

# **LABORATORIO 3**

Unidad 4

Nicolás Delgado

Camila Paladines

## **Computación Científica**

Profesor: Hernán Darío Vargas Cardona, PhD

Abril 30 de 2021

## RESUMEN

En este informe se describen los resultados conseguidos al implementar algoritmos en Python para la obtención de respuestas a problemas matemáticos, como lo es el cómputo de las funciones interpolantes para un conjunto de datos, en este caso, mediante el Método de Interpolación Polinomial, el Método de Interpolación de Lagrange, el Método de Interpolación de Newton y la Interpolación Lineal a Trozos. Además, se realizan algunos análisis sobre los métodos utilizados aplicándolos a conjuntos de datos reales, donde se observa su comportamiento dependiendo de diferentes factores. También se calcula la complejidad computacional y la exactitud, con el fin de comparar los métodos y determinar cuál obtuvo mejores resultados para los conjuntos de datos usados.

# ABSTRACT

This report describes the results obtained by implementing algorithms in Python to obtain answers to mathematical problems, such as the computation of interpolating functions for a data set, in this case, by means of the Polynomial Interpolation Method, the Lagrange Interpolation Method, the Newton Interpolation Method and the Linear Piecewise Interpolation. In addition, some analyses are performed on the methods used by applying them to real data sets, where their behavior is observed depending on different factors. The computational complexity and accuracy are also calculated, in order to compare the methods and determine which one obtained better results for the data sets used.

# Contenido

|  |           |
|--|-----------|
| <b>1. Introducción</b>                                 | <b>1</b>  |
| <b>2. Materiales y métodos</b>                         | <b>2</b>  |
| 2.1. Materiales . . . . .                              | 2         |
| 2.2. Métodos . . . . .                                 | 3         |
| 2.2.1. Interpolación . . . . .                         | 3         |
| 2.2.2. Interpolación Polinomial . . . . .              | 3         |
| 2.2.3. Interpolación de Lagrange . . . . .             | 4         |
| 2.2.4. Interpolación de Newton . . . . .               | 4         |
| 2.2.5. Interpolación Lineal a Trozos . . . . .         | 5         |
| <b>3. Resultados de las Simulaciones</b>               | <b>6</b>  |
| 3.1. Ejemplo 1 . . . . .                               | 6         |
| 3.1.1. Interpolación Polinomial . . . . .              | 6         |
| 3.1.2. Interpolación de Lagrange . . . . .             | 7         |
| 3.1.3. Interpolación de Newton . . . . .               | 7         |
| 3.1.4. Interpolación Lineal a Trozos . . . . .         | 8         |
| 3.2. Ejemplo 2 . . . . .                               | 9         |
| 3.2.1. Interpolación Polinomial . . . . .              | 9         |
| 3.2.2. Interpolación de Lagrange . . . . .             | 10        |
| 3.2.3. Interpolación de Newton . . . . .               | 10        |
| 3.2.4. Interpolación Lineal a Trozos . . . . .         | 11        |
| <b>4. Discusión y Análisis</b>                         | <b>12</b> |
| 4.1. Ejemplo 1 . . . . .                               | 12        |
| 4.1.1. Resultados de los Métodos . . . . .             | 12        |
| 4.1.2. Complejidad Computacional . . . . .             | 15        |
| 4.1.3. Límite de los Algoritmos . . . . .              | 16        |
| 4.1.4. Ventajas y Desventajas de los Métodos . . . . . | 16        |
| 4.2. Ejemplo 2 . . . . .                               | 17        |

|  |           |
|--|-----------|
| 4.2.1. Resultados de los Métodos . . . . .             | 17        |
| 4.2.2. Complejidad Computacional . . . . .             | 20        |
| 4.2.3. Límite de los Algoritmos . . . . .              | 21        |
| 4.2.4. Ventajas y Desventajas de los Métodos . . . . . | 21        |
| 4.3. Consideraciones Especiales . . . . .              | 22        |
| <b>5. Conclusiones</b>                                 | <b>23</b> |
| <b>6. Referencias</b>                                  | <b>24</b> |

## Índice de Figuras

|     |  |    |
|-----|--|----|
| 1.  | Funciones resultantes con Interpolación Lineal a Trozos . . . . .      | 5  |
| 2.  | Resultado de Interpolación Polinomial para el Ejemplo 1 . . . . .      | 6  |
| 3.  | Resultado de Interpolación de Lagrange para el Ejemplo 1 . . . . .     | 7  |
| 4.  | Resultado de Interpolación de Newton para el Ejemplo 1 . . . . .       | 7  |
| 5.  | Resultado de Interpolación Lineal a Trozos para el Ejemplo 1 . . . . . | 8  |
| 6.  | Resultado de Interpolación Polinomial para el Ejemplo 2 . . . . .      | 9  |
| 7.  | Resultado de Interpolación de Lagrange para el Ejemplo 2 . . . . .     | 10 |
| 8.  | Resultado de Interpolación de Newton para el Ejemplo 2 . . . . .       | 10 |
| 9.  | Resultado de Interpolación Lineal a Trozos para el Ejemplo 2 . . . . . | 11 |
| 10. | Error (Promedio) para el Ejemplo 1 . . . . .                           | 13 |
| 11. | Error Interpolación Lineal a Trozos para el Ejemplo 1 . . . . .        | 14 |
| 12. | Tiempo de ejecución para el Ejemplo 1 . . . . .                        | 15 |
| 13. | Error (Promedio) para el Ejemplo 2 . . . . .                           | 18 |
| 14. | Error I.L.T para el Ejemplo 2 . . . . .                                | 19 |
| 15. | Tiempo de ejecución para el Ejemplo 2 . . . . .                        | 20 |

