Root finder for quadratic functions

Lars Rikard Rådstoga

2022-01-16

1 Introduction

This report will describe a simple LabView program that finds roots for quadratic functions. The goal of the project is to practice programming and writing reports with the preferred structure.

1.1 Problem Description

Many applications of automation and simulation involve second order differential equations which need to be solved numerically. Newton's method is often used as a part of the solution. Create a LabView VI that can be used as a SubVI in future projects that finds roots of quadratic functions.

2 Methods and Results

Newton's method is used for this program, see eq. *(2.1)* for the discrete formula.

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{2.1}$$

2.1 Code structure

Functionality of the program is divided into SubVIs to make the program more readable and reusable. The file structure is shown in Figure 2-1. The main VI UserProgramNewtonSolver is meant for user input and its dependencies are only meant to be used as SubVIs. It is therefore easy to reuse these functions in future projects.

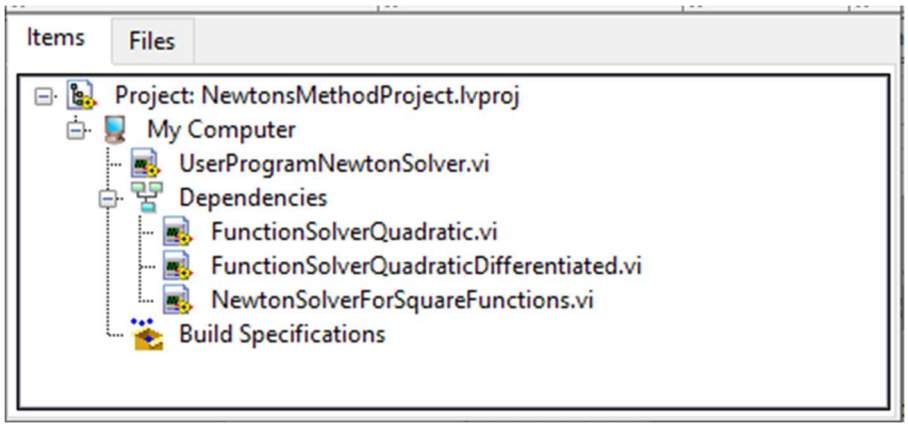

Figure 2-1: File structure.

2.2 UserProgramNewtonSolver.vi

This program is meant for user input and is a continuously looped program which calculates and shows the local root of a quadratic function.

2.2.1 Front panel

Figure 2-2 takes user input and calculates the root for the function with the given coefficients and bounds.

