Segundo Examen Parcial*

José Tecún Cano Caal, 201703690
[1]Escuela de Ciencias Físicas y Matemáticas, Departamento de Física,
Universidad de San Carlos, Edificio T7, Ciudad Universitaria, Zona 12, Guatemala.

Para mejorar el entendimiento de la utilización de métodos numéricos a través de herramientas de programación, se proponen 2 problemas distintos en los cuales se debe: analizar, crear una metodología de solución, e implementar esa metodología en un programa .c

I. OBJETIVOS

- Implementar una solución numérica ante el problema que se le presente
- Crear la documentación necesaria para cada programa elaborado
- Aplicar los conocimientos adquiridos sobre el lenguaje C
- Aplicar los conocimientos adquiridos sobre gnuplot

II. PLANTEAMIENTO DEL PROBLEMA

A. Problema 1: Mínimos Cuadrados

- A continuación se presentan los valores obtenidos de forma experimental de volumen y presión, con una incerteza de 0.1 in^3 y 0.2 $\frac{lb}{in^3}$ respectivamente.

Volumen (in^3)	Presión ($\frac{lb}{in^3}$)
54.3	61.2
61.8	49.2
72.4	37.6
88.7	28.4
118.6	19.2
194.0	10.2

De acuerdo con los principios de la termodinámica, estas variables se relacionan de la siguiente manera:

$$PV^a = b \tag{1}$$

- Genere un programa que cumpla con lo siguiente:

* Crear una gráfica que compare los valores tabulados con la recta que mejor se aproxima al comportamiento real.

$$y = mx + b \tag{2}$$

* Estime el valor de la presión (usando la aproximación lineal) cuando V = 100 in^3.

B. Problema 2: Método de Bisección

- Usando el método de Bisección, encuentre una de las raíces de la ecuación

$$f(x) = \frac{Sin(x)}{x} \tag{3}$$

- Genere un programa que cumpla con lo siguiente:

* Estime el valor de la raíz a la ecuación

* Crear una gráfica de la ecuación

III. MARCO TEÓRICO

A. Método de Mínimos Cuadrados

- El Método de Mínimos cuadrados es una técnica de análisis numérico enmarcada dentro de la optimización matemática, en la que, dados un conjunto de pares ordenados (con una variable independiente y otra dependiente), y una familia de funciones, se busca encontrar la función continua, dentro de esa familia, que mejor se aproxime a los datos. Un "mejor ajuste", de acuerdo con el criterio de mínimo error cuadrático. Este método puede ser lineal o no lineal.

El método de Mínimos Cuadrados Lineal ocurre cuando la función buscada está compuesta por una combinación lineal de los parámetros escogidos.

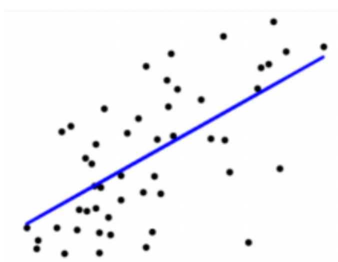
* Laboratorio de Simulación

$$f(x,c) = \sum_{i=1}^{m} c_i \phi_i(x) \tag{4}$$

Donde ϕ es una función dependiente de x.

- El método de Mínimos Cuadrados Lineales es uno de los más utilizados en análisis de data observacional y/o experimental. Es útil para generar estimaciones en análisis de regresión lineal.

  Para un modelo de regresión lineal, la familia de funciones de la cual partimos es de la forma $y = mx + b$.



La regresión lineal provee fórmulas para encontrar tanto la pendiente m como el intersecto b, de estas rectas, a partir de los datos observacionales tabulados.

$$m = \frac{\sum_{k=1}^{n}(x_k y_k) - (\sum_{k=1}^{n} x_k) * (\sum_{k=1}^{n} y_k)}{n * (\sum_{k=1}^{n} x_k^2) - (\sum_{k=1}^{n} x_k)^2} \tag{5}$$

$$b = \frac{(\sum_{k=1}^{n} y_k) - m * (\sum_{k=1}^{n} x_k)}{n} \tag{6}$$

Donde n es el número de pares ordenados.

