplicar de mínimos cuadrados se planteó lo siguiente[1]:

$$x_i(e_i) = x_{i+1}(e_{i+1})$$
$$x'_i(e_i) = x'_{i+1}(e_{i+1})$$

$$x_1(e_1) = 0$$
$$x'_1(e_1) = 0$$

$$x_n(e_n) = 0$$
$$x'_n(e_n) = 0$$

Dónde:

- $x_i(t)$ es el polinomio de grado *m* del intervalo *i*
- e_i es el extremo del intervalo *i*
- *n* es la cantidad de intervalos
- $1 \leq i < n$

[1] Se han detallado sólo expresiones genéricas puesto que en la implementación se usaron herramientas de cálculo simbólico para poder emplear grados de polinomio arbitrarios.

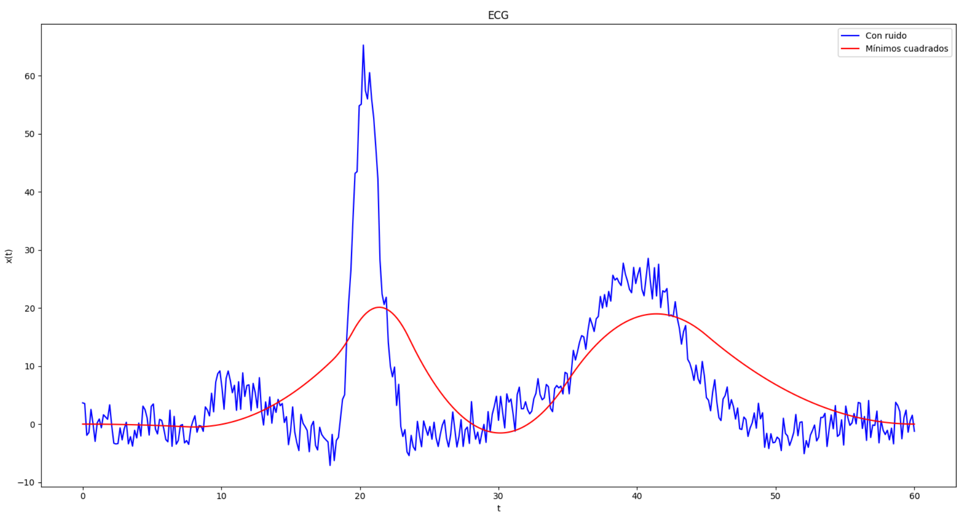
Esto constituye un sistema de ecuaciones lineales que al resolverlo resulta compatible indeterminado. Reemplazando las expresiones obtenidas en los polinomios, ya puede aplicarse mínimos cuadrados para hallar la escalares de *x(t)*. Para la implementación del método se siguió el enfoque propuesto en las clases prácticas de la cátedra (en lo que refiere a los $\varphi_i(t)$, $\gamma(t)$, tablas auxiliares, etc.), además de recurrir a métodos útiles como *numpy.dot* y *numpy.linalg.solve*, entre otros.

Por prueba y error, se eligieron los siguientes valores para los parámetros:

