

UNIVERSIDAD DE SANTIAGO DE CHILE
FACULTAD DE INGENIERÍA
Departamento de Informática



Modelación y Simulación
Laboratorio 1

Diego Mellis

Andrés Muñoz

Profesor: Gonzalo Acuña

Ayudante: Francisco Muñoz

Santiago – Chile

2018

TABLA DE CONTENIDO

Índice de tablas	v
Índice de ilustraciones	vii
1 Introducción	1
2 Marco teórico	3
2.1 MATLAB	3
2.2 Método de Newton-Raphson	3
2.3 Escala Logarítmica	4
3 Desarrollo Primera Parte	5
3.1 Parte a	5
3.2 Parte b	7
4 Desarrollo Segunda Parte	11
4.1 Newton-Raphson	11
4.2 Calculo Matemático	12
5 Conclusiones	13
6 ANEXOS	15
6.1 Manual de usuario	15
6.1.1 Como ejecutar	16
6.1.2 Ejemplos Newton-Raphson	16
6.1.2.1 Ejemplo 1	17
6.1.2.2 Ejemplo 2	17
6.1.3 Ejemplos Calculo Matemático	17
6.1.3.1 Ejemplo 1	18
6.1.3.2 Ejemplo 2	18
Bibliografía	19

ÍNDICE DE TABLAS

ÍNDICE DE ILUSTRACIONES

Figura 3.1	Gráfico de función $a(x)$	5
Figura 3.2	Gráfico de función $b(x)$	6
Figura 3.3	Gráfico de función $a(x)$ y $b(x)$	7
Figura 3.4	Gráfico de función $c(x)$	8
Figura 3.5	Gráfico de función $c(x)$ en escala logarítmica.	8
Figura 4.1	Algoritmo método de Newton-Raphson.	11
Figura 4.2	Algoritmo función fx	12
Figura 6.1	Newton-Raphson ejemplo 1.	17
Figura 6.2	Newton-Raphson ejemplo 2.	17
Figura 6.3	Calculo Matemático ejemplo 1.	18
Figura 6.4	Calculo Matemático ejemplo 2.	18

CAPÍTULO 1. INTRODUCCIÓN

A lo largo de la asignatura de Modelación y Simulación se buscará siempre modelar sistemas para tener una representación más simple de la realidad. Es por esto que como objetivo general de esta experiencia es adentrarse al uso de la herramienta que se utilizará a lo largo del semestre, que corresponde a *MATLAB*, un lenguaje específico de programación que entrega una amplia gama de funciones que serán necesarias para cada experiencia de esta asignatura.

Dentro de los objetivos específicos se tienen:

- La capacidad de graficar funciones con distintas propiedades, ya sea color, forma y escalas distintas dependiendo del gráfico.
- Implementar el método de Newton-Raphson de manera recursiva, recibiendo como entrada un polinomio, un error, un valor inicial y el número de iteraciones.

El presente informe presenta un marco teórico, en donde se definen conceptos relevantes para el entendimiento de esta experiencia, luego un desarrollo que está separado en dos partes, donde la primera consiste en el trabajo con gráficos y dos funciones definidas y la segunda parte con la implementación de métodos numéricos, en este caso, Newton-Raphson. Finalmente, se presenta una conclusión, en base a los resultados obtenidos.

CAPÍTULO 2. MARCO TEÓRICO

2.1 MATLAB

MATLAB es un lenguaje de programación diseñado específicamente para ingenieros y científicos. El corazón de MATLAB es el lenguaje MATLAB, el cual es un lenguaje basado en matrices permitiendo la expresión más natural de la matemática computacional (MathWorks (1994)). Gracias a las características matriciales que presenta este lenguaje, realizar cálculos matemáticos de alta complejidad, se vuelve más rápido y su IDE (entorno de desarrollo integrado) facilita el manejo y visualización de los datos, además de ofrecer librerías que optimizan el trabajo y lo convierten en algo intuitivo de utilizar.

2.2 MÉTODO DE NEWTON-RAPHSON

El método de Newton-Raphson es el método más conocido para hallar las raíces de la ecuación $f(x) = 0$, siendo $f(x)$ un polinomio. A menudo, es el más efectivo.