- Dado que se está lidiando con datos experimentales, siempre existirá una cierta incerteza que debe tomarse en cuenta. Por lo tanto la recta de regresión lineal también debe llevar estas incertezas.

$$y = (m \pm \Delta m)x + (b \pm \Delta b) \tag{7}$$

El método de Mínimos Cuadrados también provee las fórmulas para sacar estas incertezas.

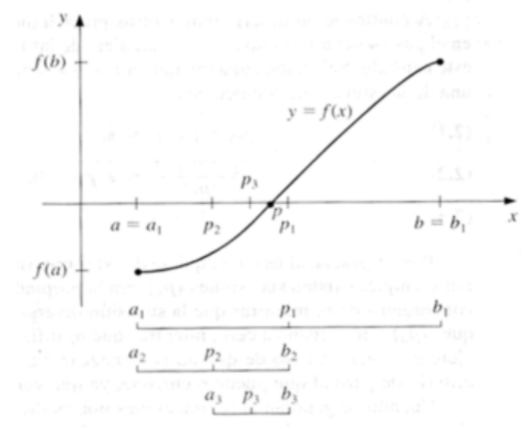
$$\Delta m = \frac{\sqrt{n}\epsilon}{\sqrt{n * (\sum_{k=1}^{n} x_k^2) - (\sum_{k=1}^{n} x_k)^2}} \tag{8}$$

$$\Delta b = \frac{\epsilon}{\sqrt{n}} \tag{9}$$

Donde n es el número de pares ordenados, y ϵ es la incerteza en la variable independiente.

B. Método de Bisección

- El método de bisección es un algoritmo de búsqueda de raíces de funciones, que trabaja dividiendo un intervalo dado a la mitad, y seleccionando el subintervalo que tiene la raíz. El resultado se perfecciona mientras más iteraciones de este método se realicen.



El Método de Bisección se basa en el teorema del valor intermedio, el cual establece que toda función continua $f(x)$ en un intervalo cerrado $[a, b]$ toma todos los valores que se hallan entre $f(a)$ y $f(b)$.

Esto es que todo valor entre $f(a)$ y $f(b)$ es la imagen de al menos un valor en el intervalo $[a, b]$. En caso de que $f(a)$ y $f(b)$ tengan signos opuestos, el valor cero sería un valor intermedio entre ambos, por lo que con certeza existe un p en $[a, b]$ que cumple $f(p) = 0$. De esta forma, se asegura la existencia de al menos una solución de la ecuación $f(x) = 0$.

- El procedimiento es el siguiente:

  Dada una función continua $f(x)$ en el intervalo $[a, b]$, tal que $f(a) * f(b) < 0$, calculamos un punto medio inicial p_0.

$$p_0 = \frac{a+b}{2} \tag{10}$$

A continuación, se evalúa $f(p)$, y se obtiene el producto con $f(a)$.

- Si $f(p) * f(a) < 0$, reemplazamos el valor de p por el valor de b, y repetimos el cálculo.
- Si $f(p) * f(a) > 0$, reemplazamos el valor de p por el valor de a, y repetimos el cálculo
- Si $f(p) * f(a) = 0$, hemos obtenido la respuesta.

- La incerteza correspondiente, es dada por la siguiente ecuación:

$$\epsilon = \frac{p_n - p_{n-1}}{p_n} * 100\,\% \tag{11}$$

Si existieran más de una raíz en el intervalo entonces el método sigue siendo convergente pero no resulta tan fácil caracterizar hacia qué raíz converge el método.

C. Gnuplot

- Gnuplot es un programa de interfaz de línea de comandos para generar gráficas de dos y tres dimensiones de funciones, datos y ajustes de datos

  gnuplot puede producir resultados directamente en la pantalla, o en muchos formatos de archivos gráficos, incluidos: png, eps, svg, jpeg, entre otros. También es capaz de producir código en el sistema de composición de textos y gráficos LaTeX que se puede incluir directamente en los documentos de LaTeX, haciendo uso de las fuentes de LaTeX y las potentes capacidades de notación de fórmulas.