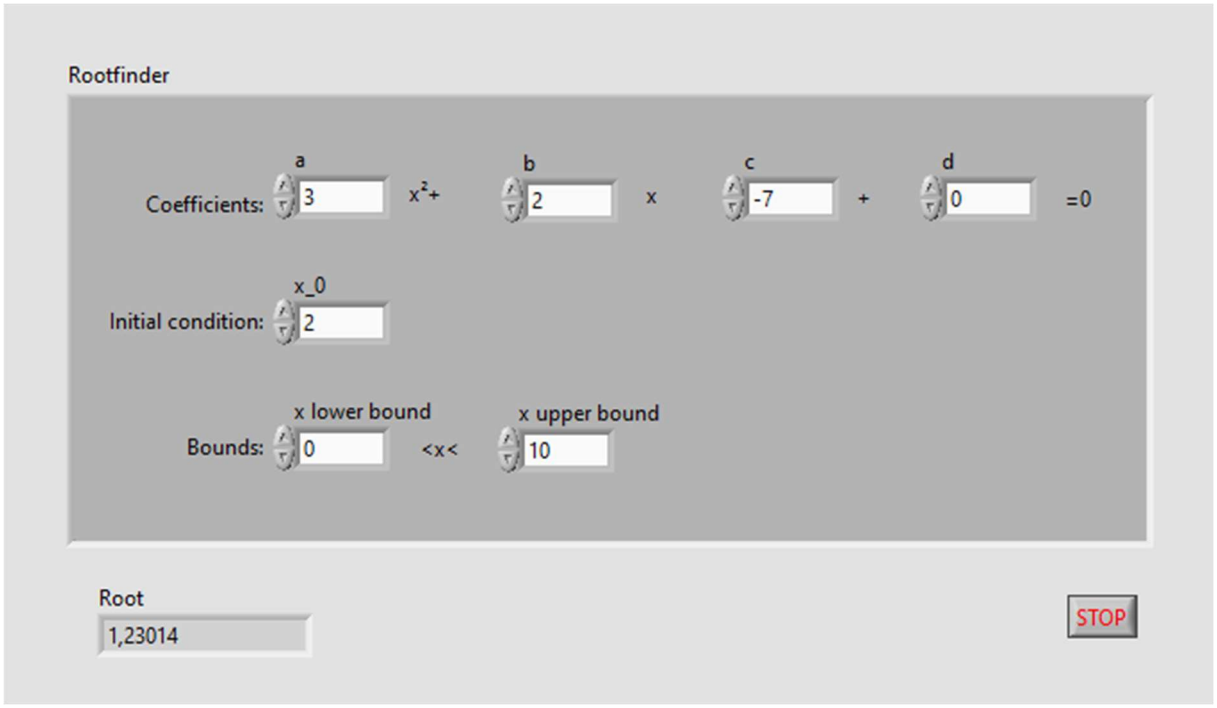

Figure 2-2: Front panel meant for user input.

2.2.2 Block diagram

The block diagram, see Figure 2-3, unbundles the cluster and uses the NewtonSolverForSquareFunctions subVI in a continuous loop.

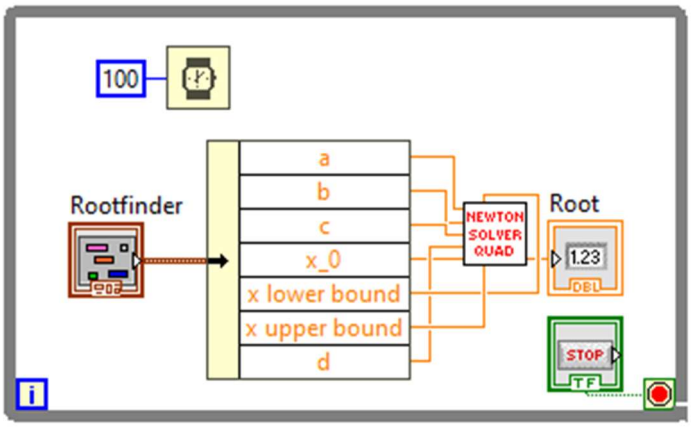

Figure 2-3: Block diagram for the user program.

2.3 NewtonSolverForSquareFunctions.vi

This VI is the main program where Newton's method is implemented.

2.3.1 Block Diagram

Figure 2-4 and Figure 2-5 shows the block diagram for the initial case 0 and the default cases of the for loop in which Newton's method is iterated 20 times. The difference is that the first iteration uses a given initial x value and the other use the iterated x_n value from the shift register.

