

**Trabajo de Algebra 2**

Integrantes: Félix Pérez

Hernán Puelles

Fecha: 17/06/2014

**Introducción**

En este trabajo explicaremos como funciona nuestro programa que realiza la aproximación de mínimos cuadrados para sistemas de ecuaciones y para la aproximación de puntos a funciones lineales, cuadráticas y cúbicas.

Decidimos programar utilizando el lenguaje de programación Python ya que es el lenguaje que habíamos utilizado más recientemente y con el cual manejábamos mejor algunas librerías que serían de utilidad.

A pesar de que conocíamos en Python funciones que realizaban directamente las operaciones que requeríamos, decidimos programar de todas formas cada una de las operaciones necesarias de la forma en que las realizaríamos nosotros, a costo de eficiencia en el código, con el objetivo de cumplir con la finalidad del trabajo, el cual era aprender a calcular y entender cómo funcionan los mínimos cuadrados.

Python Version: 3.4.1 x32

Modulos: Numpy 1.8.1 (librería matemática)

Matplotlib: 1.3.1 (librería grafica)

Archivos:

* MinCuad.py:

Incluye menú de usuario interactivo y opciones graficas.

* MinCuad\_func.py:

Incluye sólo las funciones de operaciones algebraicas con comentarios de cada función.

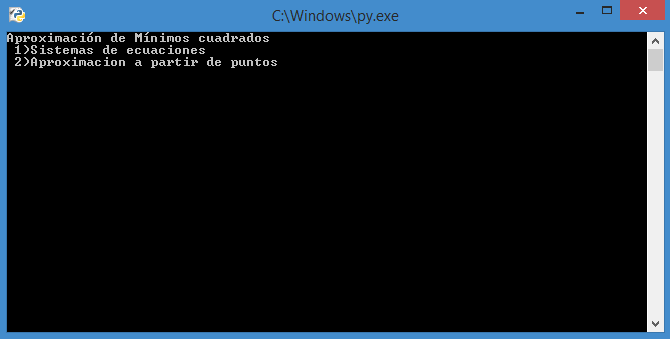
Funcionamiento

El usuario elige si desea calcular sistemas de ecuaciones o aproximar a puntos:

-Para sistemas de ecuaciones deberá ingresar la cantidad de filas y columnas, luego ingresar la matriz de coeficientes seguida de la matriz de soluciones.

Si se pueden calcular los mínimos cuadrados, el programa calculará dicha solución.

-Para puntos, el usuario ingresa los puntos que desea aproximar, dependiendo de la cantidad de coordenadas "x" distintas el usuario tendrá la opción de calcular la aproximación a una recta, a ecuación cuadrática y/o cúbica, además el programa puede calcular cual es la ecuación que mejor se aproxima.



Especificaciones

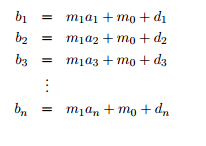
Para sistemas de ecuaciones:

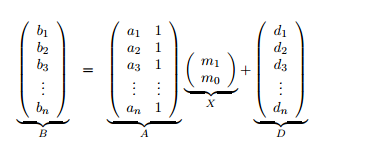
Dada una matriz de coeficientes "A" de orden (*nxm*) y una matriz de soluciones "B" orden (*nx1*), el sistema tiene solución única mediante mínimos cuadrados si y sólo si el rango de la matriz A es igual a m, en otras palabras, el número de filas linealmente independientes de A es igual al número de columnas, en definitiva

A es una matriz cuadrada y podemos entonces calcular su inversa.

Por lo que la solución única mediante mínimos cuadrados está dada por:

Para puntos:

Sea un conjunto de puntos () con i= [1,2....n] Si existen al menos m+1 puntos con coordenadas x distintas construimos un sistema de la forma:

Que es equivalente a:

Con lo cual resolvemos mediante mínimos cuadrados utilizando la formula previamente descrita.

