Natural logarithm (integral representation)

Alexander S. Madsen *

Institute for Physics og Astronomi, Aarhus Universitet, Denmark

June 30, 2018

1 Introduction

In the following we are asked to implement a function that calculates the natural logarithm of a real positive number x using the integral representation

$$\ln(x) = \int_1^x \frac{1}{t}\, dt \tag{1.0.1}$$

The integration is to be carried out using a suitable GSL integration routine. Furthermore, to improve computational efficiency, the integration interval must be *reduced* before the routine is called such that $1 \leq x < 2$ for an arbitrary positive number x. This is to be done using the formulas

$$\ln(x) = -\ln\left(\frac{1}{x}\right),\ \ln(x) = \ln(2) + \ln\left(\frac{x}{2}\right) \tag{1.0.2}$$

Finally the acquired results must be compared to a with $\ln(x)$ function from the $<$ math.h $>$ library or from GSL.

2 Implementation

A separate *c*-file called *log_int.c* is build containing a numerical implementation of eq. (1.0.1), compliant with the specification of the GSL library h, as well as a function executing the GSL integration routine itself. Specifically the function takes a *double x* corresponding to the upper integration limit in eq. (1.0.1) and returns the *double result*. The function automatically allocates and frees the workspace needed for the particular GSL integration routine which for this problem is set to be gsl_integration_qags. The "projection" of the positive variable x onto the interval $1 \leq x < 2$ is implemented by nesting the function within itself using the following if-statement sequence

```
if(x < 1){return -log_int(1.0/x, calls);}
if(x >= 2){return log(2.0) + log_int(x/2.0, calls);}
```

*e-mail: asm.arbejde@gmail.com

This results in a recursion in which the function continuously calls itself until the condition $1 \leq x < 2$ is realized at which point the GSL integration routine is carried out. The program is run from the c-file main.c by calling log_int in a for loop. Specifically the function is evaluated from 0.1 to 10 using a step size of $\Delta x = 0.1$.

3 Results and Conclusion

Figure shows a plot of the data points contained from the datafile *out.txt* created when the script is run. It is evident from fig. 1 that the routine produces sensible values of $\ln(x)$. Hence we can conclude that

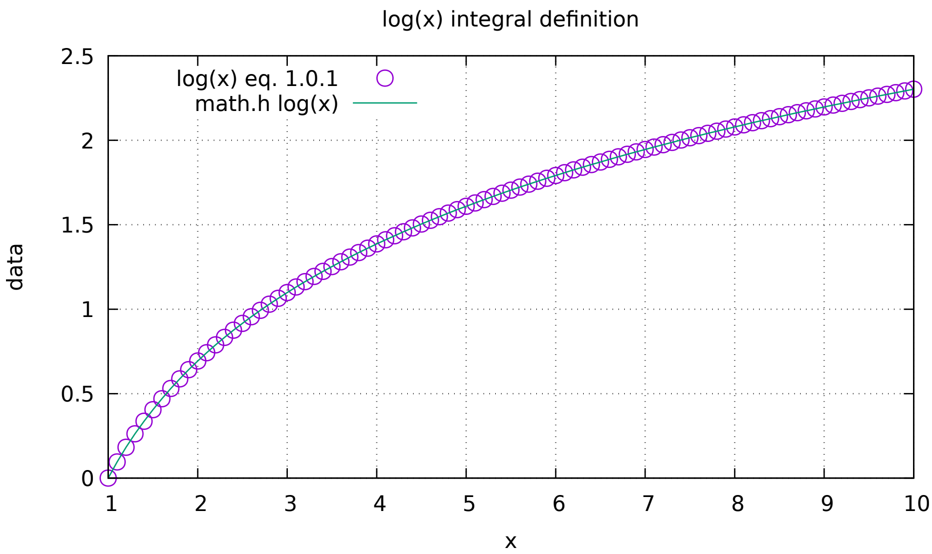

Figure 1: Result from running the main.c script. The figure shows the eq. (1.0.1) *from* $x = [1, 10]$ *and overlayed of the* $\ln(x)$ *from the math.h library*

eq. (1.0.1), with the required reduction of the integration, is meaningful and computationally efficient way to calculate $\ln(x)$