KTH Computer Science
and Communication

SimpleGraphPlotter v1.6

Programkonstruktion för F, DD1342
Laboration 4A

JIM HOLMSTRÖM
JIMHO@KTH.SE

Teacher: Ann Bengtson

Contents

Chapter 1

Introduction

In the following part firstly the problem will be explained and secondly the requirements for a basic plotter will be enlisted. A plotter is a program that can plot functions from strings which defines the functions by ordinary math syntax. This project uses `C++` programming language and the `gtkmm`[1] wrapper for the `GTK+`[2] toolkit to generate the graphical user interface. It is compiled with the `GNU gcc`compiler.

1.1 Requirements

A few basic things is needed to have a functioning math plotter:

1. Define a function given ordinary math syntax.
2. Parse the inputed function and plot it accordingly.
3. Add/Remove functions from plotarea.
4. Plotarea should be scrollable both vertical and horizontal.
5. Range should be fixed to the unit-cube. [3]
6. Display axis of the plot.
7. Parser must be properly tested.

1.2 Scope

The amount of functionality that is possible to put in a system like this is almost endless so a few delimitations has to be made in order to complete the project. The

[1] Documentation, binaries and source can be found at: www.gtkmm.org
[2] Documentation, binaries and source can be found at: www.gtk.org
[3] This restriction will be handled in section 1.2

currently biggest restriction to the plotter is the lack of ability to zoom or change the range from the unit-cube. No support for parametric nor complex functions. [4]

1.3 Assistance

Besides the reference manuals for `gtkmm` no external help for this project was received.

[4] Since no native support in `C++` for complex numbers which means all the basic math functions would have to be rewritten in order for this to work.

Chapter 2

Structure

An basic overview of the structure can be seen in figure 2.1, all parts will then be enlisted and explained in a `javadoc` like manner.

2.1 Parser

The parser code can be divided into to parts the algorithm code, that is the actual parser, and the data structure in the form of a parse tree.

2.1.1 parser

public parse(expr : std::string) Test explain test test test

Parameters:
`expr` - test test

Returns:
`iexpression*` - test test

Public <basic description of the function> TODO should i perhaps move the arguments/return as i doxygen to their own posts?

parse_exception

operators

2.1.2 unary_level

what is this used for?

2.1.3 iexpression...

2.2 Plotter

... <images with the different parts highlighted with a red border, that is the parts being described at the moment> especially point out the inheritance in the custom widgets.