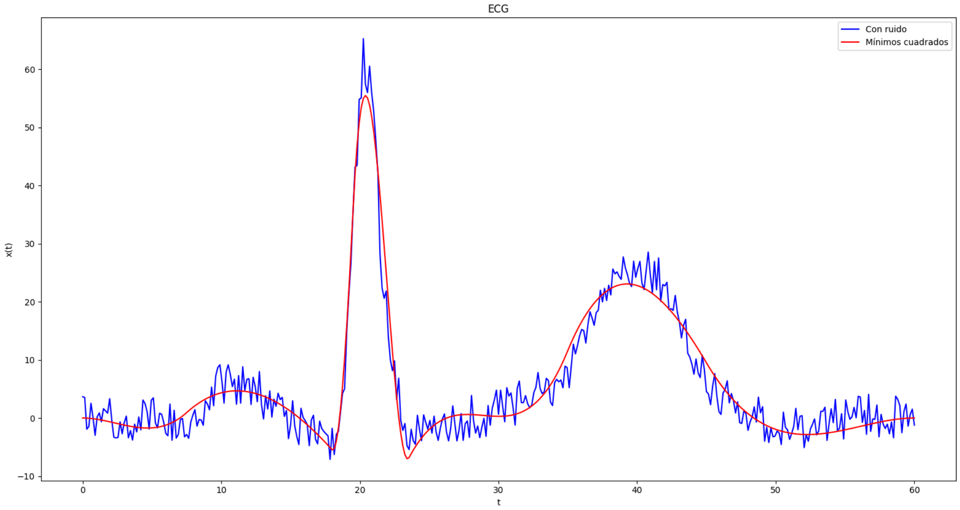
$$n = 7$$

$$e = [0.,\ 7.5,\ 18.,\ 19.5,\ 23.55,\ 34.95,\ 45.,\ 60.]$$

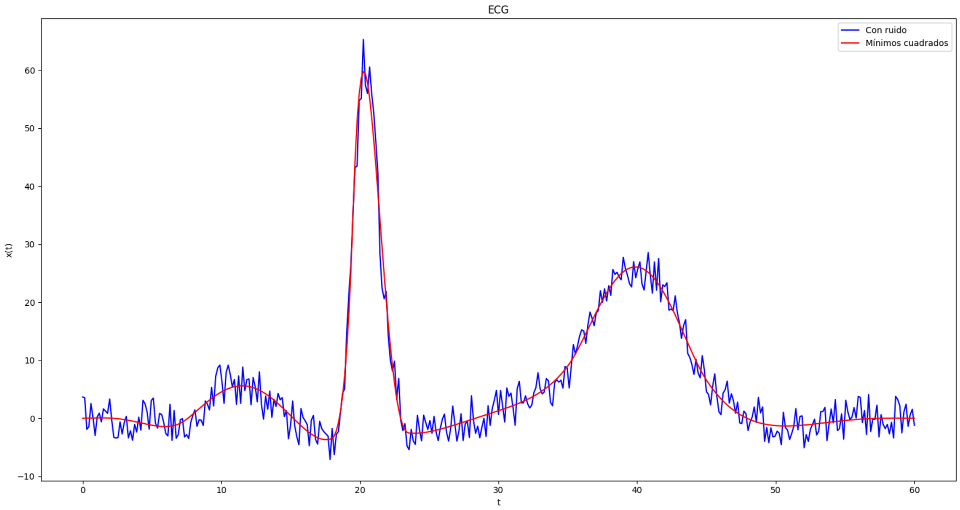
Para el caso del grado de los polinomios, se muestran a continuación gráficas para distintos valores de *m* manteniendo los intervalos constantes:



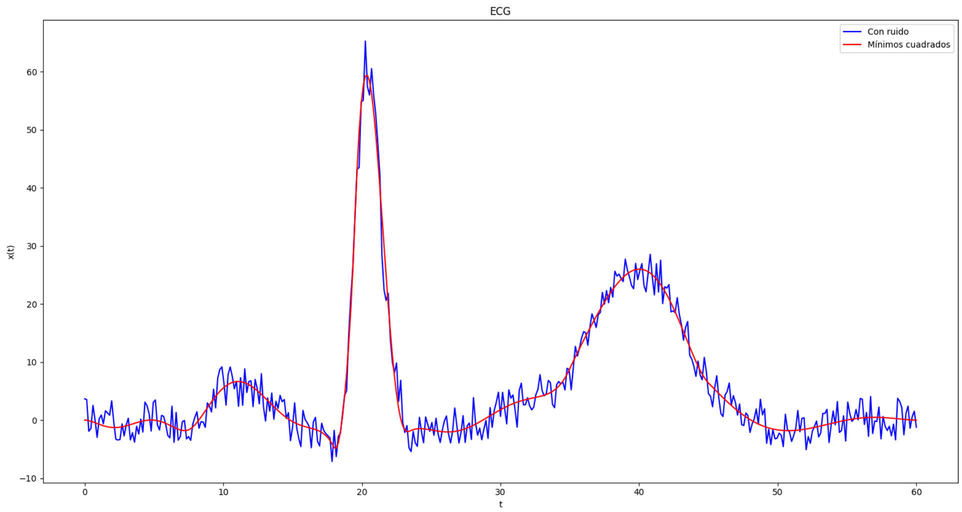
$$m = 2$$


ECG
Con ruido
Mínimos cuadrados
x(t)
t

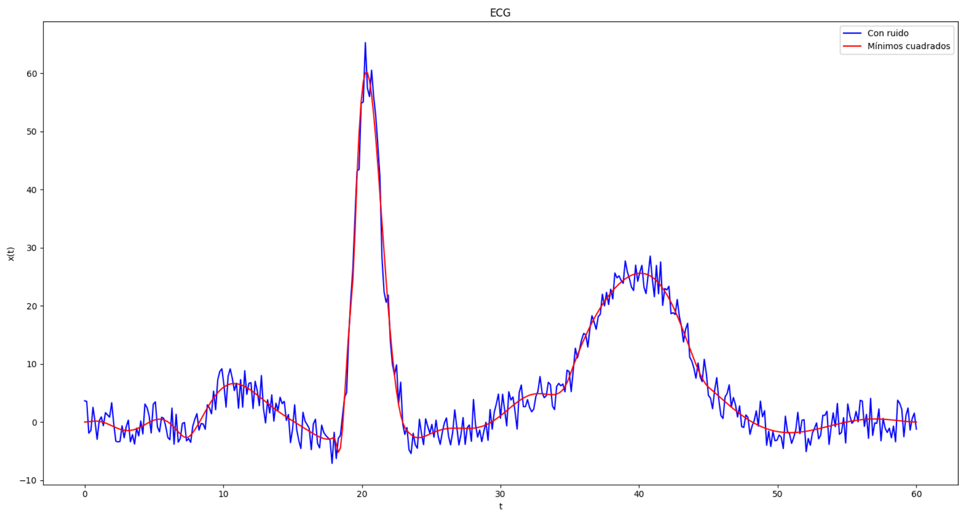
$$m = 3$$


ECG
Con ruido
Mínimos cuadrados
x(t)
t

$$m = 4$$



$$m = 5$$



$$m = 6$$

Se observa que emplear parábolas resulta insuficiente. Con cúbicas en adelante se obtienen aproximaciones más razonables. A juicio de los redactores la aproximación más óptima se logra con polinomios de grado 4: se mantiene cierta relación calidad del ajuste - costo computacional y se evita el problema de las “oscilaciones” que se da en grados superiores. En tal caso si se redondean los coeficientes a cuatro decimales *x(t)* es de la forma:

$$x(t) = \begin{cases} 0.0064\,t^4 + 0.0625\,t^3 + 0.1027\,t^2, & 0 \le t < 7.5 \\ 0.0073\,t^4 + 0.3404\,t^3 + 5.4550\,t^2 - 34.8291t + 74.6945, & 7.5 \le t < 18 \\ 4.1426\,t^4 - 301.1486\,t^3 + 8220.2616\,t^2 - 99849.9491\,t + 455358.4536, & 18 \le t < 19.5 \\ -1.2346\,t^4 + 111.7921\,t^3 - 3785.4912\,t^2 + 56797.7571\,t - 318496.4146, & 19.5 \le t < 23.55 \\ 0.0022\,t^4 - 0.2457\,t^3 + 10.4804\,t^2 - 197.8181t + 1388.7318, & 23.55 \le t < 34.95 \\ 0.0155\,t^4 - 2.4783\,t^3 + 147.5274\,t^2 - 3873.3413\,t + 37859.2746, & 34.95 \le t < 45.0 \\ 0.0008\,t^4 - 0.1761\,t^3 + 14.8697\,t^2 - 556.4358\,t + 7783.3651, & 45.0 \le t < 60.0 \end{cases}$$

2.2. Análisis de las ondas P, QRS y T:

Para el caso $m = 4$, y considerando que la duración de cada onda es el tiempo entre sus dos mínimos relativos más cercanos, se tienen los siguientes valores redondeados a dos decimales:

Onda	Duración [s]	Amplitud
P	11.7	9.28
QRS	6.3	63.45
T	26.85	28.69

2.3. Análisis de la calidad del ajuste:

2.3.1. Definición de ajuste elegida:

Para el cálculo del ajuste (en porcentaje) se decidió utilizar una fórmula del error relativo modificada para salvar los casos en los que el denominador se anula[2]:

$$ajuste\ (t_j) = \frac{|x_{aproximado}(t_j) - x_{medido}(t_j)|}{1 + |x_{aproximado}(t_j)|}.100$$

Pese a que no es una solución ideal, se eligió hacer esto por simplicidad.

