Manual
del Programador
Proyecto Metodos Numericos

Presentado por:
María Alejandra Pabón 1310263
Mayerly Suarez 1310284

Analisis y Metodos Numericos
Luis Fernando Gomez

Manual del Programador
Proyecto Final Metodos Numericos

1. Introducción

El siguiente informe explica cómo se implementó el software de aproximaciones usando los métodos de Polinomios de Lagrange y Diferencias Divididas.

El proyecto sobre aproximaciones fue realizado en Qt Creator en el lenguaje C++ y en el sistema operativo Linux Ubuntu 14. Se usaron algunas funciones de la librería Qt para la implementación en general. Y también se usaron otras librerías para la compilación y graficación de funciones, las cuales más adelante se explicará cómo se usaron.

2. Condiciones del software

Generalizando, el software de aproximaciones realiza lo siguiente:

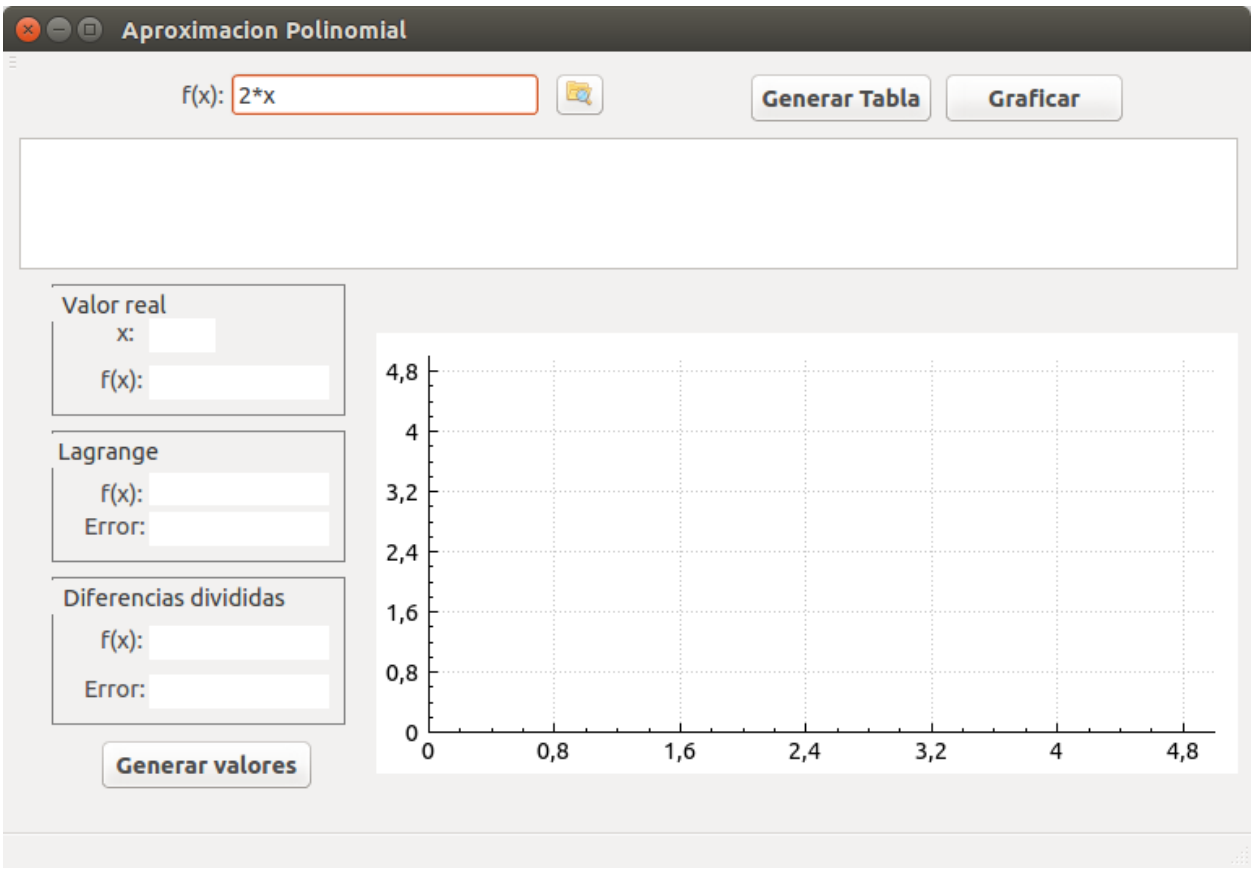
1. Se compila la función ingresada
2. Toma la cantidad de X seleccionados por el usuario
3. Toma los valores de X de un archivo .txt o .csv importado
4. Se genera la tabla con los valores de X y de Y tabulados, hasta el número de puntos seleccionados
5. Toma un x ingresado
6. Evalúa el valor de x en la función
7. Calcular el polinomio de Lagrange.
8. Evalúa el valor de x en el polinomio de lagrange,
9. Calcula el polinomio de diferencias divididas
10. Evaluar el valor de x en el polinomio de diferencias divididas
11. Muestra el error entre el valor real y el valor con polinomio de lagrange
12. Muestra el error entre el valor real y el valor con diferencias divididas
13. Muestra la gráficas de las tres funciones: función normal, función con polinomio de lagrange, función con diferencias divididas

