Informe Laboratorio 2 Computación Científica

Santiago Uribe Pastás
Pontificia Universidad Javeriana Cali

Septiembre-Octubre 2020

1. Abstract

En el siguiente documento se presenta un informe acerca de la implementación en lenguaje de programación Python de 2 métodos, Ecuaciones Normales y Transformaciones Householder, los cuales se utilizan para resolver problemas de mínimos cuadrados lineales. Posteriormente se presentarán los métodos utilizados para la realización de dichas implementaciones junto con los resultados y su respectivo análisis.

2. Introducción

Para abordar las implementaciones acerca de los métodos anteriormente mencionados es importante revisar un poco teoría acerca los mínimos cuadrados lineales y sobre los métodos.

2.1. Mínimos Cuadrados Lineales

Esta es una técnica en la que dados un conjunto de pares ordenados (variable independiente y variable dependiente) aquí los tomaremos como (t_i, y_i), se intenta encontrar la función continua que mejor se aproxime a los datos, es decir, que genere un "mejor ajuste". Un problema de mínimos cuadrados lineal se puede escribir en notación matricial así:

$$Ax \approx b$$

Donde $A_{i,j} = \phi_j(t_i)$ y $b_i = y_i$, siendo ϕ una función que depende de t.

2.2. Ecuaciones Normales

Sea $Ax = b$ un sistema lineal no singular, así que $x = A^{-1}b$ es la solución a dicho sistema si y solo si el vector residual $r = b - Ax$ satisface que $r = 0$. Aqui lo que se busca es la minimización de la norma Euclidiana cuadrática del vector

r, pues se tiene que $Ax \approx b$. Dicha minimización se reduce a un sistema lineal cuadrado que corresponde a:

$$A^T Ax = A^T b$$

La anterior ecuación se conoce como el sistema de ecuaciones normales. La matriz $A^T A$ es no singular, por lo que el sistema de ecuaciones normales tiene solución única, y es la solución al problema original de mínimos cuadrados. De igual manera, la matriz $A^T A$ es simétrica y positiva definida, por lo cual se puede computar su descomposición de Cholesky:

$$A^T A = LL^T$$

donde L es una matriz triangular inferior y L^T triangular superior. La solución x del problema de mínimos cuadrados se puede computar solucionando dos sistemas triangulares: $Ly = A^T b$ y $L^T x = y$. La idea del método de ecuaciones normales es convertir el problema así: Rectangular $\rightarrow$ Cuadrado $\rightarrow$ Triangular.

2.3. Transformaciones Householder

Una transformación Householder es una matriz de la forma:

$$H = I - 2\frac{vv^T}{v^T v}$$

donde v es un vector sin ceros. Se sabe que H es una matriz simétrica y ortogonal, por lo que $H = H^T = H^{-1}$. Dado un vector a (el cual hace referencia a columnas de una matriz A), se desea encontrar un vector v tal que:

$$Ha = \begin{bmatrix} \alpha \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \alpha \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \alpha e_1$$

Usando la ecuación de H se llega a que $v = a - \alpha e_1$ y para preservar la norma, se tiene que tomar $\alpha = \pm||a||_2 = \sqrt{a^T a}$, y el signo se escoge de forma que no haya cancelación.

Mediante transformaciones Householder sucesivas, se puede introducir ceros columna por columna, debajo de la diagonal de una matriz A, para llevarla a la forma triangular. Este proceso sucesivo genera lo siguiente:

$$H_n \cdots H_1 A = \begin{bmatrix} R \\ O \end{bmatrix}$$

donde R es una matriz triangular superior. El producto de transformaciones Householder sucesivas $H_n \cdots H_1$ es también ortogonal. Por lo tanto si tomamos $Q^T = H_n \cdots H_1$ y $Q = H_1^T \cdots H_n^T$ entonces:

$$A = Q \begin{bmatrix} R \\ O \end{bmatrix}$$

Esto quiere decir que al final se debe resolver el siguiente problema de mínimos cuadrados triangular equivalente:

$$\begin{bmatrix} R \\ O \end{bmatrix} x \approx Q^T b$$

3. Materiales y métodos

Para la realización del laboratorio se hizo uso del lenguaje de programación Python junto con librerías que facilitaron en algunos aspectos la implementación del código, como lo son numpy y matplotlib. Además se hizo uso de las diapositivas y vídeos de clase para solucionar dudas acerca de teoría y métodos de resolución. Debido a que los algoritmos se debían ejecutar con datos reales, se escogieron dos bases de datos. La primera (data1.txt) hace referencia a la evolución de muertos por Covid 19 en Colombia a lo largo de 50 días (Desde el 11/08/2020 hasta el 29/09/2020) [1]. Y la segunda (data2.txt) hace referencia a la evolución del precio del petróleo crudo WTI a lo largo de 50 meses (Agosto 2016 hasta Septiembre 2020) [2]. Ambas bases de datos se encuentran en el anexo.