## Índice de Tablas

|    |   |    |
|----|---|----|
| 1. | Error (Promedio) para el Ejemplo 1 . . . . .            | 12 |
| 2. | Error (Desviación Estándar) para el Ejemplo 1 . . . . . | 12 |
| 3. | Tiempo de ejecución para el Ejemplo 1 . . . . .         | 15 |
| 4. | Error (Promedio) para el Ejemplo 2 . . . . .            | 17 |
| 5. | Error (Desviación Estándar) para el Ejemplo 2 . . . . . | 17 |
| 6. | Tiempo de ejecución para el Ejemplo 2 . . . . .         | 20 |

## 1. Introducción

La tarea de predicción es un área ampliamente estudiada que tiene diversas aplicaciones en distintos campos. Se han utilizado métodos de mínimos cuadrados como el de Ecuaciones Normales y los ortogonales como el de Transformaciones Householder para solucionar problemas de esta naturaleza. Sin embargo, este problema también ha sido abordado mediante la interpolación, usando diferentes métodos como el Polinomial y los que proponen Lagrange y Newton, además de la Interpolación Lineal a Trozos que usa la noción de dividir y conquistar, que bajo ciertas condiciones obtiene muy buenos resultados tanto en tiempo como en exactitud.

La interpolación es un caso particular de mínimos cuadrados, en el que no sólo se intenta aproximar la función que modele los datos planteados, sino que para los datos de entrenamiento la solución es exacta, lo que permite adentrarse en un nuevo ambiente en el que se pueden computar soluciones y analizar sus resultados desde un punto de vista diferente.

## 2. Materiales y métodos

### 2.1. Materiales

Para el desarrollo de esta unidad se usó Python 3.7, con las siguientes librerías:

- **numpy**. Para funciones matemáticas como el valor absoluto, promedio, desviación estándar, entre otros.
- **pyplot**. Para graficar los datos en el plano y las estadísticas de los métodos con respecto a su exactitud y tiempo de ejecución.
- **time**. Para calcular los tiempos de cómputo de cada uno de los métodos para un conjunto de datos dado.
- **pandas**. Para cargar los datos desde un archivo `.csv` y poder procesarlos.
- **sympy**. Para modelar la variable  $t$  dentro de las operaciones de los métodos de Interpolación de Lagrange y de Interpolación de Newton.

Además, se usaron dos conjuntos de datos para probar los algoritmos en cuanto a exactitud y tiempo de ejecución, estos son:

- **Precio del oro**. En este conjunto de datos se muestran las variaciones diarias del precio del oro en el mercado bursátil. Los datos están desde el 1 de enero de 1970 hasta el 13 de marzo del 2020.
- **Preguntas sobre Python en Stack Overflow**. En este conjunto de datos se muestra la cantidad de preguntas mensuales que se realizan en la plataforma de Stack Overflow relacionadas con el lenguaje de programación Python. Los datos están desde enero del 2009 hasta diciembre del 2019.



## 2.2. Métodos

### 2.2.1. Interpolación

La interpolación consiste en ajustar una función a unos datos, tal que la función sea equivalente a ellos. El problema más sencillo de interpolación en una dimensión es de la siguiente forma:

- Para unos datos:  $(t_i, y_i)$ ,  $i = 1, \dots, n$ , con  $t_1 < t_2 < \dots < t_n$ , se busca una función tal que:

$$f(t_i) = y_i, \quad i = 1, \dots, n.$$

- En general:

$$f(t_i) = \sum_{j=1}^n x_j \phi_j(t_i) = y_i, \quad i = 1, \dots, n.$$

Siendo  $\phi_j$  funciones base y  $f$  la función interpolante.

### 2.2.2. Interpolación Polinomial

La interpolación más sencilla emplea polinomios. Hay un único polinomio de grado  $n - 1$  que pasa a través de los  $n$  puntos  $(t_i, y_i)$ ,  $i = 1, \dots, n$ , donde los  $t_i$  son distintos.