  Este programa contiene un catálogo de funciones por defecto que pueden ser ploteadas por el usuario al escribir el comando **plot** en consola. Estas son algunas de las funciones:

  * abs(x) plotea el valor absoluto de x, o la norma de x en caso de ser un valor complejo
  * acos(x) plotea el coseno inverso de x
  * acosh(x) plotea el coseno hiperbólico de x
  * asin(x) plotea el seno inverso de x
  * asinh(x) plotea el seno hiperbólico inverso de x

- Sin embargo, gnuplot es mayormente usado para plotear archivos previamente creados. Para realizar este procedimiento, se debe escribir el nombre del archivo (usando _ en lugar de espacio) y su terminación. Por ejemplo, **tabla_2.txt**. Se puede incluso plotear 2 o más archivos, al separar los nombres con espacios.

  Estas gráficas tomaran pares ordenados, y los ingresarán en una gráfica. Tomando como variable independiente la primera columna, y como variable dependiente la segunda columna, o la columna que el usuario elija. Para elejirla debe escribirse el comando **using 1:n** donde n es la n-ésima columna en el archivo.

  Una vez la gráfica es generada, gnuplot permite al usuario guardarla como un archivo .jpg, .pdf, o .tex.

- La interfaz de gnuplot permite al usuario personalizar sus gráficas, eligiendo el color de las funciones que se trabajan, el grosor de las líneas, el intervalo (tanto en x como en y) en el cual la gráfica será creada, así como la forma en que los pares ordenados de un archivo de texto aparecerán en pantalla, pudiendo ir desde simples puntos, hasta cruces, barras, vectores, y mucho más.

IV. METODOLOGÍA

A. Problema 1

- Se comenzó creando una serie de diagramas de flujo para poder facilitar la orientación a la hora de crear el programa.

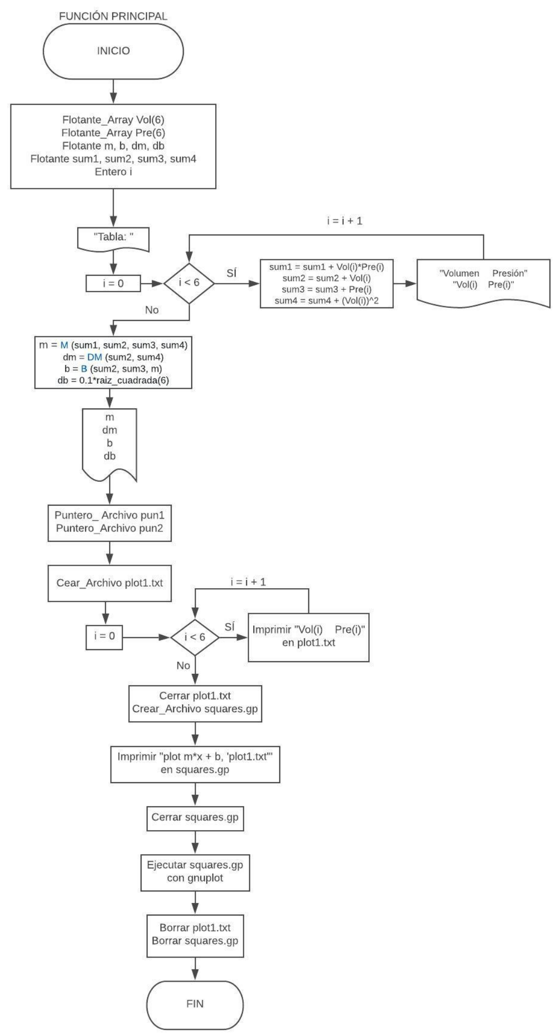


Figura 1: Diagrama de Flujo de la Función Principal

Este programa utiliza 4 funciones: la principal, y 3 funciones auxiliares que sirven para el cálculo de diversos valores para la regresión lineal: la pendiente **m**, el intersecto **b**, y la incerteza en la pendiente Δ**m**.

Se utilizaron ciclos **for** en repetidas ocasiones dentro de la función principal para realizar muchos de los procesos que involucraban los datos tabulados.

- Imprimir los datos en consola.
- Almacenar los datos en archivos de texto para su posterior ploteo.
- El cálculo de las diferentes sumatorias utilizadas en las ecuaciones (5), (6), y (8), que serán utilizadas posteriormente en las funciones auxiliares.