Para la implementación del método ecuaciones normales primeramente tenemos que armar la matriz A y el vector b para representar el problema en notación matricial con los datos dados, $(t_1, y_1), \cdots, (t_m, y_m)$, y un n que representa el numero de parámetros del polinomio, así:

$$Ax = \begin{bmatrix} 1 & t_1 & \dots & t_1^{j-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & t_m & \dots & t_m^{j-1} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \approx \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix} = b$$

Con $1 \leq j \leq n$. Entonces A seria de tamaño mxn y b de tamaño mx1. Ya con la matriz A y el vector b encontramos A^T para tener $A^T A$ y aplicar descomposicion de Cholesky para obtener la matriz L y posteriormente calcular L^T, asi se cumple $A^T A = LL^T$, ya con estas nuevas matrices se procede a resolver los sistemas $Ly = A^T b$ y $L^T x = y$ con sustitución sucesiva hacia adelante y sustitución sucesiva hacia atrás respectivamente. Y así en el vector x quedan los coeficientes del polinomio de ajuste. Cabe resaltar que esta implementación tiene un problema y es que $n \leq 12$; de lo contrario la descomposición de Cholesky no se podrá realizar.

Para la implementación del método transformaciones Householder se procede a armar la matriz A y el vector b como se mencionó anteriormente en el método de ecuaciones normales. Seguido a esto, se procede a calcular la matriz H respectiva a cada columna para hacer la aniquilación debajo de la diagonal de la matriz A siguiendo el procedimiento mencionado en la sección 2.3. Después al

quedar el sistema triangular equivalente

$$\begin{bmatrix} R \\ O \end{bmatrix} x \approx \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

Se resuelve el sistema $Rx = b_1$ por sustitución sucesiva hacia atrás. Y así en el vector x quedan los coeficientes del polinomio de ajuste. Cabe resaltar que en la implementación, al hacer las transformaciones Householder sucesivas los valores de la matriz fueron redondeados a 5 decimales para eliminar la notación científica que genera Python. Previo a hallar R se redondea a 3 decimales para manejar los valores como se manejaron en clase.

4. Resultados

Los algoritmos fueron ejecutados en un computador con un procesador i5 de 7ma generación y 16GB de RAM.
Los códigos se ejecutan así: `python3 main.py < data1.txt`

4.1. Tiempos de ejecución

A continuación se presentan los tiempos de ejecución de los algoritmos con diferentes valores de parámetros (n).

Parámetros (n)	Ecuaciones Normales	Householder
5	0.471	0.437
8	0.477	0.463
12	0.489	0.477
19	-	0.486

Cuadro 1: Tiempos de ejecución en los diferentes métodos con data1.txt

Parámetros (n)	Ecuaciones Normales	Householder
5	0.457	0.513
8	0.466	0.525
12	0.476	0.564
19	-	0.572

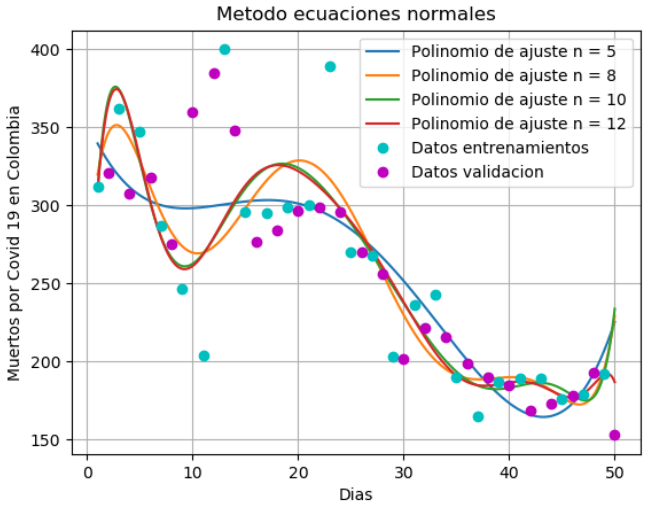
Cuadro 2: Tiempos de ejecución en los diferentes métodos con data2.txt

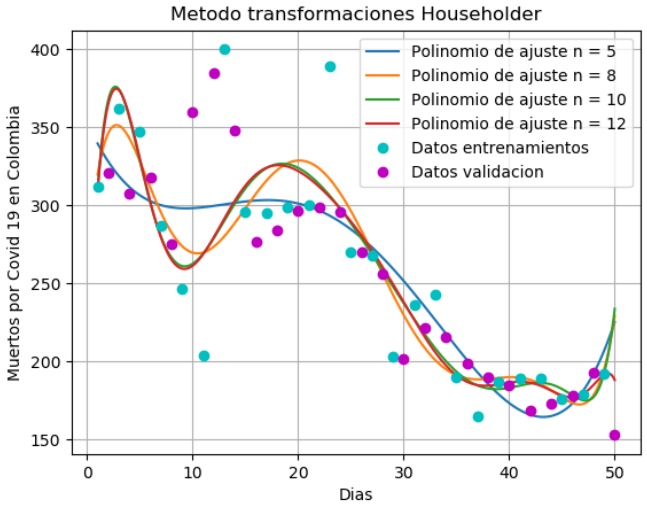
Como se puede apreciar en las tablas de tiempos el algoritmo de transformaciones Householder posee un mejor rendimiento de tiempo en comparación con el de ecuaciones normales en data1.txt y el ecuaciones normales posee un mejor tiempo en data2.txt. Sin embargo, cabe resaltar la restricion que tiene el metodo de ecuaciones normales anteriormente mencionada de $n \leq 12$.