El método de Newton-Raphson consiste en tomar una aproximación inicial \bar{x} , y a continuación obtener una aproximación más refinada mediante una fórmula. Es decir, se trata de acercarse a la raíz p por medio de la fórmula recursiva: (Gorostizaga)

$$p_n = p_{n-1} - \frac{f(p_{n-1})}{f'(p_{n-1})} \quad (2.1)$$

Es importante mencionar que este método tiene dificultades cuando la función tiene pendientes grandes o múltiples puntos de inflexión cerca de la raíz, lo que trae como consecuencia un aumento de probabilidad de que no exista convergencia (Plaza (2007)).

2.3 ESCALA LOGARÍTMICA

Una escala logarítmica es una escala de medida que utiliza el logaritmo de una cantidad física en lugar de la propia cantidad.

La utilidad fundamental de la escala logarítmica consiste en que se pueden representar valores de magnitudes muy diferentes. También son convenientes cuando permiten convertir el gráfico que relaciona dos variables en una recta (Román).

CAPÍTULO 3. DESARROLLO PRIMERA PARTE

3.1 PARTE A

A continuación se presentan los gráficos de las funciones requeridas para esta experiencia.

Para la función $a(x) = 5\log_5(2x + 5)$, se tiene el siguiente gráfico.

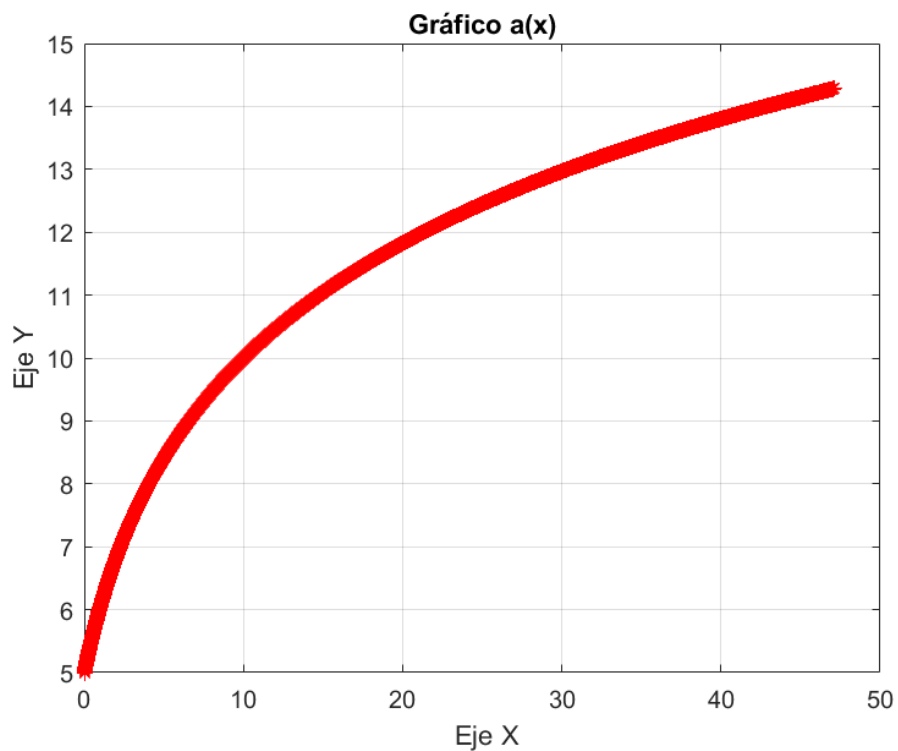


Figura 3.1: Gráfico de función $a(x)$.

Para la función $b(x) = \text{sen}(3(\log_2(x + 3))) + \cos(4(\log_6(2x + 44)))$ se tiene el siguiente gráfico.