- En la función **M**, se utilizan los datos de las 4 sumatorias. Estos son posteriormente operados en la misma forma que aparecen en la ecuación (5), y su resultado, denominado como la variable **res**, es devuelto una vez el proceso se lleva a cabo y la función no detecta ningún error. Este valor **res** es el que tomará la variable **m**.

$$m = -0.3212 \tag{12}$$

- Para la función **B**, se utilizan los datos de las sumatorias 2 y 3. Los datos se operan tal como aparecen en la ecuación (6). El resultado, que adquiere el nombre de la variable **res**, es devuelto una vez el proceso se lleva a cabo y la función no detecta ningún error. La variable **b** pasa entonces a adquirir su valor.

$$b = 65.88 \tag{13}$$

- Por último, dentro de la función **DM**, se utilizan los datos de las sumatorias 2 y 4. Se procede a operarlas como en la ecuación (8). La variable local **res** almacena el resultado, y este es devuelto una vez el proceso se lleva a cabo y la función no detecta ningún error. Su valor es pasado a la variable **dm** dentro de la función principal.

$$dm = 0.0009 \tag{14}$$

$$db = 0.04 \tag{15}$$

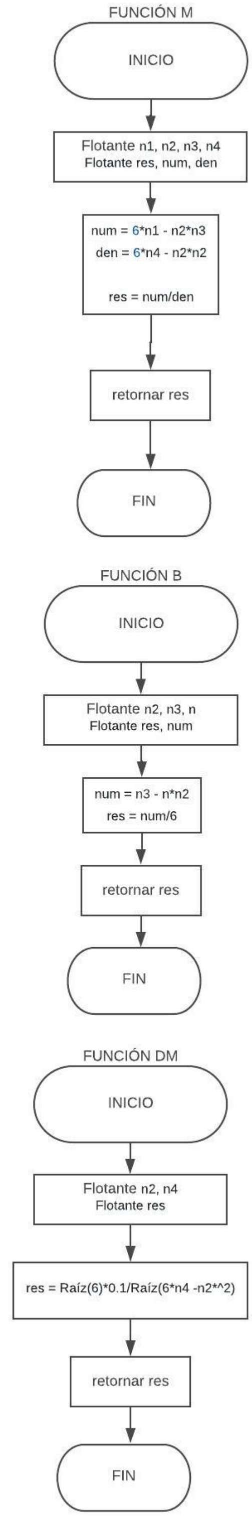


Figura 2: Diagrama de Flujo de las Sub-Funciones

Una vez hecho esto, la función principal pasa a imprimir, uno por uno, los datos de **m**, **b**, **dm**, y **db**, el último de los cuales es calculado en la función main, ya que no requiere de una función auxiliar.

Estos datos son impresos, junto con encabezados para indicarle al usuario cuál de ellos es cuál, y por último, se pasa a imprimir la ecuación de recta completa, con incertezas incluidas.

$$y = (-0.3212 \pm 0.0009)x + (65.88 \pm 0.04) \tag{16}$$

Asimismo, se imprime el valor aproximado de la presión, si el volumen fuera igual a 100 in^3.

$$P_{100} = 33.8\frac{lb}{in^3} \tag{17}$$

- A continuación, se procede a crear la gráfica en gnuplot. Esto se hace creando 2 archivos, y ordenándole a gnuplot que los ejecute. Eso se logra con las herramientas proveidas por la librería **stdlib.h**.

  Se empieza creando 2 punteros que puedan referir a los archivos deseados. Para ello se escribe la palabra **FILE** para declarar el tipo de puntero que serán.

  Luego, se le asigna el valor de los punteros, para que creen los archivos deseados, con el nombre de **plot1.txt** y **squares.gp** respectivamente.

  El primer archivo almacenará los datos tabulados de presión y volumen. Con el comando **fprintf**, y la ayuda de un ciclo **for**, la tarea resulta más fácil. Una vez el ciclo ya haya ingresado todos los datos tabulados, se procede a cerrarlo. Para luego abrir el segundo archivo: **squares.gp**.

  squares.gp puede ser ejecutado por gnuplot desde consola, pero también es posible hacerlo desde un archivo .c. Para el caso de este archivo, se imprimió un pequeño *script*, en el cual se le ordena al archivo que imprima la tabla (tomando la información de **plot1.txt**), así como la función de recta sacada a partir de la regresión lineal con mínimos cuadrados.