[2] https://stats.stackexchange.com/a/86710

2.3.2. Valores del ajuste para polinomios de distinto grado:

Se decidió calcular el ajuste mínimo, promedio y máximo para cada intervalo, así como para toda la duración de la señal. Así, redondeando a dos decimales, se tiene:

Ajuste mínimo					
Intervalo	**m = 2**	**m = 3**	**m = 4**	**m = 5**	**m = 6**
[0; 7.5)	6.43	4.35	2.57	9.89	5.35
[7.5; 18)	2.33	2.26	0.92	0.62	0.93
[18; 19.5)	4.07	3.02	4.08	1.34	1.47
[19.5; 23.55)	2.92	0.49	0.48	0.20	0.44
[23.55; 34.95)	3.85	1.75	2	0.23	1.59
[34.95; 45)	0.31	0.48	0.11	0.24	0.54
[45; 60)	15.67	0.14	2.49	2.08	3.02
[0; 60)	0.31	0.14	0.11	0.26	0.25

Ajuste promedio					
Intervalo	**m = 2**	**m = 3**	**m = 4**	**m = 5**	**m = 6**
[0; 7.5)	169.48	112.47	133.55	123.74	118.58
[7.5; 18)	182.98	66.66	62.6	60.66	60.42
[18; 19.5)	79.35	21.92	31.16	19.1	20.85
[19.5; 23.55)	110.13	39.76	23.52	37.59	30.81
[23.55; 34.95)	129.77	110.74	67.66	59.05	60.73
[34.95; 45)	29.99	13.21	9.04	8.77	8.78
[45; 60)	107.31	81.77	113.03	100.35	99.02
[0; 60)	119.1	72.68	72.74	67.05	66.12

Ajuste máximo					
Intervalo	**m = 2**	**m = 3**	**m = 4**	**m = 5**	**m = 6**
[0; 7.5)	366	366	366	366	366
[7.5; 18)	847.38	435.59	327.05	334.5	303.63
[18; 19.5)	142.92	57.06	83.98	51.59	47.41
[19.5; 23.55)	238.10	560.37	217.94	542.89	362.88
[23.55; 34.95)	396.02	362.24	333.36	297.11	285.71
[34.95; 45)	56.34	52.86	39.91	41.63	42.27
[45; 60)	323.48	324.29	382.65	348.67	343.02
[0; 60)	847.38	560.37	382.65	554	402.47

3. Polinomio de Newton modificado

3.1. Generalización de la tabla de diferencias divididas para derivadas de cualquier orden:

Basado en la construcción de la tabla de diferencias divididas presentada en clase de práctica, se generaliza la interpolación de derivadas de cualquier orden de la siguiente manera:

- Por cada condición de derivada, se agrega una columna al lado de la que contiene el punto (x, y), siendo x el valor en donde se quiere interpolar dicha derivada, repitiendo dichos valores.
- La diferencia dividida *i-ésima* ($1 \leq i \leq n - 1$condiciones) que no puede ser calculada (dado que su denominador es cero), es reemplazada por el valor de $y^{(i)}(x)$ deseado dividido por $i!$

Esto puede verse mejor con el siguiente ejemplo, acompañado de gráficos:

Supóngase que se desea interpolar dos puntos: (1, 1) y (2, 4)

X	1	2
Y	1	4

Pero también se quiere que $y'(1) = 0.5$, $y''(1) = 1$, $y'''(1) = 1.5$. Entonces, se agregan tres columnas más, repitiendo (1,1) porque las tres derivadas corresponden a $x = 1$:

X	1	1	1	1	2
Y	1	1	1	1	4

Luego, se calculan las diferencias divididas y se reemplaza el valor de la derivada deseada dividido su factorial en la posición mencionada:

X	1	1	1	1	2
Y	1	1	1	1	4

Diferencia dividida: 1 → $\frac{0.5}{1!}$ $\frac{0.5}{1!}$ $\frac{0.5}{1!}$ 3

Diferencia dividida: 2 → $\frac{1}{2!}$ $\frac{1}{2!}$ 2.5

Diferencia dividida: 3 → $\frac{1.5}{3!}$ 2

Diferencia dividida: 4 → 1.75

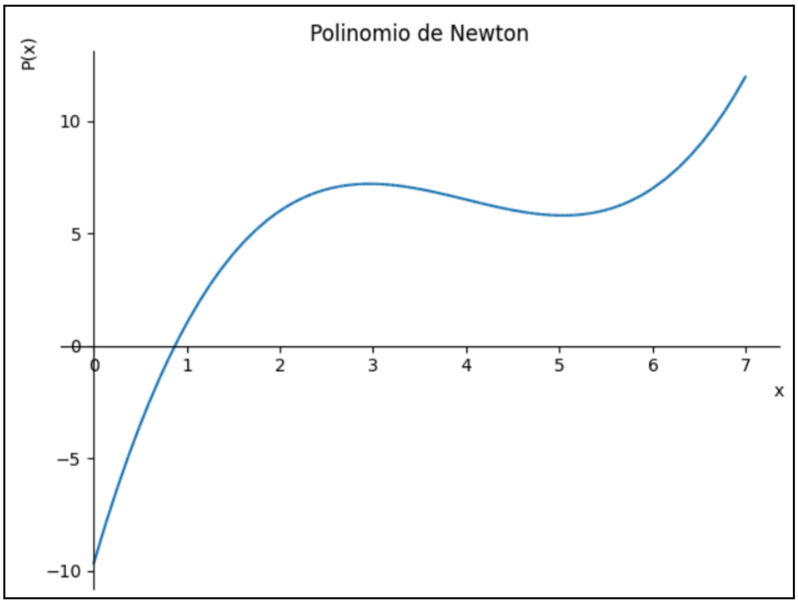
El polinomio de Newton resultante, y su gráfica se pueden ver en el 5to ejemplo de la siguiente sección (3.2 Ejemplos).

Se considera que, si se desea interpolar la i-ésima derivada en un valor de x, también se desea $y(x)$ y la derivada $i - 1$, dado que son datos imprescindibles para la construcción de la tabla.

3.2. Ejemplos:

Ejemplo 1

X	1	2	4	6
Y	1	6	6.5	7



$$P(x) = 0.317\,x^3 - 3.8\,x^2 + 14.183\,x - 9.7$$

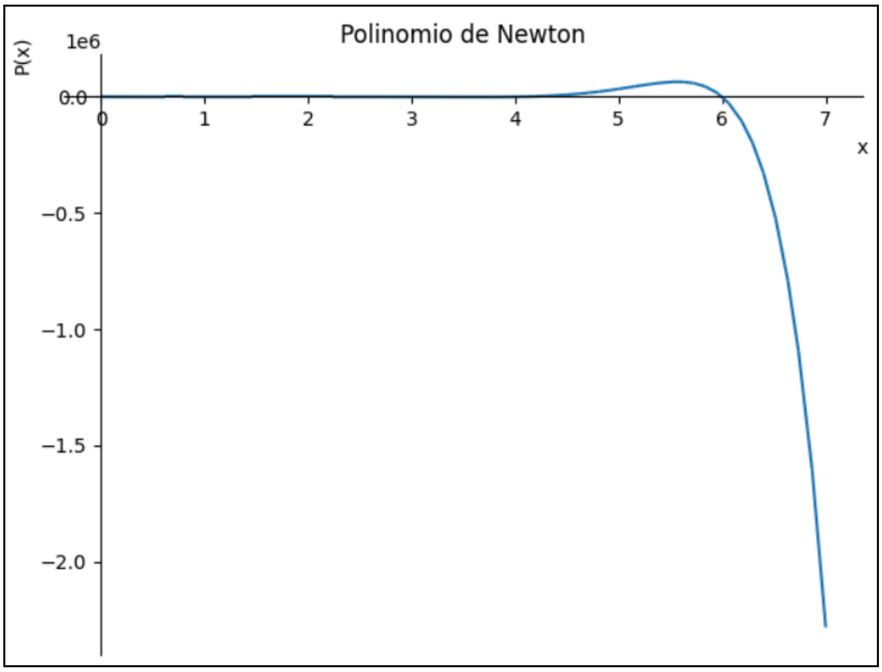
Salida:

Puntos:
P(1) = 1.00000000000000
P(2) = 6.00000000000000
P(4) = 6.50000000000000
P(6) = 7.00000000000000

Ejemplo 2

X	1	2	4	6
Y	1	6	6.5	7
Y’	9	5		
Y”	3	7		
Y’’’	7			
Y ^{iv}	2			

(esta es una prueba para testear de manera más completa el algoritmo)



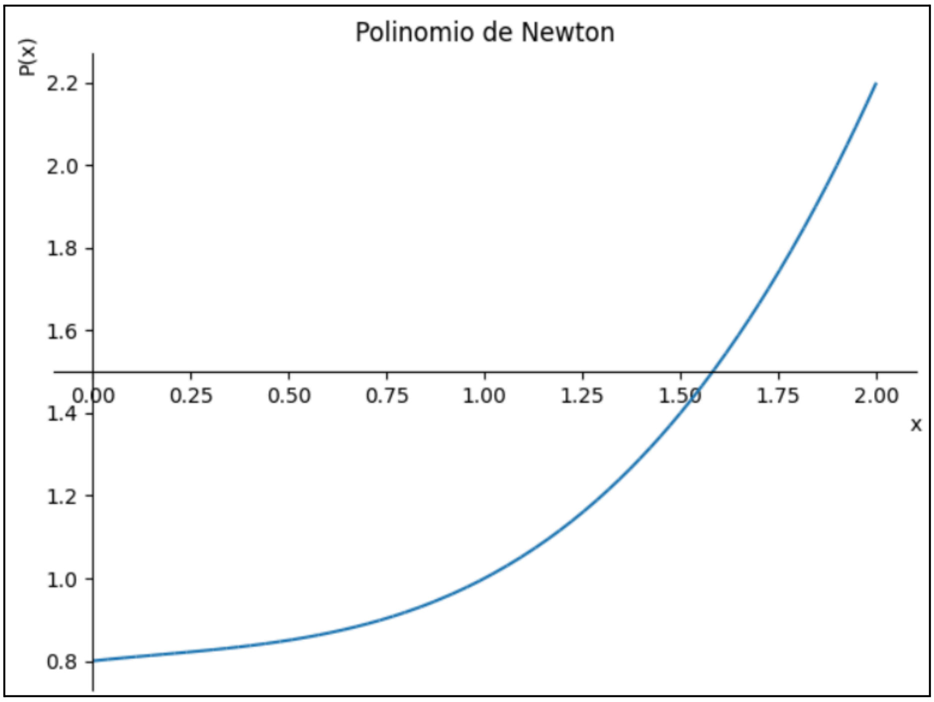
$$P(x) = -4.338\,x^9 + 84.689\,x^8 - 681.332\,x^7 + 2985.768\,x^6 - 7913.962\,x^5 + 13248.418\,x^4 - 14099.763\,x^3 + 9256.257\,x^2 - 3411.833\,x + 537.097$$

Salida:

Puntos:	*Derivadas:*
P(1) = 1.00000000000000 *P(2) = 6.00000000000000* *P(4) = 6.50000000000728* *P(6) = 7.00000000000000*	*P^(1)(1) = 9.00000000000000* *P^(1)(2) = 5.00000000000000* *P^(2)(1) = 3.00000000000000* *P^(2)(2) = 7.00000000000001* *P^(3)(1) = 7.00000000000000* *P^(4)(1) = 2.00000000000000*

