PYU33C01 Numerical Methods Assignment 1

Completed by Seán Alexander 21364546

Introduction:

This assignment will examine and plot the use of Heron's root finding method, determine the overflow, underflow and machine precision of floating point numbers of the executing computer, and compare the forward and central difference methods as the step size approaches machine precision.

Part A: Heron's Root Finding Method

Heron's root finding method was applied to estimate the square root of a positive real number: $x_{n+1} = \frac{1}{2}(x_n + \frac{a}{x_n})$; Where x_{n+1} is the next iteration after the n_{th} approximation of x where $x^2 = a$. The relative error of each approximation iteration was then plotted to demonstrate the relationship between the relative error and the number of approximations n. This was done for the square root of 2, 3, and 5.

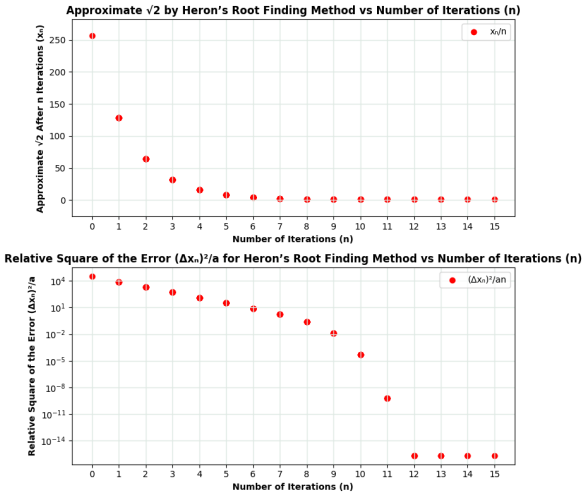

Figure 1: *Heron's Root Finding Method For* $\sqrt{2}$

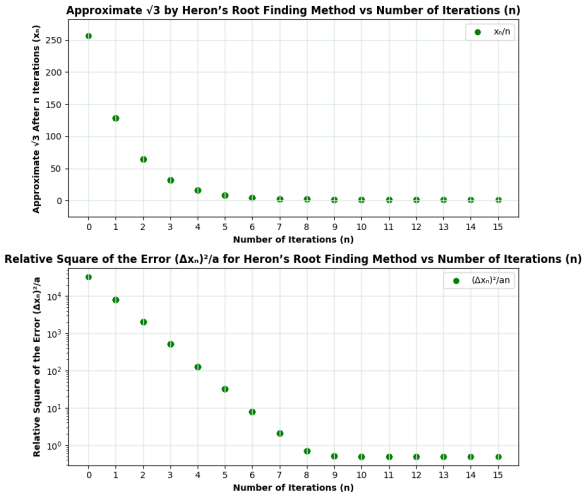

Figure 2: *Heron's Root Finding Method For* $\sqrt{3}$

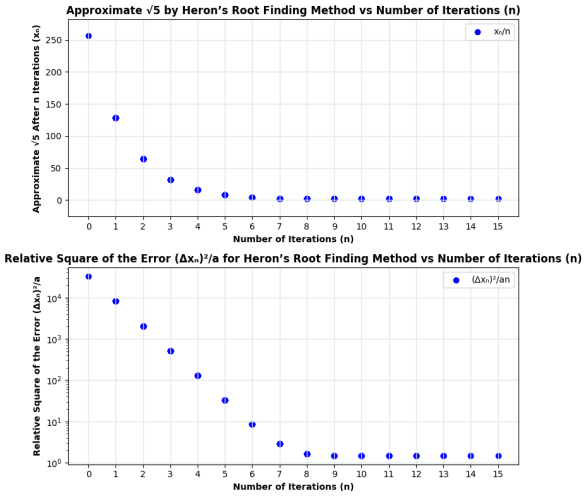

Figure 3: *Heron's Root Finding Method For* $\sqrt{5}$

The initial estimate or x_0 for each of these examples was $x_0 = 256$ which explains that the relative error of the earlier estimates are much larger for a = 2, than a = 3, or a = 5. The relative error decreases exponentially until it hits a minimum. The approximation in each case reaches the minimum relative error at iterations 12, 9, 8 respectively which is likely due to the fact that the relative error of x_0 is much larger for the former than the latter a values.

Part B: Numerical Accuracy

The underflow and overflow limits that were experimentally determined by iteratively halving and doubling. The results being: 2^{-1074} or 5×10^{-324}, and 2^{1023} or $8.98846567431158307 \times 10^{307}$ respectively align with those theorised in "Computational Problems for Physics" by Landau and Paez of 4.9×10^{-324} and 1.8×10^{308} respectively. The experimentally derived values are one operation away from being considered 0 or "inf" respectively ("inf" being considered infinite or not a number) with the limits aligning with those theorized. The limits in both directions were therefore determined for floating point numbers.

