Halfway Project: Exercise 9 - Exponential function (ODE representation)

Mikkel Elkjaer Pedersen

This short report outlines how the Ordinary Differential Equation (ODE) representaiton of the exponential function may be acchieved. In short, the object is to calculate $\exp(x)$ for a real number x using the ODE representation of the exponential function given as

$$\frac{dy}{dx} = y,\ y(0) = 1, \tag{1}$$

where the latter expression is the initial condition for the exponential function. Knowing the initial condition and the differential equation one then integrates the differential equation up to the desired real number x to get $\exp(x)$. As with any numerical integration one should aim to implement one or more reductions of the argument to avoid integration over too large an interval. For the exponential function the argument x can be reduced to the interval $[0, 1)$ by exploting the following identities:

$$\exp(-x) = \frac{1}{\exp(x)},\ \exp(x) = \exp\left(\frac{x}{2}\right)^2. \tag{2}$$

Also, for the representation plotted in fig. 1 the initial condition has been explicitly enforced when performing the integration. To perform the integration routines from Gnu Scientific Library (GSL) has been used. To utilize the library for solving ODEs the header **gsl_odeiv2.h** must be included in the program. The program should then set up (that is, define) the ODE system. This consists of at least a function defining the differential equation and a statement of the dimensionality of the problem. Typically the problem depends on one or several parameters which should also be comitted to the ODE-system. Additionally, one may include a function to calculate the Jacobian if this is available for the problem. Once the system is set up the initial conditions of the problem, the initial step size, tolerances and the prefered integrator method can be specified to a driver that then automatically combines evolution, stepper and control objects to perform the integration. In this project the Runge-Kutta-Fehlberg (4,5) method is chosen as the integrator algorithm as it is a good general-purpose integrator. The initial guess for a proper step size is 0.1 while both the relative and absolute tolerances have been set to $1e-5$. In a main function the program described above is then called with x being the argument. The range is chosen

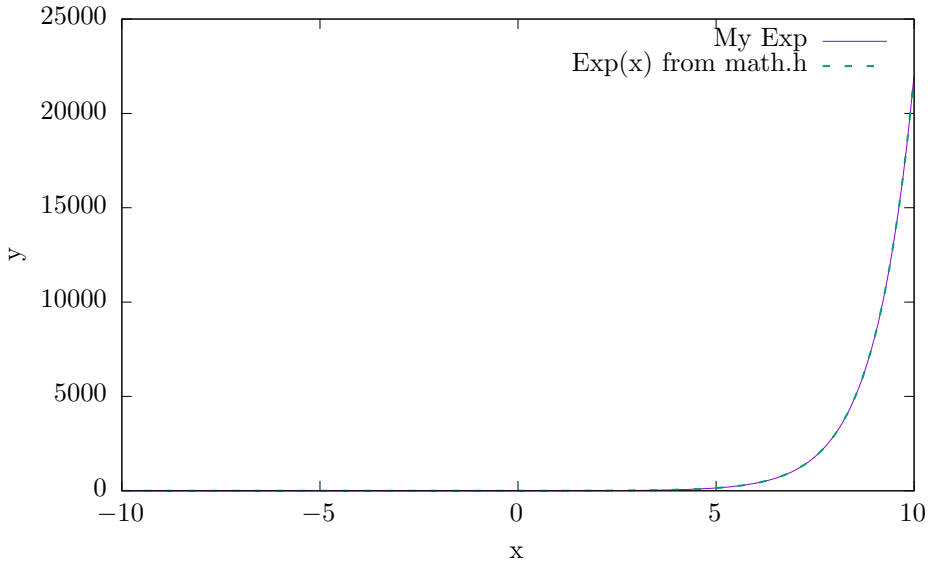

Figure 1: A plot of the ODE representation of the exponential function and the one from the standard C library math.h in the range $[-10, 10]$.

to be $[-10; 10]$ to show that the reductions work. That is, when the program is called with e.g. $x = 7.2$ the program recursively calls itself with $x/ = 2$ until x is in the desired range $[0, 1)$. The integration is then performed and the result is carried through the levels of recursion where eqn. 2 is then properly utilized to get the result for $x = 7.2$. The result is plotted in fig. 1 along the exponential function from math.h in GSL.