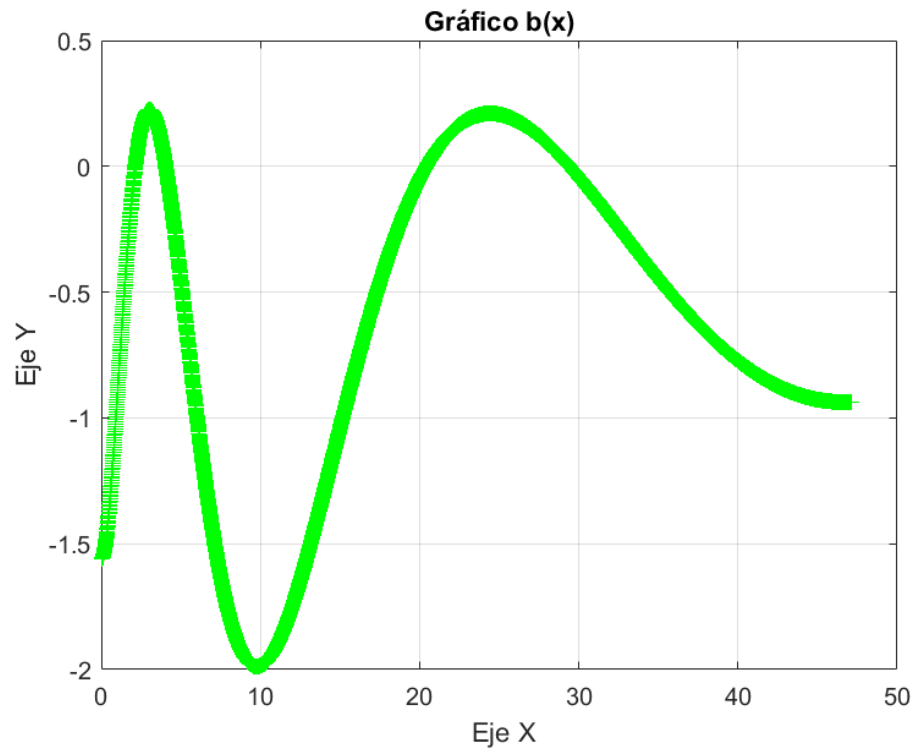


Figura 3.2: Gráfico de función $b(x)$.

Finalmente, se muestran ambas funciones en un mismo gráfico, en escala normal, diferenciadas por colores y con un cuadro informativo para identificar cada una.

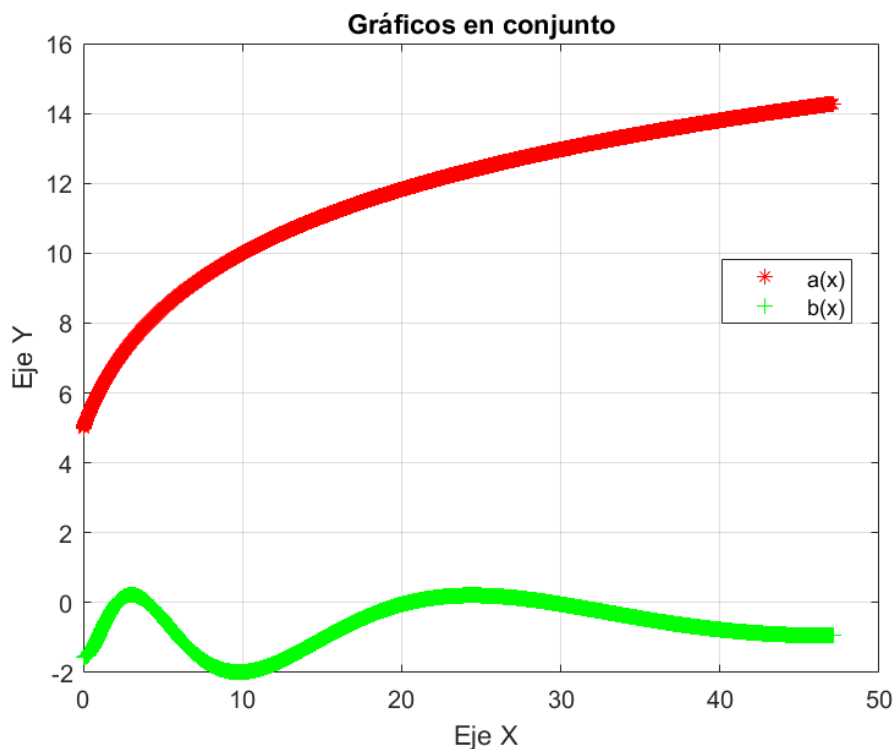


Figura 3.3: Gráfico de función $a(x)$ y $b(x)$.