Ejemplo 3

X	1
Y	1
Y’	0.5
Y’’	1
Y’’’	1.2



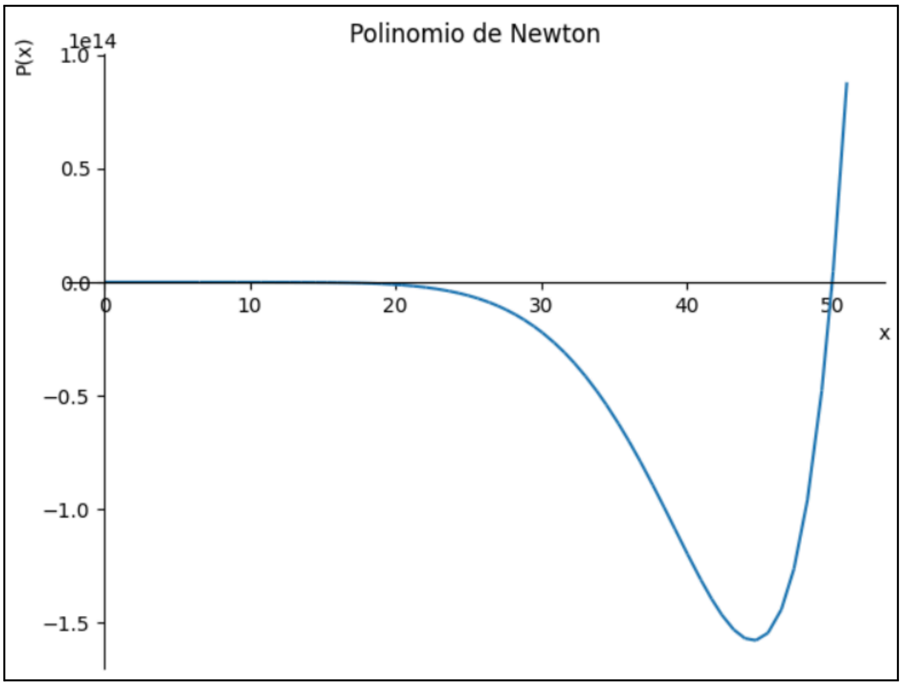
$$P(x) = 0.2\,x^3 - 0.1\,x^2 + 0.1\,x + 0.8$$

Salida:

Puntos: *P(1) = 1.00000000000000*	*Derivadas:* *P^(1)(1) = 0.500000000000000* *P^(2)(1) = 1.00000000000000* *P^(3)(1) = 1.20000000000000*

Ejemplo 4

X	1	2	50
Y	-1	-6	-6.5
Y’	0.5	0.3	
Y”	-0.8	1	
Y”’	π		
Y^{iv}	8		
Y^{v}	0		



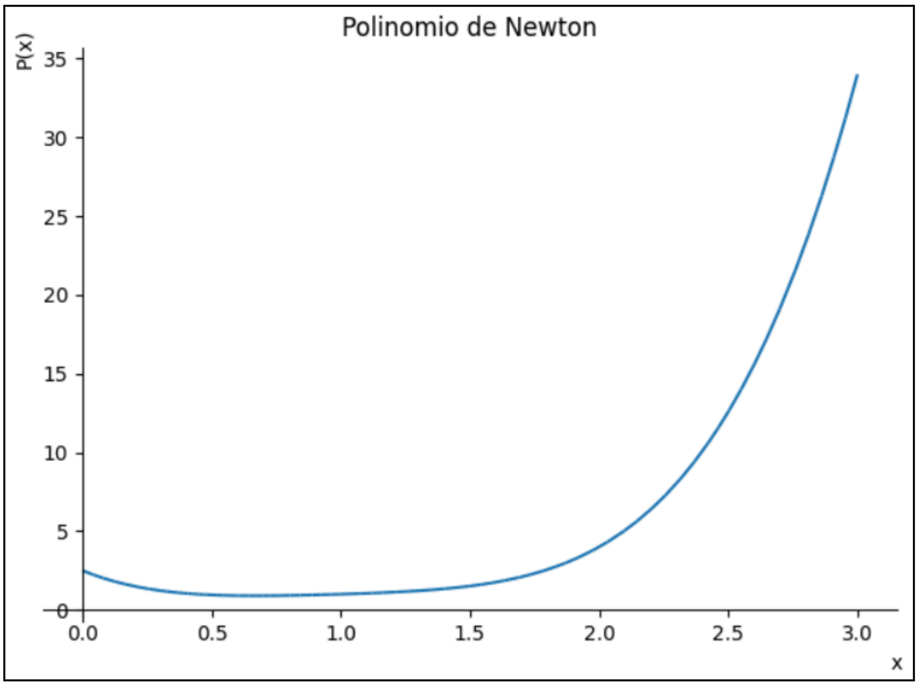
$$P(x) = 2.359\,x^9 - 142.253\,x^8 + 1321.491\,x^7 - 5621.217\,x^6 + 13644.913\,x^5 - 20497.692\,x^4 + 19482.327\,x^3 - 11445.255\,x^2 + 3803.89\,x - 549.563$$