  Después, se cierra el archivo, para después ejecutarlo. Se utilizó el comando **-p**, para que la gráfica persista aún cuando los archivos son cerrados o eliminados, que es el siguiente paso.

  Una vez se le ordena a gnuplot que ejecute **squares.gp**, tanto él como **plot1.txt** son borrados, en pos de dejar el directorio intacto. La gráfica, así como los datos impresos en consola, permanecerán ahí hasta que sean manualmente cerrados.

B. Problema 2

- Para el programa que utiliza el método de bisección, se comenzó realizando los diagramas de flujo de 2 funciones. Una principal, y una auxiliar.

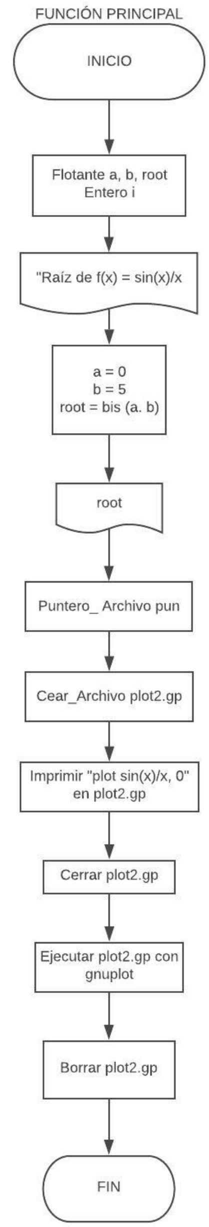


Figura 3: Diagrama de Flujo de la Función Principal

La función principal comienza declarando las variables del intervalo inicial [a,b]. En este caso, dado que la función tiene raíces en cada múltiplo de π, y que una valuación directa en 0 podría causar problemas por indeterminación, se eligió el intervalo [1,5].

La función auxiliar, llamada convenientemente **BIS**, se encargará de hacer el método numérico de bisección.

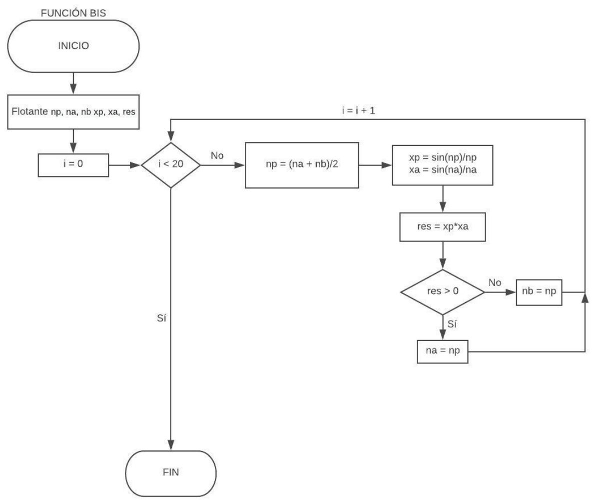


Figura 4: Diagrama de Flujo de la Función Auxiliar

- En la función auxiliar, se declaran 4 variables locales: **np**, **xp**, **xa**, **res**, y se toman los valores de los extremos en el intervalo. Esto para ingresarlos en un loop **for**.

  La variable **np** será el "*punto medio*" de la función. Después se valúa tanto en **np**, como en **na**. A continuación, se multiplican ambos resultados, y dependiendo de si el producto es negativo o positivo, se interacambia el valor de **np** con a, o con b, respectivamente, para luego recalcular **np**.

  Se utiliza un ciclo **for** y no un ciclo **while** (el cual se ejecutaría mientras el producto no sea igual a 0), porque la raíz de esta función es un irracional, y por tanto el producto de **xp** con **xa** nunca sería cero. Así que se optó por un ciclo de 20 iteraciones para llegar a un resultado aproximado.

  Finalmente, cuando el **for** llegue a la vigésima iteración, se cerrará el ciclo, y el valor que **np** haya obtenido en esa vigésima iteración, pasa a la variable **res**, que la retornará a la función principal si no existe ningún error en el proceso.