3.2 PARTE B

Para esta parte se pide utilizar la función:

$$c(x) = 3e^{x+14} \quad (3.1)$$

El intervalo en el cual se pidió evaluar esta función fue de -10 a 10, con una separación de 0.05 entre cada punto a evaluar.

En esta parte, a diferencia de la anterior, la función debe ser mostrada en dos escalas diferentes, siendo estas, la normal y en escala logarítmica, respectivamente. En las figuras que se muestran a continuación pueden apreciarse ambas escalas.

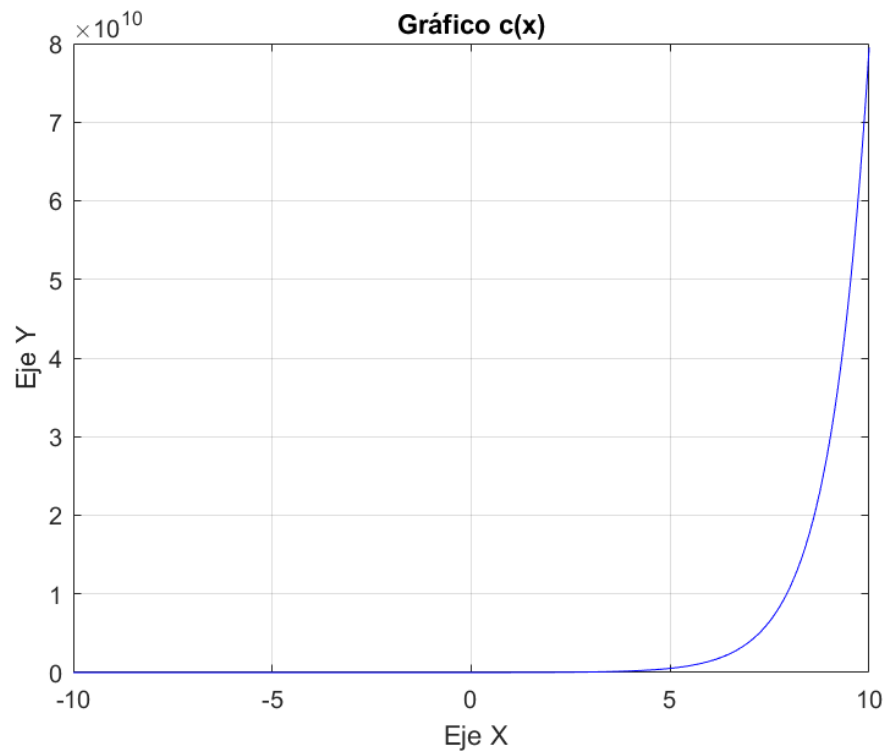


Figura 3.4: Gráfico de función $c(x)$.