- Por ejemplo, con la base monomial:

$$\phi_j(t) = t^{j-1}, \quad j = 1, \dots, n$$

- El polinomio interpolante tiene la forma:

$$p_{n-1} = x_1 + x_2 t + x_3 t^2 + \dots + x_n t^{n-1}$$

- Los coeficientes  $x_i$  están determinados por el siguiente por el siguiente sistema lineal:

$$\begin{bmatrix} 1 & t_1 & \dots & t_1^{n-1} \\ 1 & t_2 & \dots & t_2^{n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & t_n & \dots & t_n^{n-1} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

### 2.2.3. Interpolación de Lagrange

Para un conjunto de datos  $(t_i, y_i)$ ,  $i = 1, \dots, n$ , las funciones base de Lagrange están dadas por:

$$l_j(t) = \frac{\prod_{k=1, k \neq j}^n (t - t_k)}{\prod_{k=1, k \neq j}^n (t_j - t_k)}$$

Para la base de Lagrange se tiene:

$$l_j(t_i) = \begin{cases} 1, & \text{si } i = j \\ 0, & \text{si } i \neq j \end{cases}$$

Lo que significa que la matriz  $A$  del sistema lineal  $Ax = y$  es la identidad  $I$ . Por lo tanto, en la base de Lagrange el polinomio interpolante tiene la forma:

$$p_{n-1}(t) = y_1 l_1(t) + y_2 l_2(t) + \dots + y_n l_n(t)$$

La forma Lagrangiana hace más fácil la determinación del polinomio interpolante, pero más costosa para evaluar, diferenciar e integrar.

### 2.2.4. Interpolación de Newton

Para un conjunto de datos  $(t_i, y_i)$ ,  $i = 1, \dots, n$ , el polinomio interpolante de Newton tiene la forma:

$$p_{n-1} = x_1 + x_2(t - t_1) + x_3(t - t_1)(t - t_2) + \dots + x_n(t - t_1)(t - t_2) \dots (t - t_{n-1})$$

Por lo tanto, las funciones base de Newton tienen la forma:

$$\phi_j(t) = \prod_{k=1}^{j-1} (t - t_k), \quad j = 1, \dots, n$$

Lo cual forma una matriz  $A$  triangular inferior, por lo que los coeficientes  $x_i$ ,  $i = 1, \dots, n$  se puede hallar mediante sustitución sucesiva hacia adelante  $O(n^2)$ .

### 2.2.5. Interpolación Lineal a Trozos

Para un conjunto de datos  $(t_i, y_i)$  se emplea un polinomio distinto entre cada subintervalo  $[t_i, t_{i+1}]$ . El ejemplo más sencillo es la interpolación lineal, donde cada punto sucesivo está conectado con una línea recta.

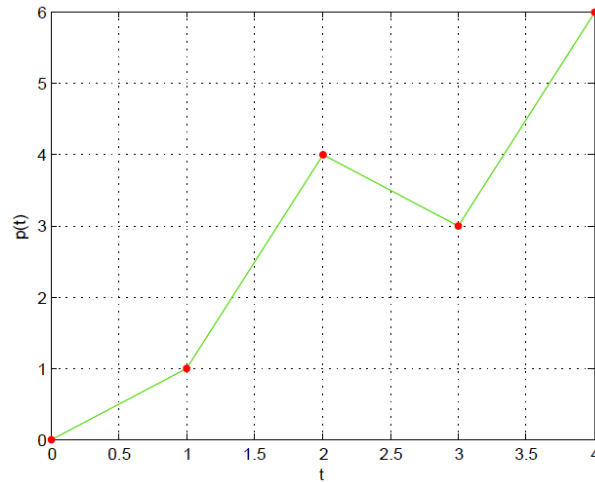


Figura 1: Funciones resultantes con Interpolación Lineal a Trozos

### 3. Resultados de las Simulaciones

#### 3.1. Ejemplo 1

En este ejemplo se tomó el conjunto de datos del precio del oro. Para mostrar los resultados de los métodos de interpolación polinómica se usaron los primeros 100 datos, para el método de Interpolación Lineal a Trozos fueron 300 datos.

Para las simulaciones de los métodos de interpolación polinómica, se obtuvo aproximadamente el mismo polinomio de solución:

$$f(t) = 8,04 \times 10^{-6} t^4 - 0,00165 t^3 + 0,0713 t^2 + 3,47 t + 31,7$$

El error absoluto promedio fue de 12 y su desviación estándar 13,03. Con porcentajes de datos de entrenamiento mayores al 5 % el *overfitting* se va haciendo más notable, por lo que se eligió este límite con el fin de visualizar mejor los datos. A continuación se muestran las gráficas obtenidas por cada método.

##### 3.1.1. Interpolación Polinomial

Los resultados son los siguientes, con un tiempo de ejecución de 0.00010s.

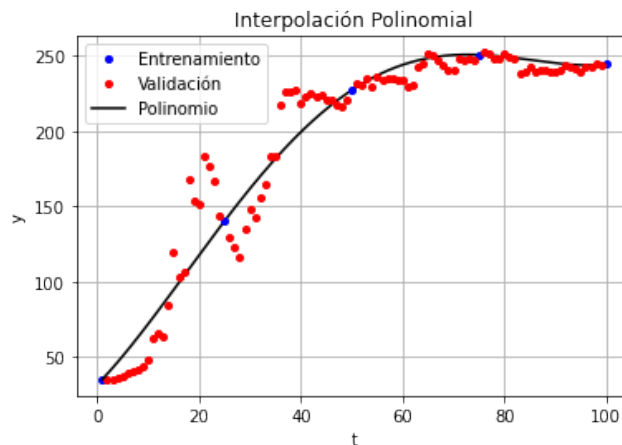


Figura 2: Resultado de Interpolación Polinomial para el Ejemplo 1

### 3.1.2. Interpolación de Lagrange

Los resultados son los siguientes, con un tiempo de ejecución de 0.00053s.