- Por último, la función principal pasará el valor retornado por **BIS**, a la variable local **root**. Luego, después de haber impreso un encabezado para orientar al usuario, se imprime el resultado de la raíz.

- A continuación, se procede a graficar en gnuplot. Esto este caso, solo se crea un archivo .gp.

  Se empieza creando el puntero correspondiente, al escribir la palabra **FILE** antes del $*$ y el nombre del puntero.

  Después se asigna el valor, con el archivo **plot2.gp**.

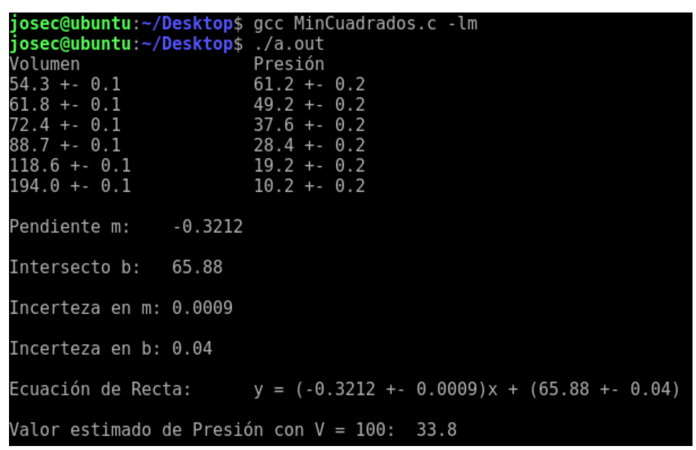
  Con el archivo creado se imprimió otro *script*, en el cual se le ordena al programa que imprima la función del problema 2, junto con el adicional de la función $x = 0$. Esto con el objetivo de que se facilite la comparación de lo graficado con lo calculado.

  Después, se cierra el archivo, para después ejecutarlo, recordando escribir **-p** para que la gráfica no se cierre.

  Una vez más, se le ordena a gnuplot que ejecute **plot2.gp**, y de inmediato el archivo es borrado. La gráfica, así como los datos impresos en consola, permanecerán ahí hasta que sean manualmente cerrados.

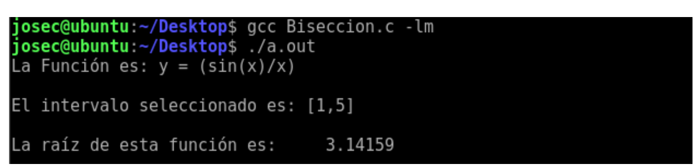
V. RESULTADOS

- Ambos programas, al haber utilizado funciones de la librería **math.h**, necesitaron que se escbribiera el comando **-lm** al ser compiladas para funcionar debidamente. Por lo demás, funcionaron como es debido, siendo posible incluso personalizar las gráficas creadas por **gnuplot**



```
josec@ubuntu:~/Desktop$ gcc MinCuadrados.c -lm
josec@ubuntu:~/Desktop$ ./a.out
Volumen                 Presión
54.3 +- 0.1             61.2 +- 0.2
61.8 +- 0.1             49.2 +- 0.2
72.4 +- 0.1             37.6 +- 0.2
88.7 +- 0.1             28.4 +- 0.2
118.6 +- 0.1            19.2 +- 0.2
194.0 +- 0.1            10.2 +- 0.2

Pendiente m:    -0.3212

Intersecto b:   65.88

Incerteza en m: 0.0009

Incerteza en b: 0.04

Ecuación de Recta:      y = (-0.3212 +- 0.0009)x + (65.88 +- 0.04)

Valor estimado de Presión con V = 100:  33.8
```

Figura 5: Programa 1 luego de ser ejecutado



```
josec@ubuntu:~/Desktop$ gcc Biseccion.c -lm
josec@ubuntu:~/Desktop$ ./a.out
La Función es: y = (sin(x)/x)

El intervalo seleccionado es: [1,5]

La raíz de esta función es:     3.14159
```