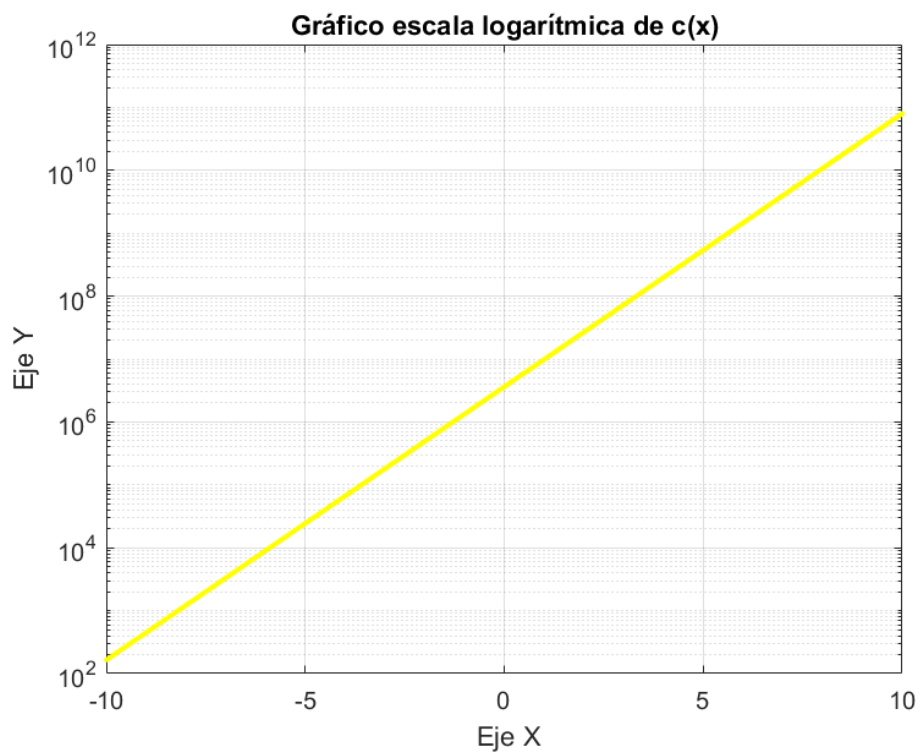


Figura 3.5: Gráfico de función $c(x)$ en escala logarítmica.

Si bien es cierto, ambos gráficos representan la misma función, la forma es diferente. En el gráfico de la figura 3.4, se representa la función en su escala normal, permitiendo ver el verdadero comportamiento de la función que, en este caso, es exponencial, que presenta la mayoría de sus valores cercanos a cero y se genera un fuerte incremento a partir de los valores superiores a 5. Pero, al tener este único gráfico, a simple vista, no se puede obtener mucha información, por que como se menciona anteriormente, todos los valores menores a 5 se acercan a cero, por lo que no hay claridad, además de presentarse una explosión en los valores superiores a 5, sin poder notar de manera detallada el por qué de esta "explosión" en los valores. Es aquí donde entra en juego la escala logarítmica, ya que al ser una función exponencial, permite estudiar la función en forma lineal, llevando así a otros estudios, que pueden ser estadísticos o con otros objetivos, ya que normaliza la función y llevar la función o a una lineal, simplifica en parte su estudio y observación.

CAPÍTULO 4. DESARROLLO SEGUNDA PARTE

4.1 NEWTON-RAPHSON

A continuación, se presenta el algoritmo de Newton-Raphson implementado en MATLAB. La característica principal de este algoritmo es que se ha desarrollado de manera recursiva.

```
1  function [raiz] = newton_raphson(poly, iter, tol, xi)
2  -      xrOld = xi;
3  -      xr = xrOld - (polyval(poly,xrOld)/polyval(polyder(poly),xrOld));
4  -      errorAbs = abs(xr-xrOld);
5  -      if(errorAbs < tol)
6  -          raiz = xr;
7  -      elseif (iter == 0)
8  -          raiz = xr;
9  -      else
10 -          raiz = newton_raphson(poly, iter-1, tol, xr);
11 -      end
12 -  end
```

Figura 4.1: Algoritmo método de Newton-Raphson.

Como se puede apreciar en el código, la función encargada de realizar el método se llama 'newton_raphson.m'. Esta función recibe cuatro parámetros, entre ellos esta el polinomio, cantidad de iteraciones máximas (llamados recursivos), la tolerancia al error y el punto inicial.

La función primero calcula el siguiente punto, respecto al punto anterior. Cabe mencionar que el siguiente punto en el primer llamado a la función, es el punto inicial. Posteriormente, en cada llamado recursivo, el punto inicial pasa a ser el valor siguiente. Este algoritmo, realiza dos verificaciones antes de realizar el llamado recursivo. Primero se verifica que el error sea menor a la tolerancia al error, luego verifica si la cantidad de llamados recursivos es igual a cero, ya que estos dos casos son los casos de borde, en los cuales la función retorna simplemente el valor calculado. En caso de no cumplir las situaciones anteriores, se realiza el llamado recursivo como se mencionó anteriormente.

4.2 CALCULO MATEMÁTICO

En esta sección se desarrolló un calculo matemático pedido en el enunciado de la experiencia. El código implementado se muestra a continuación.

```
1 function y = fx(x)
2     if length(x) < 8
3         disp(' El largo del vector debe ser mayor o igual que 8');
4         y = NaN;
5         return;
6     end
7     a = sum(maxk(x,4));
8     b = sum(mink(x,4));
9     if (a < 0 )
10        disp(' La suma de los valores mas grandes debe ser mayor que 0');
11        y = NaN;
12        return;
13    end
14    if (b < 0 )
15        disp(' La suma de los valores mas pequeños debe ser mayor que 0');
16        y = NaN;
17        return;
18    end
19    y = sqrt(a) - sqrt(b);
20 end
```

Figura 4.2: Algoritmo función fx.