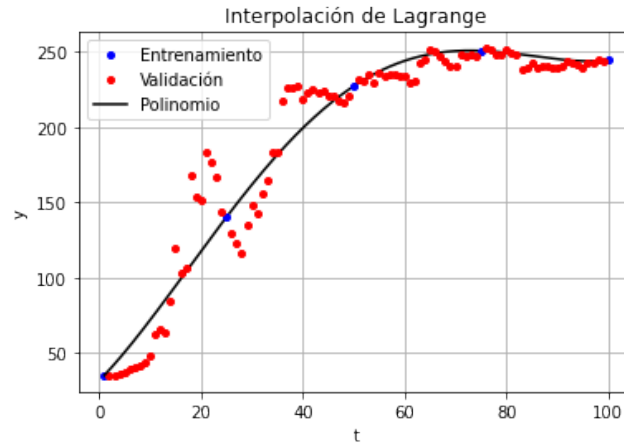


Figura 3: Resultado de Interpolación de Lagrange para el Ejemplo 1

### 3.1.3. Interpolación de Newton

Los resultados son los siguientes, con un tiempo de ejecución de 0.00076s.

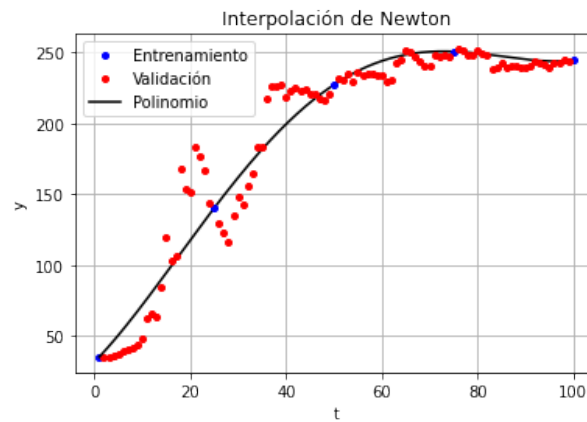


Figura 4: Resultado de Interpolación de Newton para el Ejemplo 1

### 3.1.4. Interpolación Lineal a Trozos

Los resultados se muestran en la siguiente gráfica, con un tiempo de ejecución de 0,00014s. Además, se obtuvo un error promedio de 4,18 y una desviación estándar de 5,54. Fueron tomados los primeros 300 datos del conjunto original, un 50 % de ellos se usaron como datos de entrenamiento.

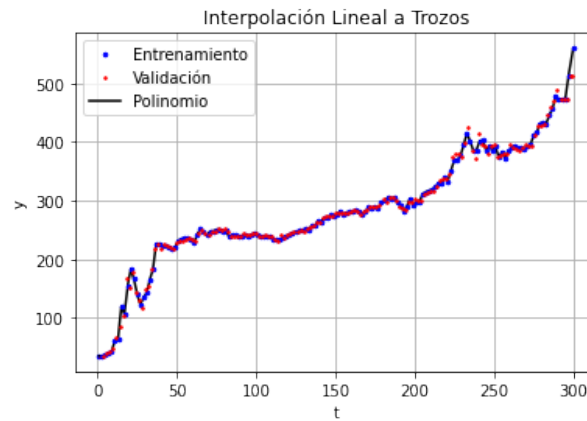


Figura 5: Resultado de Interpolación Lineal a Trozos para el Ejemplo 1

### 3.2. Ejemplo 2

En este ejemplo se tomó el conjunto de datos de la cantidad de preguntas en Stack Overflow sobre Python. Para mostrar los resultados de los métodos de interpolación polinómica se usaron los primeros 50 datos, para el método de Interpolación Lineal a Trozos fueron 100 datos.

Para las simulaciones de los métodos de interpolación polinómica, se obtuvo aproximadamente el mismo polinomio de solución:

$$f(t) = -0,000131 t^5 + 0,0169 t^4 - 0,747 t^3 + 13,9 t^2 + 1,23 t + 617$$

El error absoluto promedio fue de 185,87 y su desviación estándar 192,13. Con porcentajes de datos de entrenamiento mayores al 13 % el *overfitting* se va haciendo más notable, por lo que se eligió este límite con el fin de visualizar mejor los datos. A continuación se muestran las gráficas obtenidas por cada método.

#### 3.2.1. Interpolación Polinomial

Los resultados son los siguientes, con un tiempo de ejecución de 0,00010s.

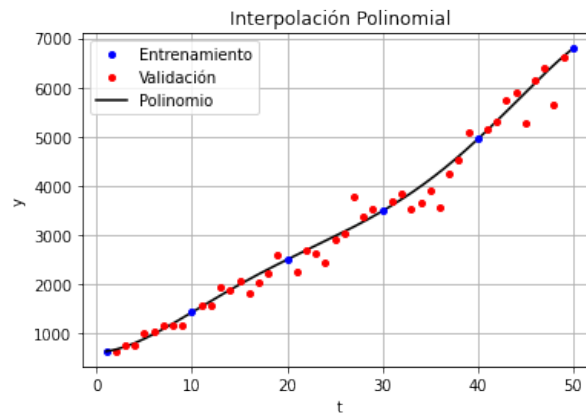


Figura 6: Resultado de Interpolación Polinomial para el Ejemplo 2

### 3.2.2. Interpolación de Lagrange

Los resultados son los siguientes, con un tiempo de ejecución de 0,00042s.