Figura 6: Programa 2 luego de ser ejecutado

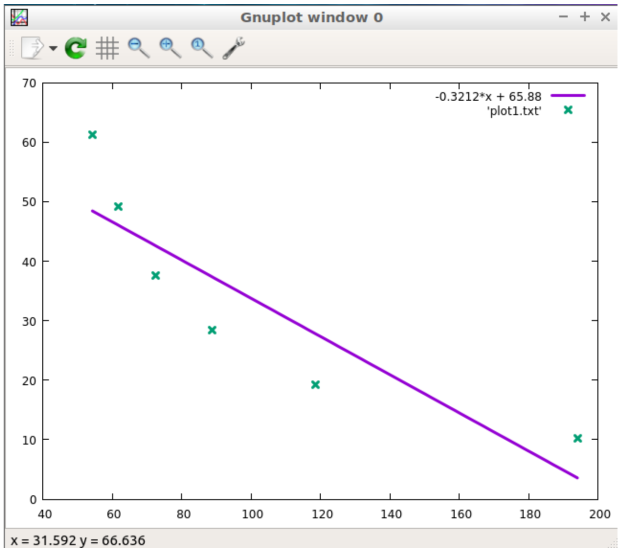


Figura 7: Gráfica comparativa de los datos ploteados vs la regresión lineal

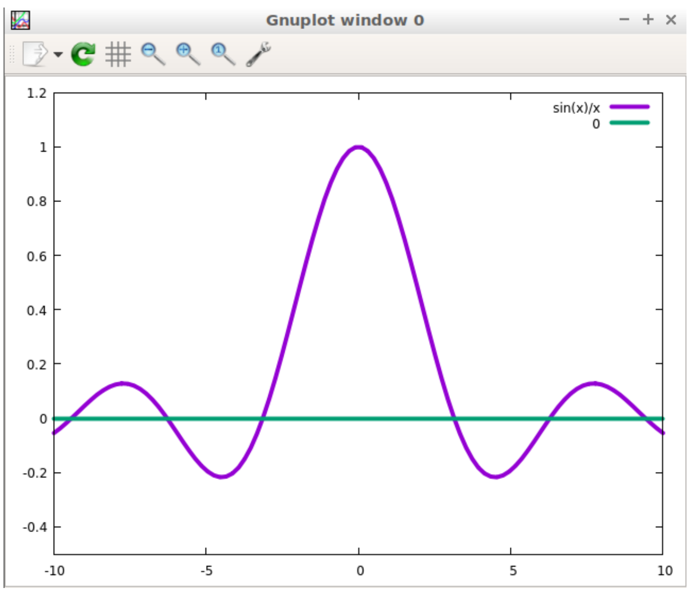


Figura 8: Gráfica de la función ploteada, con raíz en π

VI. CONCLUSIONES

- Los métodos numéricos de bisección y de mínimos cuadrados son muy útiles para solucionar problemas que pueden resultar en extremo complicados de ser resueltos por métodos analíticos. Dan una aproximación muy buena, y tienen mucha utilidad práctica.
- La programación por ordenador tiene grandes aplicaciones para la creación de programas que realicen estos métodos. Siendo posible que realicen hasta cientos de iteraciones en un tiempo muy corto. Algo que resultaría casi imposible de ser intentado manualmente.
- Gnuplot es una poderosa herramienta que permite generar gráficas comparativas tanto de funciones algebráicas como de datos observacionales, siendo muy utilizable en las ciencias fácticas.

[1] **Least Squares Method**
https : //en.wikipedia.org/wiki/Least_squares
Wikipedia.org

[2] **Linear Least Squares Method**
https : //en.wikipedia.org/wiki/Linear_least_squares
Wikipedia.org.

[3] **Ordinary Least Squares Method**
https : //en.wikipedia.org/wiki/Ordinary_least_squares
Wikipedia.org

[4] **Linear Regression Method**
https : //en.wikipedia.org/wiki/Linear_regression
Wikipedia.org

[5] **Bisection Method**
https : //en.wikipedia.org/wiki/Bisection_method
Wikipedia.org

[6] **Intermediate Value Theorem**
https : //en.wikipedia.org/wiki/Intermediate_value_theorem

Wikipedia.org https : //en.wikipedia.org/wiki/Gnuplot Wikipedia.org

[7] **gnuplot**