Salida:

Puntos:	*Derivadas:*
P(1) = -1.00000000000000 *P(2) = -6.00000000000000* *P(50) = -6.50000000000000*	*P^(1)(1) = 0.500000000000000* *P^(1)(2) = 0.300000000000006* *P^(2)(1) = -0.800000000000000* *P^(2)(2) = 1.00000000000000* *P^(3)(1) = 3.14159265358979* *P^(4)(1) = 8.00000000000000* *P^(5)(1) = 0*

Ejemplo 5

X	1	2
Y	1	4
Y’	0.5	
Y”	1	
Y”’	1.5	



$$P(x) = 1.75\,x^4 - 6.75\,x^3 + 10.25\,x^2 - 6.75\,x + 2.5$$

Salida:

Puntos: *P(1) = 1.00000000000000* *P(2) = 4.00000000000000*	*Derivadas:* *P^(1)(1) = 0.500000000000000* *P^(2)(1) = 1.00000000000000* *P^(3)(1) = 1.50000000000000*

5. Webgrafía

- https://docs.sympy.org/latest/modules/numeric-computation.html
- https://docs.sympy.org/latest/modules/utilities/lambdify.html
- https://docs.sympy.org/latest/tutorial/basic_operations.html
- https://math.stackexchange.com/questions/677852/how-to-calculate-relative-error-when-true-value-is-zero
- https://numpy.org/doc/stable/reference/generated/numpy.amax.html
- https://numpy.org/doc/stable/reference/generated/numpy.amin.html
- https://numpy.org/doc/stable/reference/generated/numpy.append.html
- https://numpy.org/doc/stable/reference/generated/numpy.argmin.html
- https://numpy.org/doc/stable/reference/generated/numpy.average.html
- https://numpy.org/doc/stable/reference/generated/numpy.dot.html
- https://numpy.org/doc/stable/reference/generated/numpy.split.html
- https://numpy.org/doc/stable/reference/generated/numpy.zeros.html#numpy.zeros
- https://numpy.org/doc/stable/user/absolute_beginners.html
- https://stackoverflow.com/questions/16819222/how-to-return-dictionary-keys-as-a-list-in-python
- https://stackoverflow.com/questions/16819222/how-to-return-dictionary-keys-as-a-list-in-python
- https://stackoverflow.com/questions/30018977/how-can-i-get-a-list-of-the-symbols-in-a-sympy-expression
- https://stackoverflow.com/questions/31547657/how-can-i-solve-system-of-linear-equations-in-sympy
- https://stackoverflow.com/questions/31638508/plot-piecewise-function-in-python
- https://stackoverflow.com/questions/4624970/finding-local-maxima-minima-with-numpy-in-a-1d-numpy-array
- https://stackoverflow.com/questions/574730/python-how-to-ignore-an-exception-and-proceed
- https://stackoverflow.com/questions/6828722/python-set-to-list
- https://stackoverflow.com/questions/986006/how-do-i-pass-a-variable-by-reference
- https://stats.stackexchange.com/questions/86708/how-to-calculate-relative-error-when-the-true-value-is-zero
- https://thispointer.com/python-numpy-select-an-element-or-sub-array-by-index-from-a-ndarray/
- https://www.geeksforgeeks.org/newtons-divided-difference-interpolation-formula/