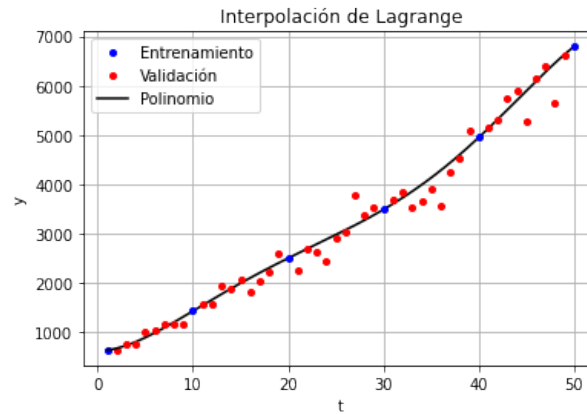


Figura 7: Resultado de Interpolación de Lagrange para el Ejemplo 2

### 3.2.3. Interpolación de Newton

Los resultados son los siguientes, con un tiempo de ejecución de 0,00083s.

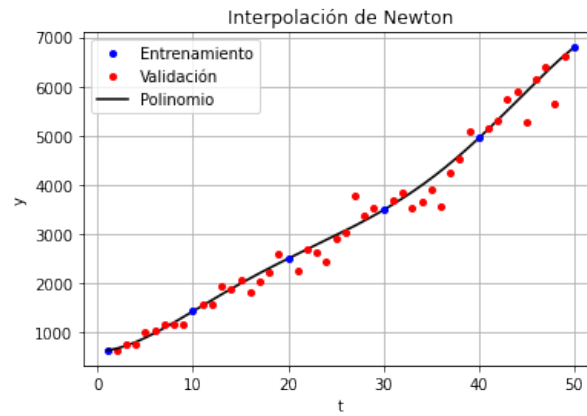


Figura 8: Resultado de Interpolación de Newton para el Ejemplo 2



### 3.2.4. Interpolación Lineal a Trozos

Los resultados se muestran en la siguiente gráfica, con un tiempo de ejecución de 0,00006s. Además, se obtuvo un error promedio de 384,69 y una desviación estándar de 403,93. Fueron tomados los primeros 100 datos del conjunto original, un 50 % de ellos se usaron como datos de entrenamiento.

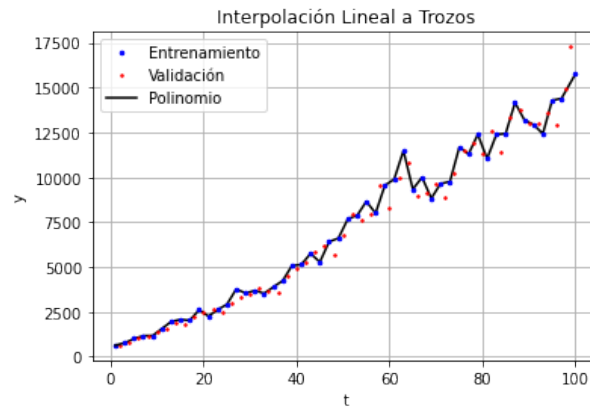


Figura 9: Resultado de Interpolación Lineal a Trozos para el Ejemplo 2

## 4. Discusión y Análisis

Al usar los métodos implementados en conjuntos de datos reales, se pueden establecer las siguientes consideraciones para cada uno de los conjuntos de datos.

### 4.1. Ejemplo 1

#### 4.1.1. Resultados de los Métodos

En las siguientes tablas se pueden observar el error promedio y la desviación estándar para cada uno de los métodos en los diferentes porcentajes de datos de entrenamiento.

| Porcentaje | Polinomial | Lagrange  | Newton    | A trozos |
|------------|------------|-----------|-----------|----------|
| 2          | 54.28757   | 54.28757  | 54.28757  | 27.04524 |
| 4          | 13.60550   | 13.60550  | 13.60550  | 12.83203 |
| 6          | 16.46299   | 16.46299  | 16.46299  | 10.86429 |
| 8          | 27.39471   | 27.39471  | 27.39471  | 8.06256  |
| 10         | 20.13007   | 20.13007  | 20.13007  | 8.19420  |
| 12         | 83.53486   | 83.53487  | 83.53487  | 7.17168  |
| 14         | 139.24712  | 139.24711 | 139.24711 | 6.86167  |
| 16         | 144.39879  | 144.39462 | 144.39462 | 5.88772  |
| 18         | 692.04359  | 691.11685 | 691.11685 | 5.60457  |
| 20         | 399.61933  | 405.79704 | 405.79705 | 4.72656  |
| 22         | 241.60323  | 835.26957 | 835.26956 | 4.82831  |