La función implementada recibe el nombre de 'fx' la cual se encuentra en el archivo 'fx.m'. Esta función recibe un solo parámetro, el cual consta de un vector que contiene números. Para el calculo del resultado se saca la resta entre las raíces cuadradas de la suma de los mayores y la suma de los menores números del vector.

Para obtener los mayores y menores números del arreglo se implementaron dos funciones que retornan los k números mayores o menores. La forma de entregar estos números es ordenarlos de menor a mayor y obtener los 'k' últimos o los 'k' primeros valores respectivamente.

Es importante mencionar que la función indica cuando no puede realizar un calculo, es decir, cuando el vector tiene un largo menor que 4 o cuando la suma de los mayores o menores números del vector es negativa, ya que no se puede calcular la raíz cuando recibe un valor negativo.

CAPÍTULO 5. CONCLUSIONES

Durante esta experiencia fue posible sacar a flote nuevamente conocimientos previos que se tenían acerca de *MATLAB*, además de entender lo básico para el uso de este lenguaje y recordar funciones internas que presenta con ayuda de su IDE. Además de a poco se entiende el objetivo que tendrá en la asignatura. Por otro lado, se refuerzan conceptos que se han visto en el transcurso de la carrera de Ingeniería Civil en Informática, tales como escala logarítmica que permitió obtener otra visualización de los datos. También se presenta el método de Newton-Raphson que por detrás conlleva el uso de la recursividad y que permitió encontrar raíces de funciones usando las herramientas de *MATLAB* que permitieron el cálculo con todo lo que conlleva detrás.

Los objetivos de la experiencia, tanto el general como los específicos fueron cubiertos a cabalidad, ya que cada una de las actividades fueron desarrolladas de manera completa y con éxito en sus resultados. Esto queda demostrado con el programa que se hizo de forma modular y ordenada para el entendimiento del lector.

Se espera que con esta experiencia se de inicio a nuevos conocimientos enfocados en la modelación y simulación de sistemas y que se pueda obtener una retroalimentación de tal manera de evitar los errores cometidos en esta experiencia y mejorar también en el uso de *MATLAB*.

CAPÍTULO 6. ANEXOS

6.1 MANUAL DE USUARIO

Dentro de la carpeta existen diferentes archivos que sirven para la ejecución de las funciones realizadas. La estructura propuesta por el equipo de trabajo es la siguiente:

- Carpeta Principal
 - Primera Parte - main.m
 - Segunda Parte
 - Parte Newton Raphson - main.m
 - Parte Calculo Matemático - main.m

Cada uno de los archivos "main.m" contenidos en las carpetas señaladas (Primera Parte, Parte Newton Raphson, Parte Calculo Matemático), contiene las instrucciones necesarias para ejecutar las funciones programadas.

El archivo "main.m" contenido en la carpeta "Primera Parte", ejecuta las funciones desarrolladas para mostrar los siguientes gráficos:

- Gráfico de la función $a(x)$.
- Gráfico de la función $b(x)$.
- Gráfico de la función $a(x)$ en conjunto a $b(x)$.
- Gráfico de la función $c(x)$.
- Gráfico de la función $c(x)$ en escala logarítmica.

El archivo 'main.m' contenido en la carpeta 'Parte Newton Raphson', ejecuta la función de Newton-Raphson para tres ejemplos, y entrega los resultados obtenidos por pantalla.

Finalmente el archivo 'main.m' contenido en la carpeta 'Parte Calculo Matemático', ejecuta la función que realiza el calculo pedido, el cual consiste en una resta de dos raíces.

6.1.1 Como ejecutar

Para poder ejecutar el programa, es necesario tener instalado MATLAB. Con esta herramienta, se puede proceder a abrir el archivo `main.m` que desees descritos anteriormente, el cual es el archivo que se debe ejecutar para que el programa funcione. No olvidar que para que el programa funcione de forma correcta, es necesario que no se altere la distribución de los archivos en su respectivo directorio. Una vez se pulse el botón play en MATLAB en el código `main.m`, MATLAB procederá a preguntar si desea cambiar o agregar la carpeta a la que éste apunta, a lo cual se debe presionar 'Change Folder' para que finalmente se ejecute el programa.