Implementación

Como ya hemos explicado el funcionamiento del método de aproximación de mínimos cuadrados, ahora es necesario describir como realizamos cada paso en nuestro código, para ello programamos las funciones que nos permiten resolver la ecuación de mínimos cuadrados:

A continuación se describirán las funciones utilizadas para sistemas de ecuaciones:

* mult(matrizA,matrizB):

Multiplicación de matrices, primero verifica que 2 matrices A(*nxm*) y B(*pxq*) sean multiplicables es decir m=p, luego crea una nueva matriz de dimensiones (*nxq*) con los nuevos elementos producto de la multiplicación de *AxB.*

* transp(matriz):

Transpuesta de una matriz, dada una matriz de dimensiones (nxm) creamos una matriz transpuesta de dimensiones (*mxn*) invirtiendo las filas por columnas.

* det(matriz):

Determinante de una matriz, dada una matriz de , realiza el producto de la diagonal principal menos la diagonal secundaria, es decir Para matrices mayores aplicamos teorema de Laplace, utilizando la primera fila calculamos la suma de cada elemento multiplicado por el determinante (de forma recursiva) de su matriz menor complementaria, o submatriz, con el correspondiente signo.

* cofactores(matriz):

Esta función obtiene la matriz de cofactores substituyendo en cada termino *a[i][j]* de la matriz por su cofactor que corresponde al determinante de la submatriz con signo correspondiente a la posición en que se encuentre.

* inversa(matriz):

Para obtener la matriz inversa, si está es de dimensión 2x2 se aplica la formula:

Para matrices de orden superior aplicamos:

Donde

* subm(matriz, i, j):

Función submatriz, devuelve la matriz resultante de eliminar una fila "i" y columna "j".

* mincuad(A,B):

Teniendo todas las funciones necesarias podemos resolver la ecuación de mínimos cuadrados:

A continuación se describirá las funciones utilizadas para la aproximación por puntos:

* aprox(pts,grf):

Dado un conjunto de puntos de la forma [()] Primero obtenemos las coordenadas (*x, y*) por separado en x1, y1 respectivamente, formamos la matriz A1 de la forma , calculamos los mínimos cuadrados entre la matriz construida A1 y la matriz de coordenadas "y", la matriz solución que obtenemos corresponde a los valores de pendiente y coeficiente de posición de la recta, estos valores los podemos evaluar en la ecuación de la recta usando las coordenadas x originales, obteniendo las coordenadas "y" correspondientes a la recta aproximada, teniendo esto podemos obtener la suma de los cuadrados de los residuos De forma similar operamos para encontrar la ecuación cuadrática y cúbica que más se aproxima, construyendo matrices de la forma A2= y

A3= respectivamente.

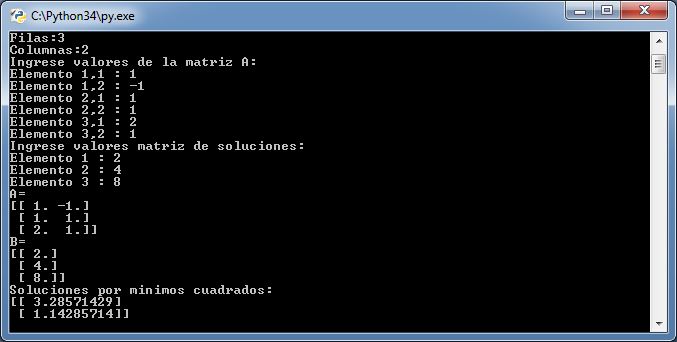
Ejemplo de uso:

Queremos calcular la solución al siguiente sistema usando mínimos cuadrados:

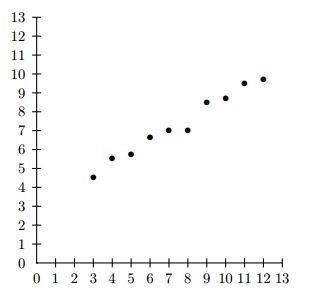
Entonces la matriz A de coeficientes y la matriz B de soluciones:

*con*buscamos resolver:

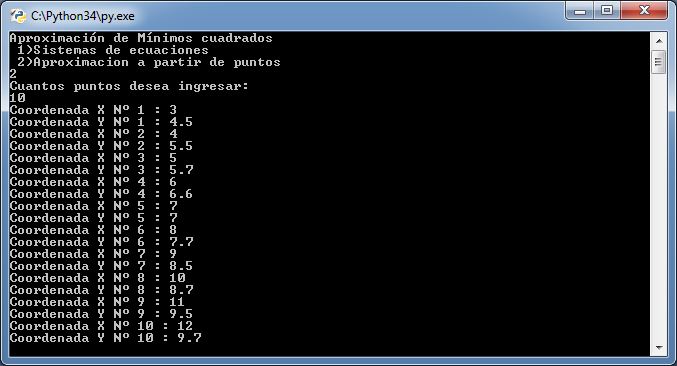
Ingresamos estos datos al programa y obtenemos la solución:



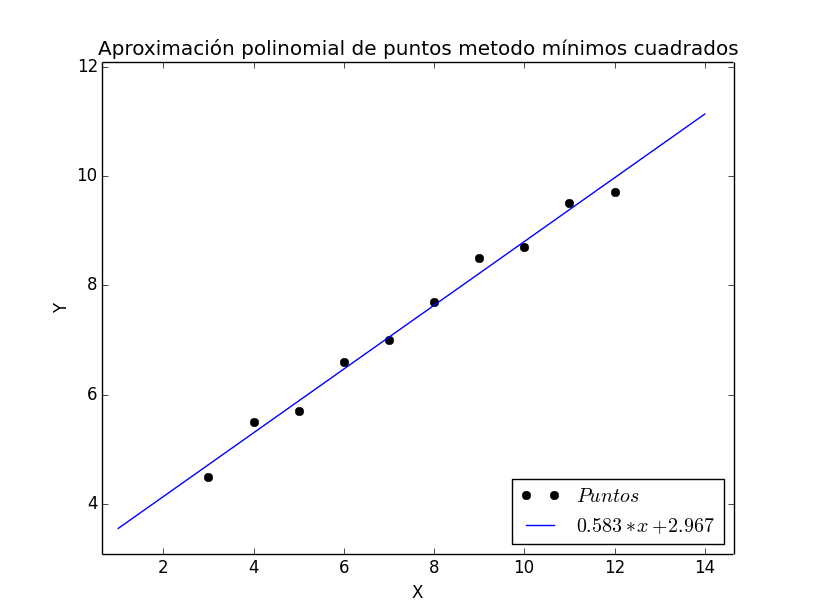
Ejemplo para puntos:



|  |  |
| --- | --- |
| X | Y |
| 3 | 4.5 |
| 4 | 5.5 |
| 5 | 5.7 |
| 6 | 6.6 |
| 7 | 7 |
| 8 | 7.7 |
| 9 | 8.5 |
| 10 | 8.7 |
| 11 | 9.5 |
| 12 | 9.7 |

Ingresamos los datos:

Aproximamos a la recta:



Obtuvimos el grafico y la ecuación de la recta:

**Conclusión**

Podemos decir de forma honesta que estamos bastante satisfechos con los resultados de nuestro trabajo, consideramos que cumplimos el objetivo de aprender a calcular y entender los mínimos cuadrados. El hecho de programar cada función nos sirvió mucho para repasar y grabarnos en la memoria la forma en que realizamos las operaciones con matrices y si bien nos encontramos algunas dificultades al momento de llamar las funciones a partir de datos ingresados, debido a las condiciones que debían cumplir estas, como por ejemplo que debe tener inversa para poder calcular una solución única por mínimos cuadrados, o la cantidad de puntos con coordenadas X distintas que nos determina qué tipo de funciones podemos encontrar; estás dificultades fueron las que más nos ayudaron a entender cómo y cuándo podemos encontrar una solución por mínimos cuadrados.