ODE representation - Square root

Julie Thiim Gadeberg
201604903

March 2019

1 Introduction

In this report we seek to create an algorithm that is capable of calculating the square root of any given real positive number x. The chosen programming language for the algorithm is C, and the algorithm is based on functions ([1]) from GNU Scientific Library that can solve the ordinary differential equations. This report firstly gives a short description of the created algorithm in section 2, while a presentation of the results is given in section 3.

2 The algorithm

We use the ODE representation given by:

$$\frac{\mathrm{d}y}{\mathrm{d}x} = \frac{1}{2y}, \tag{1}$$

as a starting point in order to calculate the square root of x. The first part of the algorithm is to define this equation in a separate function that depends on the variables x and y, as well as the value of the equation parameters. The function stores the solution in the defined array $dydx$, while returning the message `GSL_SUCCESS`.

Afterwards, it is possible to set up the function that solves the ODE. The function only takes x as an input, but i order to avoid complex numbers, it is asserted that the given x value is larger than or equal to zero. If this is not the case an error occurs, and the program terminates. The range of the argument is also reduced using the formula:

$$\frac{1}{\sqrt{\frac{1}{x}}} \text{ if } x < 1 \tag{2}$$

$$2\sqrt{\frac{x}{4}} \text{ if } x > 4. \tag{3}$$

Once the range is appropriate, we proceed to define the system of equations by using `gsl_odeiv2_system` as the data type, and combine it with the equation function. The problem is one-dimensional, and no parameters need to be defined, as the ODE does not include any such parameters. Afterwards, a driver combining the steps, control, and evolution of the ODE system is chosen. We choose to use the `gsl_odeiv2_driver`. It takes the chosen data type, step type (`gsl_odeiv2_step_rk8pd`), step size, absolute and relative precision as arguments. The driver is then applied stepping from the initial value of x to the given x argument. The result is stored in the given vector y that is returned from this function, but if an error occurs an error message indicating the error type is returned instead. However, before returning either results, the driver is freed from the memory.

The last part of the algorithm is the main function. This function only includes a for loop that prints the variable x, the ODE solution and the value obtained from the `<math.h>` library.

3 Results

For our purposes we let x range from 0 to 25 with $\Delta x = 0.2$. The given initial condition is $y(1) = 1$, and the result is seen in figure 1. The figure depicts the results of the created algorithm, and the corresponding function from the `<math.h>` library. It is clear that the created algorithm works as intended as both of the functions are in agreement.

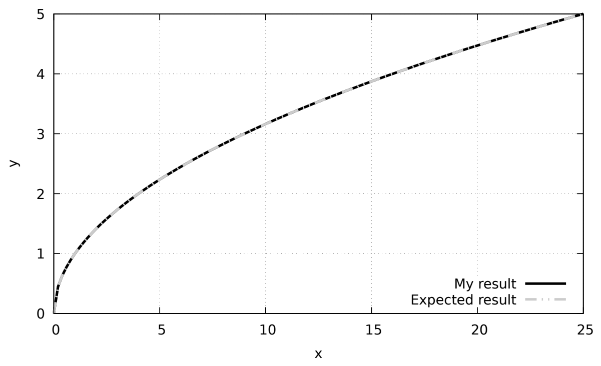

Figure 1: A plot of the results from the created algorithm and the square root function from the `<math.h>` library in black and grey respectively. The abscissa shows the different values of x, while the ordinate shows the corresponding value of y

References

[1] GSL. *Ordinary Differential Equations.* `https://www.gnu.org/software/gsl/doc/html/ode-initval.html`.