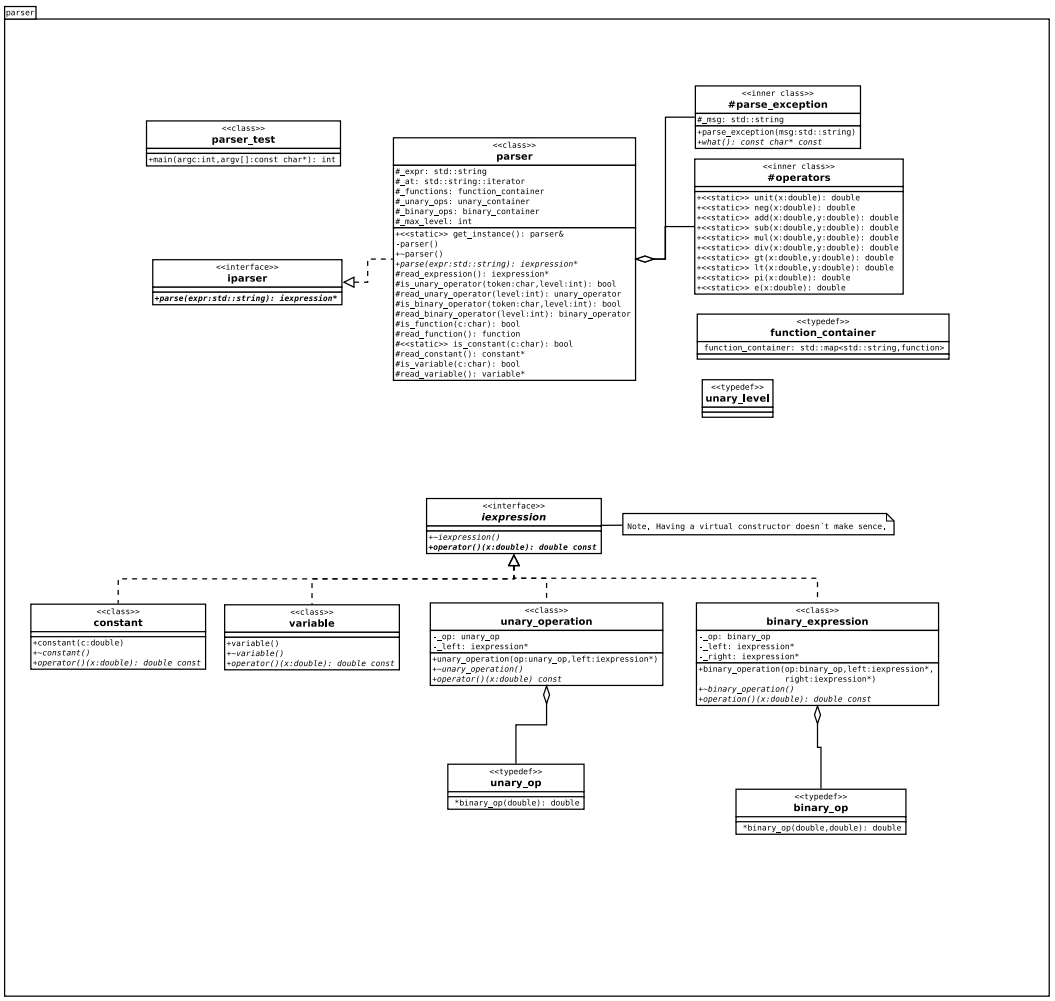

Figure 2.1. An UML showing the structure and the enclosure.

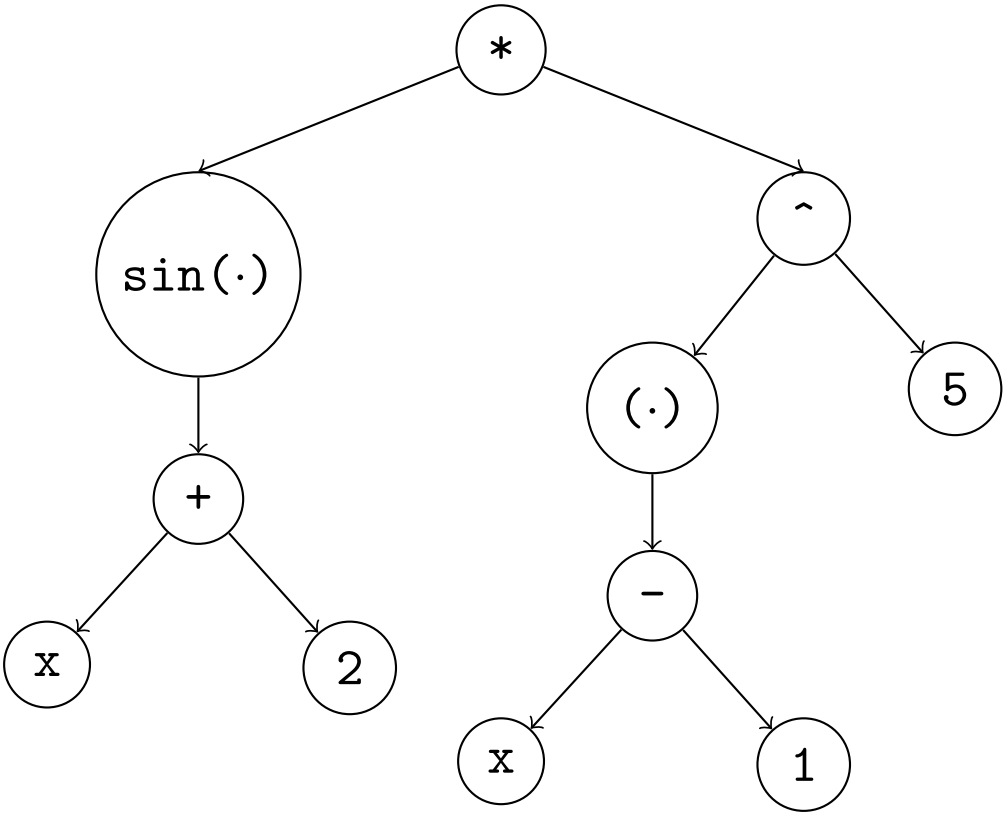

Figure 2.2. An example of the parse tree for the expression `sin(x+2)*(x-1)^5`. Trivial nodes where left out.

Chapter 3

Results and Discussion

3.1 Results

«screenshots» Runned trough valgrind, results?.

3.2 Discussion

Problems with the unofficial `C++`wrapper `gtkmm`, only used it to avoid missing out inheritance, polymorphism and to get it compatible with the standard `C++`Library.