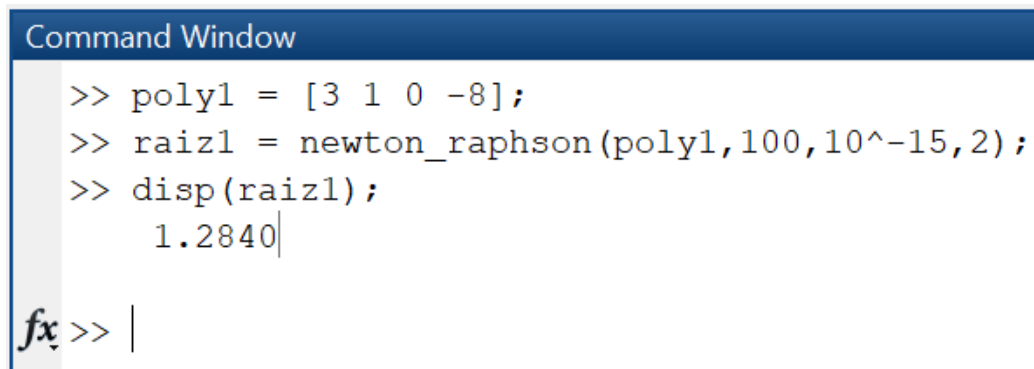
6.1.2 Ejemplos Newton-Raphson

La función implementada para el método de Newton-Raphson, recibe los siguiente parámetros.

- **Polinomio:** Arreglo de números que representan los coeficientes dentro del polinomio.
- **Iteraciones máximas:** Número que indica las iteraciones máximas para resolver el método.
- **Tolerancia al error:** Número que indica tolerancia al error para terminar con la ejecución de la función.
- **Punto Inicial:** Número que representa el valor inicial desde donde se comenzará a desarrollar el método.

Cabe recalcar que la salida que retorna esta función, es la raíz del polinomio que se entrega como parámetro. También, es importante mencionar que para ejecutar los siguientes comandos mostrados en los ejemplos, se debe estar ubicado en la carpeta de 'Parte Newton-Raphson'.

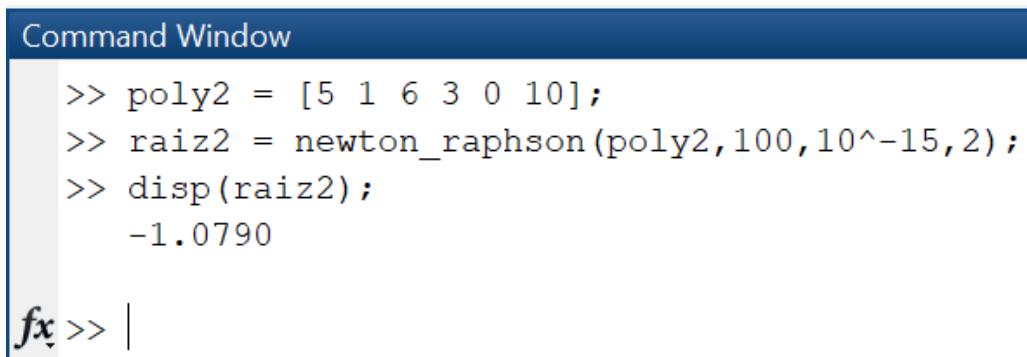
6.1.2.1 Ejemplo 1



```
Command Window
>> poly1 = [3 1 0 -8];
>> raiz1 = newton_raphson(poly1,100,10^-15,2);
>> disp(raiz1);
    1.2840
fx>> |
```

Figura 6.1: Newton-Raphson ejemplo 1.

6.1.2.2 Ejemplo 2



```
Command Window
>> poly2 = [5 1 6 3 0 10];
>> raiz2 = newton_raphson(poly2,100,10^-15,2);
>> disp(raiz2);
   -1.0790
fx>> |
```

Figura 6.2: Newton-Raphson ejemplo 2.