3. Configuraciones iniciales para abrir el proyecto

royecto se da clic derecho sobre el nombre del proyecto y luego en “build”. Si al compilar hay un error donde diga “..target”, se debe dar clic derecho al proyecto y luego en “run qmake”.

4. Ejecución del software

Primero se escribe la función matemática. Más adelante en el punto 4, se explican las forma en que se deberían ingresar todas las funciones matemáticas, aceptadas por el compilador de funciones matemáticas utilizado, llamado “C++ Mathematical Expression Toolkit Library (ExprTk)” :



5. Clases implementadas

La declaración de variables o de funciones se realizó en los archivos con extensión .h,. Las clases implementadas fueron main.cpp: encargada de ejecutar el programa y mainwindow.cpp: contiene todas las funciones implementadas.

6. **Importar archivo con valores de X**

Se asignó la opción de importar un archivo .txt o .csv para cargar los valores de x con los cuales se va a trabajar, esto para no limitar al usuario en el tamaño de n valores de x, puesto que con un archivo importado puede importar tanta cantidad de valores de x como desee. La función encargada de importar el

archivo y cargar los valores de x en una tabla se llama *importararchivo()*, la cual hizo uso de la clase ifstream y carga el archivo haciendo uso del botón trivial “examinar”.

7. Compilación de funciones

Para la compilacion de funciones matematicas se utilizo la libreria “C++ Mathematical Expression Toolkit Library (ExprTk)” descargada de la pagina http://partow.net/programming/exprtk/. El manual sobre como escribir las funciones matemáticas se encuentra en el manual de usuario. La función encargada de compilar la función y cargar los valores de y tabulador en la tabla se llama *compilarfuncion()*.

8. Graficación de funciones

Se utilizó la librería QCustomPlot para graficar las funciones. Esta librería se descargo de la pagina http://www.qcustomplot.com/ y facilito la graficacion de funciones y asi mismo agregarle ciertos detalles a la grafica, por ejemplo con la funcion SetInteractions se pudo configurar la grafica para el usuario pueda manipularla y mover el eje x, el eje y, o realizar zoom sobre ella. La función encargada de graficar se llama *graficar()* y adicionalmente se tuvieron que importar los archivos *qcustomplot.h* y *qcustomplot.cpp*.

9. Cálculo del método polinomio de Lagrange

Para calcular la evaluación de un valor x ingresado en el polinomio de lagrange se creo la funcion *generarLagrange(double x)*, la cual calcula el polinomio con el procedimiento respectivo del método, recibe el valor a evaluar y devuelve el valor evaluado en el polinomio.

Tambien se hizo la funcion *generrarErrorLagrange(doublex, double y*), la cual recibe el valor x evaluado en la función normal y el valor de x evaluado en el polinomio de lagrange. La función calcular el error entre estos dos valores y devuelve el resultado del error.

10. Cálculo del método diferencias divididas (Interpolación de Newton)

Para calcular la evaluación de un valor x ingresado en el polinomio de diferencias divididas se creo la funcion *generarDiferenciasDivididas(double x)*, la cual calcula el polinomio con el procedimiento respectivo del método, recibe el valor a evaluar y devuelve el valor evaluado en el polinomio.

También se hizo la funcion *generrarErrorDiferenciasDivididas(double x, double y*), la cual recibe el valor x evaluado en la función normal y el valor de x evaluado en el polinomio de diferencias divididas. La función calcular el error entre estos dos valores y devuelve el resultado del error.

11. Otras funciones implementadas

Otra funcion implementada fue la del evento “on_lineEdit_textChanged(const QString &arg1)” y “on_lineEdit_textChanged4(const QString &arg1)” , es decir, algo sucede cuando se edite el texto donde se ingresa la función matemática o donde se ingrese el valor a evaluar. Ese algo sucede quiere decir que cuando se editar ese texto, el software entiende que se va a calcular todo con otra función matemática y entonces limpia la ventana.