Approximating the exponential function

René Munk Thalund

March 2022

Abstract

We examine a `C#` implementation of the exponential function that uses only multiplications and additions.

1 Introduction

The exponential function e^x is ubiquitous in math and science. It's defined as the unique function that is it's own derivative, and has value 1 for argument 0, that is:

$$\frac{df}{dx} = f(x) \text{ and } f(0) = 1 \tag{1}$$

The definition extends to complex arguments. Notably with an entirely imaginary argument $e^{i\phi}Z$ rotates the complex number Z by ϕ radians in the complex plane.

2 Approximation

We're considering the following `C#`-function.

```
static double ex(double x){
    if(x<0)return 1/ex(-x);
    if(x>1.0/8)return Pow(ex(x/2),2);
    return 1+x*(1+x/2*(1+x/3*(1+x/4*(1+x/5*(1+x/
    6*(1+x/7*(1+x/8*(1+x/9*(1+x/10)))))))));
}
\label{code}
```

For small arguments e^x is approximated using the geometric series up to the ninth term:

$$e^x \approx \sum_{n=0}^{9} \frac{x^n}{n!}\ , \quad x \in [0, 0.125] \tag{2}$$

In the source code the sum us convoluted so the minimum number of multiplications and additions is used. For larger arguments e^x is evaluated recursively, using the algebraic identity:

$$e^x = (e^{x/2})^2, \ \text{ used for } \ x > 0.125 \tag{3}$$

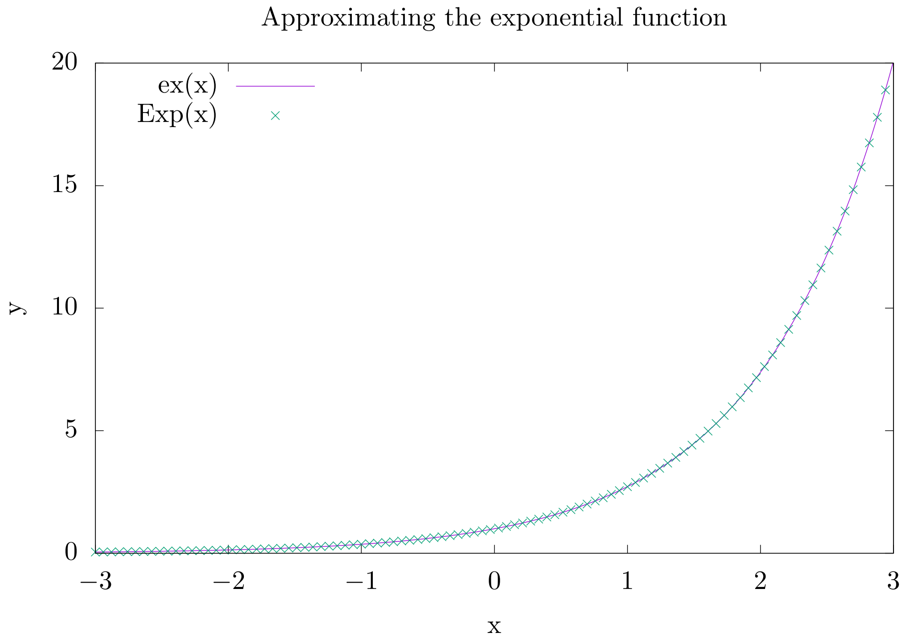

Figure 1: The approximation is very precise

As seen the argument is halfed in each recursion step until the regime defined by eq. 2 is reached. For negative arguments e^x is evaluated as the reciprocal of the correponding positive argument:

$$e^{-x} = \frac{1}{e^x} \quad \text{used for} \quad x < 0 \tag{4}$$

3 Results

For moderate values of x the approximation is very precise. The deviation of $ex(x)$ from $exp(x)$ is so small we have no chance of observing the difference in a direct plot of the values as in fig. 1.

Instead we consider the relative deviation from the system math value of $exp(x)$, see fig. 2. We notice that the deviation is larger for larger absolute values of x. The deviation rises in steps, which is natural since the approximation for each octave interval $]2^n, 2^{n+1}]$ is inherited from the previous down to the 'mould' of the approximation at $]2^{-4}, 2^{-3}]$. Since

$$\frac{ex(2x)}{exp(2x)} = \left(\frac{ex(x)}{exp(x)}\right)^2 \tag{5}$$

we can estimate the deviation at $e^{709} \approx 1 \times 10^{308}$, the largest double allowed in `C#`. The approximation for this value is six 'generations' up from the the $]8, 16]$-interval which we read from fig. 2 to have a relative deviation at about $1 + 1.5 \times 10^{-14}$. Hence, using $(1 + \epsilon)^2 \approx (1 + 2\epsilon)$ the relative deviation at the maximum calculable value will be about $1 + 9.6 \times 10^{-13}$. Not that bad.

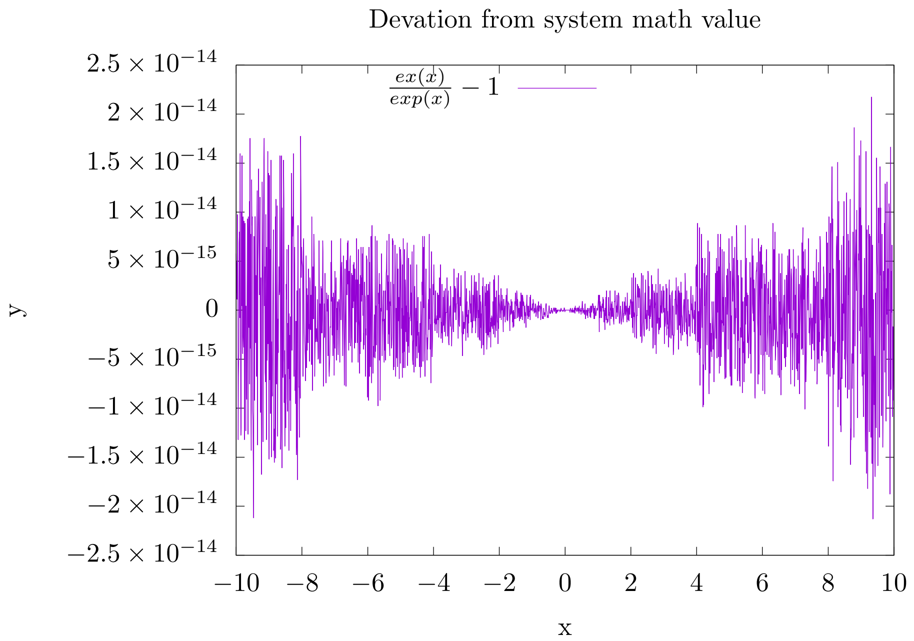

Figure 2: The relative error rises in steps due to the recursive definition of the approximation

4 Conclusion

We have successfully shown that e^x can be estimated using only multiplications and additions.