Cuadro 1: Error (Promedio) para el Ejemplo 1

| Porcentaje | Polinomial | Lagrange   | Newton     | A trozos |
|------------|------------|------------|------------|----------|
| 2          | 33.63395   | 33.63395   | 33.63395   | 27.73523 |
| 4          | 12.33908   | 12.33908   | 12.33908   | 16.12841 |
| 6          | 16.13774   | 16.13774   | 16.13774   | 15.16090 |
| 8          | 32.08074   | 32.08074   | 32.08074   | 11.70270 |
| 10         | 25.18292   | 25.18292   | 25.18292   | 10.13837 |
| 12         | 160.66042  | 160.66042  | 160.66042  | 8.68420  |
| 14         | 297.70646  | 297.70645  | 297.70645  | 8.40970  |
| 16         | 408.06436  | 408.05199  | 408.05199  | 7.61686  |
| 18         | 1989.84255 | 1987.58270 | 1987.58270 | 7.17666  |
| 20         | 1531.86645 | 1546.86361 | 1546.86365 | 5.64702  |
| 22         | 893.34363  | 2929.87727 | 2929.87722 | 6.28921  |

Cuadro 2: Error (Desviación Estándar) para el Ejemplo 1

El error promedio se puede observar mejor en la siguiente gráfica:

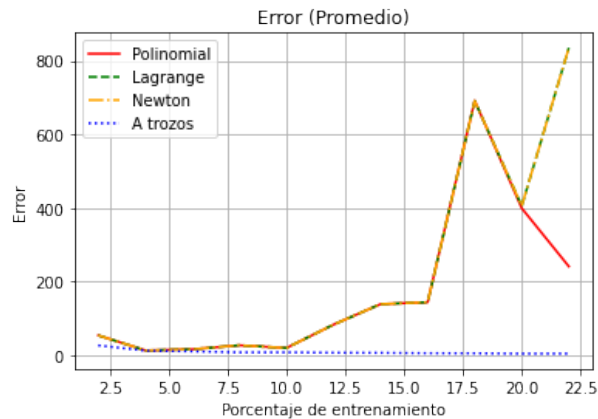


Figura 10: Error (Promedio) para el Ejemplo 1

Se puede notar que el error de los tres métodos polinómicos es aproximadamente el mismo hasta el 20 %. El porcentaje en el que se obtienen los mejores resultados es 4 % (en este rango donde el *overfitting* es poco) para estos métodos.

Por otro lado se puede observar que los métodos polinómicos casi siempre obtienen un mayor error que la Interpolación Lineal a Trozos, esto se debe a que para la Interpolación Lineal a Trozos se usaron 300 datos en total, y como este método obtiene muy buenos resultados cuando hay bastantes datos de entrenamiento, este obtuvo menor error que los demás.

Se analizó el error del método de Interpolación Lineal a Trozos en los diferentes porcentajes (2 % a 99 %) y se obtuvo el siguiente resultado:



Figura 11: Error Interpolación Lineal a Trozos para el Ejemplo 1

Aquí va el análisis del error de ILT

#### 4.1.2. Complejidad Computacional

En la siguiente tabla se puede observar el tiempo de ejecución para cada uno de los métodos en los diferentes porcentajes de datos de entrenamiento.

| Porcentaje | Polinomial | Lagrange | Newton  | A trozos |
|------------|------------|----------|---------|----------|
| 2          | 0.00009    | 0.00624  | 0.00092 | 0.00002  |
| 4          | 0.00008    | 0.00648  | 0.00247 | 0.00003  |
| 6          | 0.00012    | 0.02058  | 0.00415 | 0.00002  |
| 8          | 0.00014    | 0.03040  | 0.00624 | 0.00003  |
| 10         | 0.00016    | 0.04813  | 0.00951 | 0.00003  |
| 12         | 0.00019    | 0.07841  | 0.01190 | 0.00005  |
| 14         | 0.00024    | 0.10396  | 0.01240 | 0.00004  |
| 16         | 0.00031    | 0.15572  | 0.01501 | 0.00005  |
| 18         | 0.00269    | 0.20322  | 0.03030 | 0.00006  |
| 20         | 0.00051    | 0.24930  | 0.04023 | 0.00007  |
| 22         | 0.00052    | 0.31692  | 0.04629 | 0.00007  |

Cuadro 3: Tiempo de ejecución para el Ejemplo 1

Que se pueden observar mejor en la siguiente gráfica:

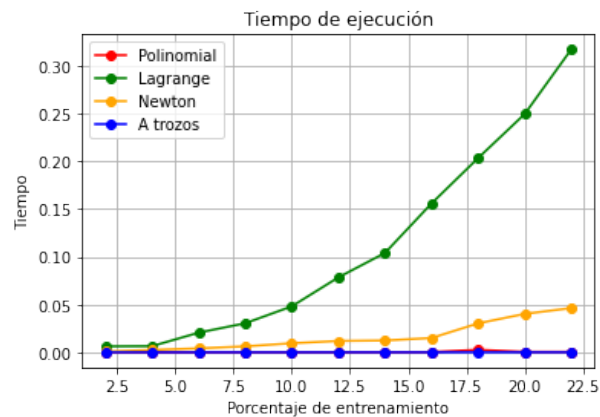


Figura 12: Tiempo de ejecución para el Ejemplo 1

Aquí se puede notar que la complejidad computacional de la Interpolación de Lagrange es mucho mayor que las demás ya que