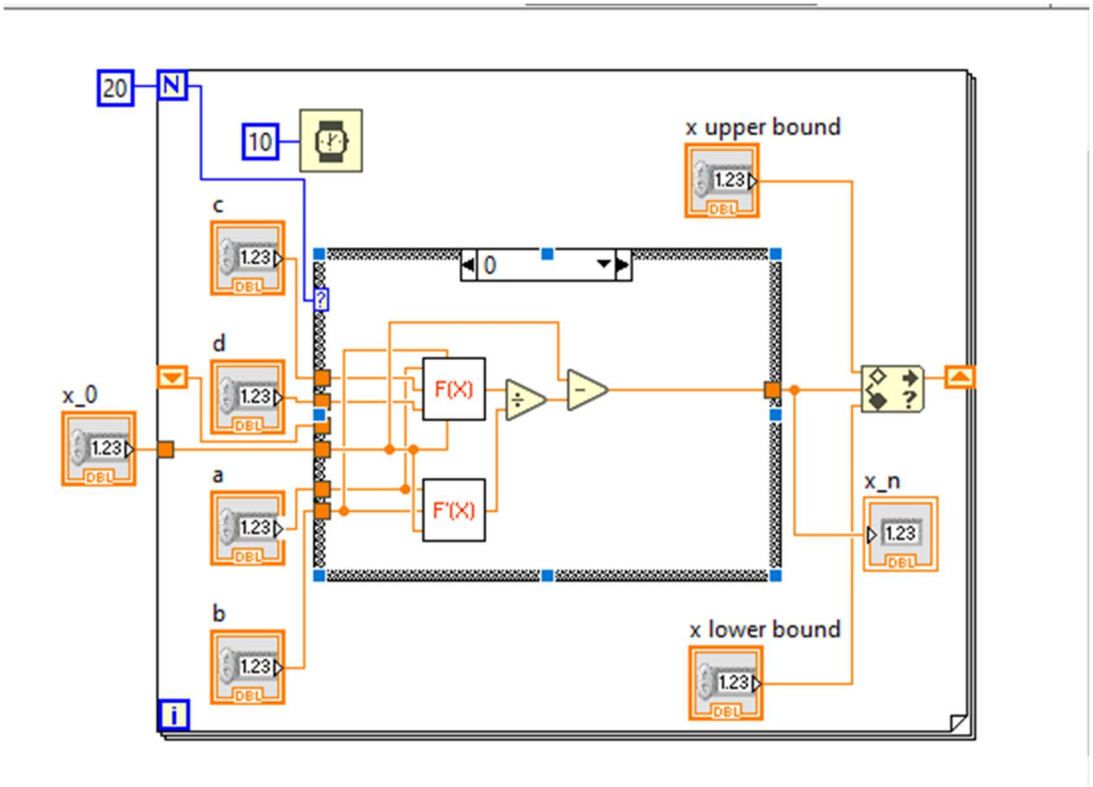

Figure 2-4: Case 0

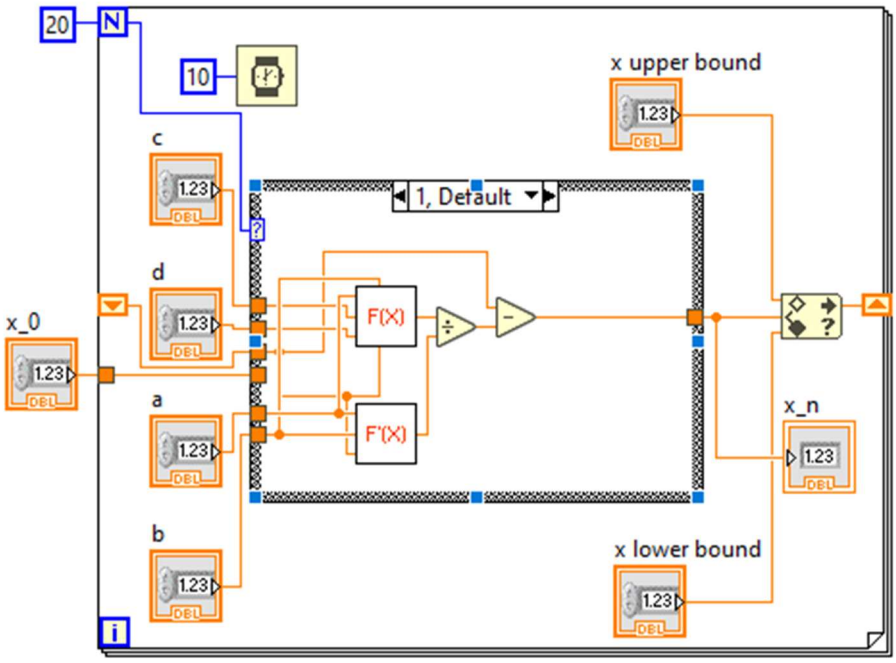

Figure 2-5: Default case

2.4 FunctionSolverQuadratic.vi

Takes coefficient inputs and an x value to evaluate a quadratic function.

2.4.1 Block Diagram

Figure 2-6 shows the block diagram for this VI.

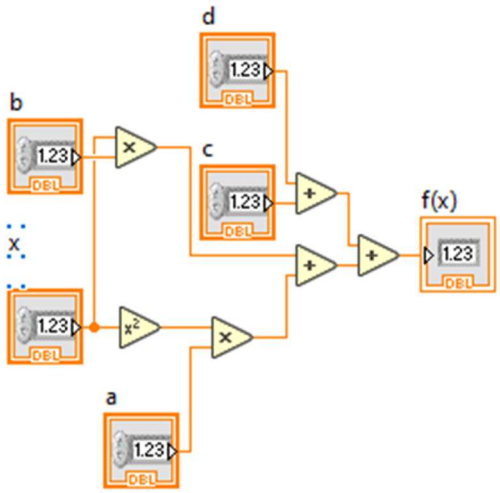

Figure 2-6: Calculation of the quadratic function

2.5 FunctionSolverQuadraticDifferentiated.vi

Takes coefficient inputs and an x value to evaluate a differentiated quadratic function.

2.5.1 Block Diagram

Figure 2-7 shows the block diagram for this VI.

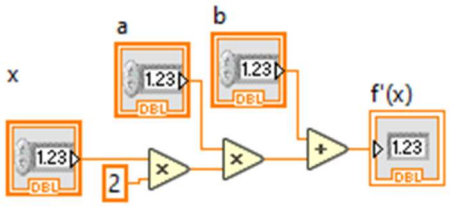

Figure 2-7: Calculation of the differentiated quadratic function

3 Discussion

The program works as expected. Edge cases can still be ironed out, however. E.g., if a local root can not be found then the user won't know that it was not found.