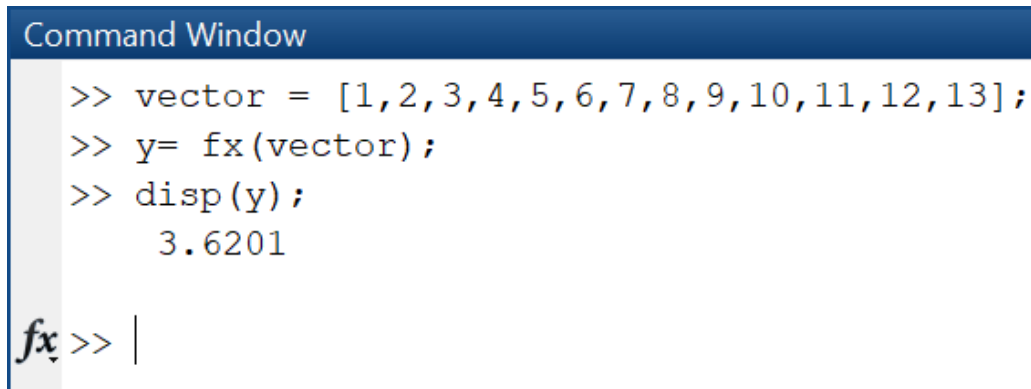
6.1.3 Ejemplos Calculo Matemático

La función para realizar el calculo matemático descrito en el enunciado, recibe el nombre de 'fx.m'.

- **Entrada:** Vector con números.
- **Salida:** Calculo matemático descrito en los capítulos anteriores, el cual consta en la resta de dos raíces cuadradas.

Es importante tener en consideración las mismas advertencias que se hicieron para ejecutar los ejemplos del caso anterior. Es decir, en esta ocasión estar ubicado en la carpeta 'Parte Calculo Matematico'.

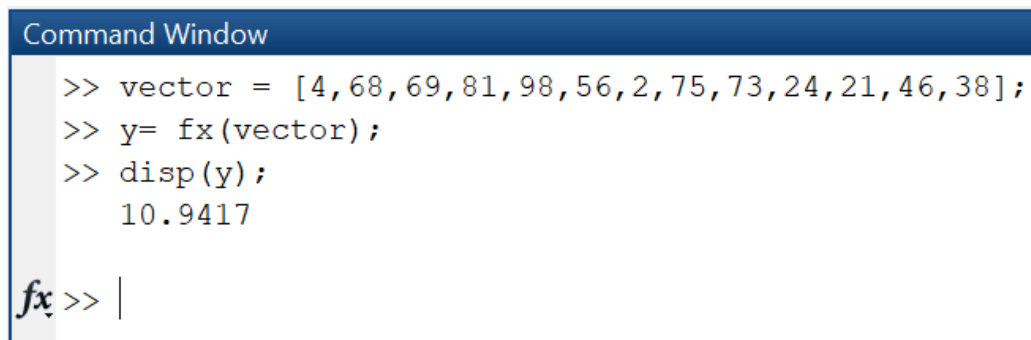
6.1.3.1 Ejemplo 1



```
Command Window
>> vector = [1,2,3,4,5,6,7,8,9,10,11,12,13];
>> y= fx(vector);
>> disp(y);
    3.6201
fx>> |
```

Figura 6.3: Calculo Matemático ejemplo 1.

6.1.3.2 Ejemplo 2



```
Command Window
>> vector = [4,68,69,81,98,56,2,75,73,24,21,46,38];
>> y= fx(vector);
>> disp(y);
   10.9417
fx>> |
```

Figura 6.4: Calculo Matemático ejemplo 2.

BIBLIOGRAFÍA

Gorostizaga, J. C. (????). El método de newton-raphson. Recuperado desde http://www.ehu.eus/juancarlos.gorostizaga/mn11b/temas/newton_ecuac.pdf.

MathWorks (1994). What is matlab. Recuperado desde <https://la.mathworks.com/discovery/what-is-matlab.html%22.html>.

Plaza, S. (2007). Métodos numéricos.

Román, F. J. S. S. (????). ¿qué es una escala logarítmica? Recuperado desde http://hidrologia.usal.es/Complementos/papeles_log/fundamento_log.pdf.