#### 4.1.3. Límite de los Algoritmos

#### 4.1.4. Ventajas y Desventajas de los Métodos

- Interpolación Polinomial
  - Ventajas:
  - Desventajas:
- Interpolación de Lagrange
  - Ventajas:
  - Desventajas:
- Interpolación de Newton
  - Ventajas:
  - Desventajas:
- Interpolación Lineal a Trozos
  - Ventajas:
  - Desventajas:

## 4.2. Ejemplo 2

### 4.2.1. Resultados de los Métodos

En las siguientes tablas se pueden observar el error promedio y la desviación estándar para cada uno de los métodos en los diferentes porcentajes de datos de entrenamiento.

| Porcentaje | Polinomial | Lagrange   | Newton     | A trozos   |
|------------|------------|------------|------------|------------|
| 2          | 2617.91837 | 2617.91837 | 2617.91837 | 1115.84962 |
| 4          | 536.91667  | 536.91667  | 536.91667  | 531.00063  |
| 6          | 196.24886  | 196.24886  | 196.24886  | 535.80045  |
| 8          | 264.95258  | 264.95258  | 264.95258  | 539.48810  |
| 10         | 257.75459  | 257.75459  | 257.75459  | 561.79091  |
| 12         | 185.86920  | 185.86920  | 185.86920  | 467.49874  |
| 14         | 258.38228  | 258.38228  | 258.38228  | 474.84510  |
| 16         | 525.57655  | 525.57655  | 525.57655  | 490.34439  |
| 18         | 459.86781  | 459.86781  | 459.86781  | 463.18984  |
| 20         | 360.44556  | 360.44556  | 360.44556  | 483.38750  |
| 22         | 336.56560  | 336.56560  | 336.56560  | 461.80897  |

Cuadro 4: Error (Promedio) para el Ejemplo 2

| Porcentaje | Polinomial | Lagrange   | Newton     | A trozos  |
|------------|------------|------------|------------|-----------|
| 2          | 1736.65173 | 1736.65173 | 1736.65173 | 619.13234 |
| 4          | 334.85866  | 334.85866  | 334.85866  | 551.22753 |
| 6          | 205.11392  | 205.11392  | 205.11392  | 526.68298 |
| 8          | 195.47044  | 195.47044  | 195.47044  | 528.34115 |
| 10         | 159.38026  | 159.38026  | 159.38026  | 498.91963 |
| 12         | 192.12679  | 192.12679  | 192.12679  | 462.42058 |
| 14         | 259.74503  | 259.74503  | 259.74503  | 478.07572 |
| 16         | 545.73771  | 545.73771  | 545.73771  | 497.32177 |
| 18         | 553.61391  | 553.61391  | 553.61391  | 451.38092 |
| 20         | 425.96527  | 425.96527  | 425.96527  | 444.12198 |
| 22         | 512.92947  | 512.92947  | 512.92947  | 434.44899 |

Cuadro 5: Error (Desviación Estándar) para el Ejemplo 2

El error promedio se puede observar mejor en la siguiente gráfica:

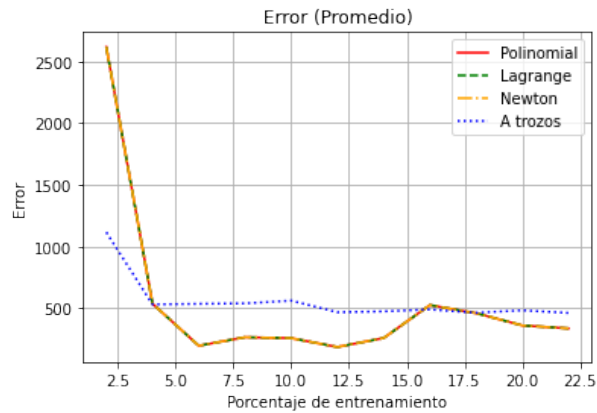


Figura 13: Error (Promedio) para el Ejemplo 2

Se puede notar que el error de los tres métodos polinómicos es aproximadamente el mismo hasta el 22 %. El porcentaje en el que se obtienen los mejores resultados es 12 % (en este rango donde el *overfitting* es poco) para estos métodos.

Además se puede observar que los métodos polinómicos casi siempre obtienen un menor error que la Interpolación Lineal a Trozos, esto se debe a que aunque para la Interpolación Lineal a Trozos se usaron más datos de entrenamiento, estos no fueron suficientes y no se obtuvo un mejor resultado que los demás.

Se analizó el error del método de Interpolación Lineal a Trozos en los diferentes porcentajes (2% a 99%) y se obtuvo el siguiente resultado:



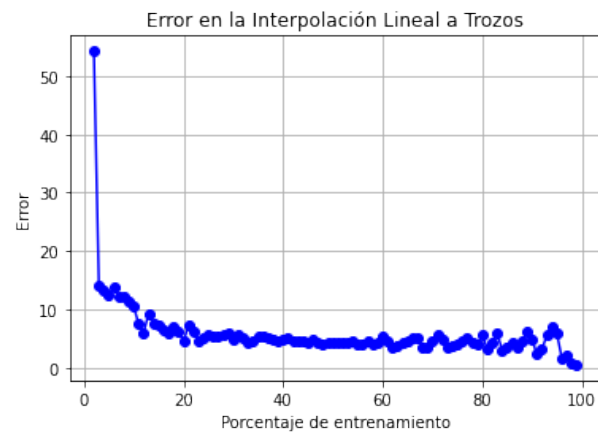


Figura 14: Error I.L.T para el Ejemplo 2

Aquí va el análisis del error de ILT

#### 4.2.2. Complejidad Computacional

En la siguiente tabla se puede observar el tiempo de ejecución para cada uno de los métodos en los diferentes porcentajes de datos de entrenamiento.