4.2. Gráficas

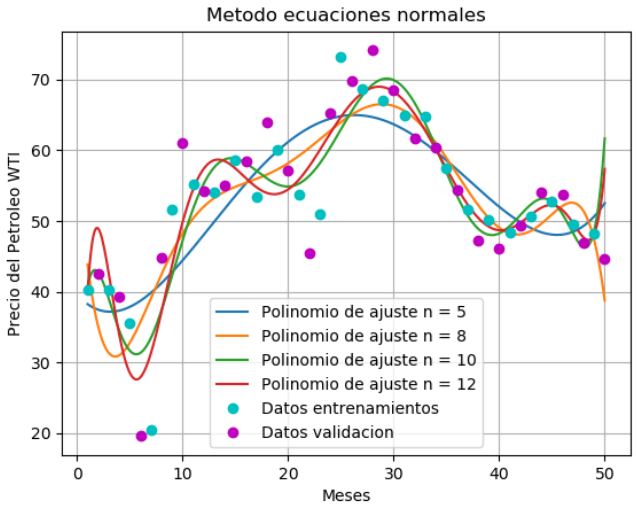
A continuación se muestran los resultados gráficos de los algoritmos para algunas simulaciones.

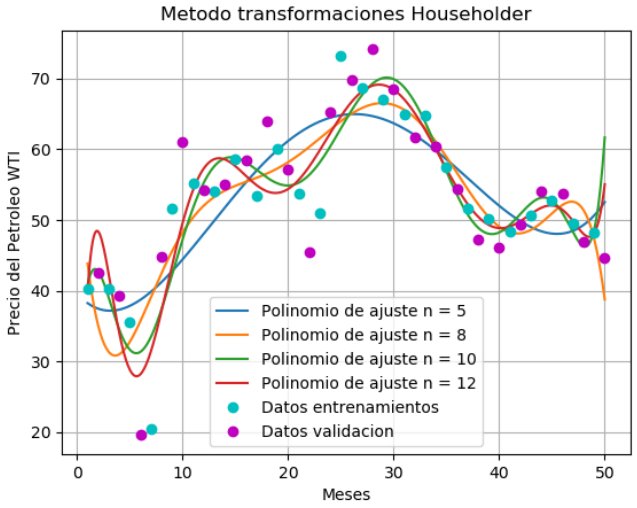
- data1.txt





- data2.txt





Como se puede observar en las gráficas a medida que que el numero de parámetros (n) aumenta el polinomio de ajuste se va acercando a los datos reales. Sin embargo, en algunos puntos el polinomio no se ajusta bien, esto ocurre debido a la alta variabilidad que poseen los datos.

4.3. Error absoluto promedio y desviación del error

A continuación se presentan el error medio y la desviación del error para las simulaciones realizadas.

- data1.txt

Parametros (n)	Error Promedio	Desv. Error
5	17.53	21.79
8	25.49	27.18
10	26.22	28.12
12	25.92	28.20

Cuadro 3: Error medio y desviación del error con método ecuaciones normales

Parametros (n)	Error Promedio	Desv. Error
5	17.53	21.79
8	25.49	27.18
10	26.22	28.12
12	25.92	28.17
19	43.46	56.56

Cuadro 4: Error medio y desviación del error con transformaciones Householder

- data2.txt

Parametros (n)	Error Promedio	Desv. Error
5	5.73	5.20
8	4.97	4.37
10	3.95	3.88
12	4.16	3.47

Cuadro 5: Error medio y desviación del error con método ecuaciones normales

Parametros (n)	Error Promedio	Desv. Error
5	5.73	5.20
8	4.97	4.37
10	3.95	3.88
12	4.17	3.45
19	6.79	16.36

Cuadro 6: Error medio y desviación del error con transformaciones Householder

5. Conclusión

Como se pudo observar los métodos de ecuaciones normales y transformaciones Householder sirven para hacer un ajuste de datos bastante aceptable, sin embargo, estos métodos están bastante limitados. El de ecuaciones normales particularmente porque no puede recibir un parámetro $n > 12$ porque la descomposición de Cholesky no funciona. Además, ambos modelos están condicionados a que la variabilidad de los datos no puede ser muy alta, pues como se puede apreciar en la sección 4.3 al tener unos datos con gran variabilidad el error promedio es bastante elevado, esto se nota en data1.txt.

Referencias

[1] "COVID-19 Coronavirus Data", European Union Open Data Portal, 2020. [Online]. Available: `https://data.europa.eu/euodp/en/data/dataset/covid-19-coronavirus-data`.

[2] "Histórico del precio de Petróleo crudo WTI", Investing.com, 2020. [Online]. Available: `https://es.investing.com/commodities/crude-oil-historical-data`.