| Porcentaje | Polinomial | Lagrange | Newton  | A trozos |
|------------|------------|----------|---------|----------|
| 2          | 0.00008    | 0.00002  | 0.00018 | 0.00001  |
| 4          | 0.00011    | 0.00133  | 0.00147 | 0.00001  |
| 6          | 0.00011    | 0.00323  | 0.00256 | 0.00001  |
| 8          | 0.00013    | 0.00521  | 0.00258 | 0.00002  |
| 10         | 0.00010    | 0.00891  | 0.00358 | 0.00002  |
| 12         | 0.00012    | 0.01189  | 0.00428 | 0.00004  |
| 14         | 0.00015    | 0.01658  | 0.00511 | 0.00002  |
| 16         | 0.00015    | 0.02376  | 0.00639 | 0.00002  |
| 18         | 0.00016    | 0.02944  | 0.00717 | 0.00002  |
| 20         | 0.00019    | 0.03822  | 0.00792 | 0.00003  |
| 22         | 0.00017    | 0.05336  | 0.00875 | 0.00003  |

Cuadro 6: Tiempo de ejecución para el Ejemplo 2

Que se pueden observar mejor en la siguiente gráfica:

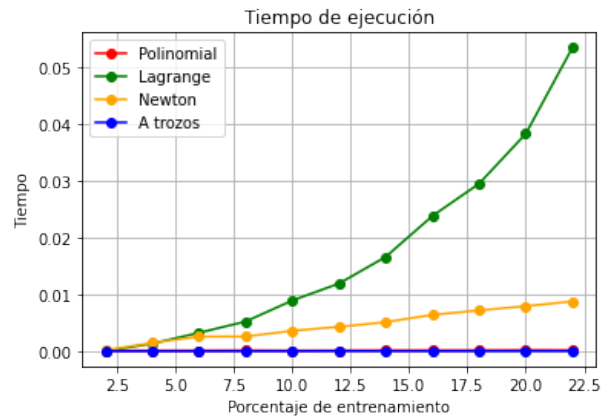


Figura 15: Tiempo de ejecución para el Ejemplo 2

Aquí se puede notar que la complejidad computacional de la Interpolación de Lagrange es mucho mayor que las demás ya que

#### 4.2.3. Límite de los Algoritmos

#### 4.2.4. Ventajas y Desventajas de los Métodos

- Interpolación Polinomial
  - Ventajas:
  - Desventajas:
- Interpolación de Lagrange
  - Ventajas:
  - Desventajas:
- Interpolación de Newton
  - Ventajas:
  - Desventajas:
- Interpolación Lineal a Trozos
  - Ventajas:
  - Desventajas:

### 4.3. Consideraciones Especiales

En los métodos de Interpolación Polinomial e Interpolación de Newton se tenían problemas de aproximación de las funciones, es decir, estos no daban una solución exacta sino aproximada, debido a que estos usaban o el método de eliminación de Gauss o el método de Sustitución Sucesiva Hacia Atrás, que usaban divisiones o multiplicaciones nativas de Python, que son inestables con flotantes y la representación binaria de los mismos dentro del lenguaje de programación.

Sin embargo, estos problemas pudieron ser solucionados usando la librería `numpy`, que provee una solución *exacta* a problemas de álgebra lineal, con la instrucción para solucionar sistemas de ecuaciones lineales `numpy.linalg.solve(A, b)`. El uso de esta función fue aceptada por el profesor.

## 5. Conclusiones

Se comprendió que al usar la computación para resolver problemas matemáticos se puede profundizar en el comportamiento de los métodos tanto a nivel de exactitud como a nivel de complejidad, con lo que se logra aprender más sobre ellos e identificar en qué caso usarlos y de qué manera se puede aprovechar mejor.

Se logró comprender las diferentes formas de obtener el polinomio interpolante y la importancia de los métodos aprendidos para diversas aplicaciones.

Es relevante analizar los diferentes métodos definidos para un problema, tanto en exactitud como en complejidad, ya que esto permite decidir, según el problema y los datos, cuál es el más apropiado para realizar los cálculos.

## 6. Referencias

- Material del curso, disponible en BlackBoard
- Bornemann, F., 2016. *Numerical linear algebra*. 1st ed. *Simson, W.*
- Mathews, J., Fink, K., Fernández Carrión, A. & Contreras Márquez, M., 2011. *Métodos Numéricos con MATLAB*. 3rd ed. Madrid: *Pearson Prentice Hall*.
- Precio del oro (conjunto de datos del ejemplo 1)
- Preguntas sobre Python en StackOverflow (conjunto de datos del ejemplo 2)
- Librería Numpy
- Librería Pyplot (Matplotlib)
- Librería Time
- Librería Pandas
- Librería Sympy