The machine precision for floating point numbers was experimentally determined by iteratively halving until the computer calculates the following: $\epsilon + 1.0 = 1.0$. The theory outlined in "Computational Problems for Physics" by Landau and Paez being that ϵ in the order of 10^{16} aligns with the experimentally determined values of the machine precision ϵ being 2^{-52} or $2.220446049250313 \times 10^{16}$. Therefore the machine precision for floating point numbers was determined.

The machine precision for complex numbers was experimentally determined by iteratively halving until the computer calculates the following: $\epsilon + 1.0 + 1.0i = 1.0 + 1.0i$. The machine precision for complex numbers was experimentally determined to be $2.220446049250313 \times 10^{-16} + 2.220446049250313 \times 10^{-16}i$ which makes sense as the real and imaginary portions of the complex numbers are saved as floating point numbers and therefore it makes sense that the machine precision would hold the same value.

The central difference method: $f'_{central}(x) \approx \frac{f(x+h)-f(x-h)}{2h}$,
and the forward difference method: $f'_{forward}(x) \approx \frac{f(x+h)-f(x)}{h}$,
were applied to the functions $cos(t)$ and e^t. This was performed for the input values of t = 0.1, 1, and 100. This was repeated iteratively for iteratively halved step sizes h. The results were printed as a function of the step size h.

The results of the first, an intermediate and the last of these prints are included below:

First:

```
Inputs: t = [0.1, 1, 100], h = 1.0
Central Difference Derivatives for cos(t): [-0.08400692342254354, -0.7080734182735712, 0.42609199469751063]
with a relative error of [0.15852901519210347, 0.15852901519210352, 0.15852901519210344]
Forward Difference Derivatives for cos(t): [-0.5414080438524485, -0.9564491424152821, 0.02968599750047629]
with a relative error of [4.423114444412334, 0.13663947977200253, 0.9413743842583474]
Central Difference Derivatives for e: [1.298798182102917, 3.194528049465325, 3.159078473716686e+43]
with a relative error of [0.17520119364380143, 0.17520119364380154, 0.17520119364380135]
Forward Difference Derivatives for e: [1.8989951058707857, 4.670774270471606, 4.618942837551931e+43]
with a relative error of [0.7182818284590453, 0.7182818284590456, 0.7182818284590448]
```

Intermediate:

```
Inputs: t = [0.1, 1, 100], h = 0.001953125
Central Difference Derivatives for cos(t): [-0.09983335317448905, -0.841470449815148, 0.5063653191712092]
with a relative error of [6.357824988741299e-07, 6.357827639994628e-07, 6.357827693268397e-07]
Forward Difference Derivatives for cos(t): [-0.100805036620784, -0.8419980886180269, 0.5055232111652117]
with a relative error of [0.009732412318342801, 0.0006264075881960102, 0.0016636791206859978]
Central Difference Derivatives for e: [1.1051716207245477, 2.718283556696406, 2.688118850875357e+43]
with a relative error of [6.357830164191552e-07, 6.357829944940007e-07, 6.357830150310969e-07]
Forward Difference Derivatives for e: [1.1062508895422525, 2.720938129638398, 2.690743966106057 5e+43]
with a relative error of [0.0009771985933951664, 0.0009771985934434264, 0.0009771985934166286]
```

Final:

```
Inputs: t = [0.1, 1, 100], h = machine precision: 2.220446049250313e-16
Central Difference Derivatives for cos(t): [-0.25, -0.75, 0.0]
with a relative error of [1.5041715329086942, 0.10870367066640908, 1.0]
Forward Difference Derivatives for cos(t): [-0.5, -1.0, 0.0]
with a relative error of [4.008343065817388, 0.1883951057781212, 1.0]
Central Difference Derivatives for e: [1.0, 3.0, 0.0]
with a relative error of [0.0951625819640405, 0.10363832351432703, 1.0]
Forward Difference Derivatives for e: [1.0, 4.0, 0.0]
with a relative error of [0.0951625819640405, 0.4715177646857694, 1.0]
```

The relative errors in the first examples with a step size $h = 1.0$ has a high relative error whilst the intermediate example with a step size $h = 0.001953125$ has a low relative error and the relative error calculations in the final examples seem to break down being relatively high for the final example with step size $h =$ machine precision: $2.220446049250313 \times 10^{-16}$.

The relative errors were then graphed for equally spaced discrete step sizes h:

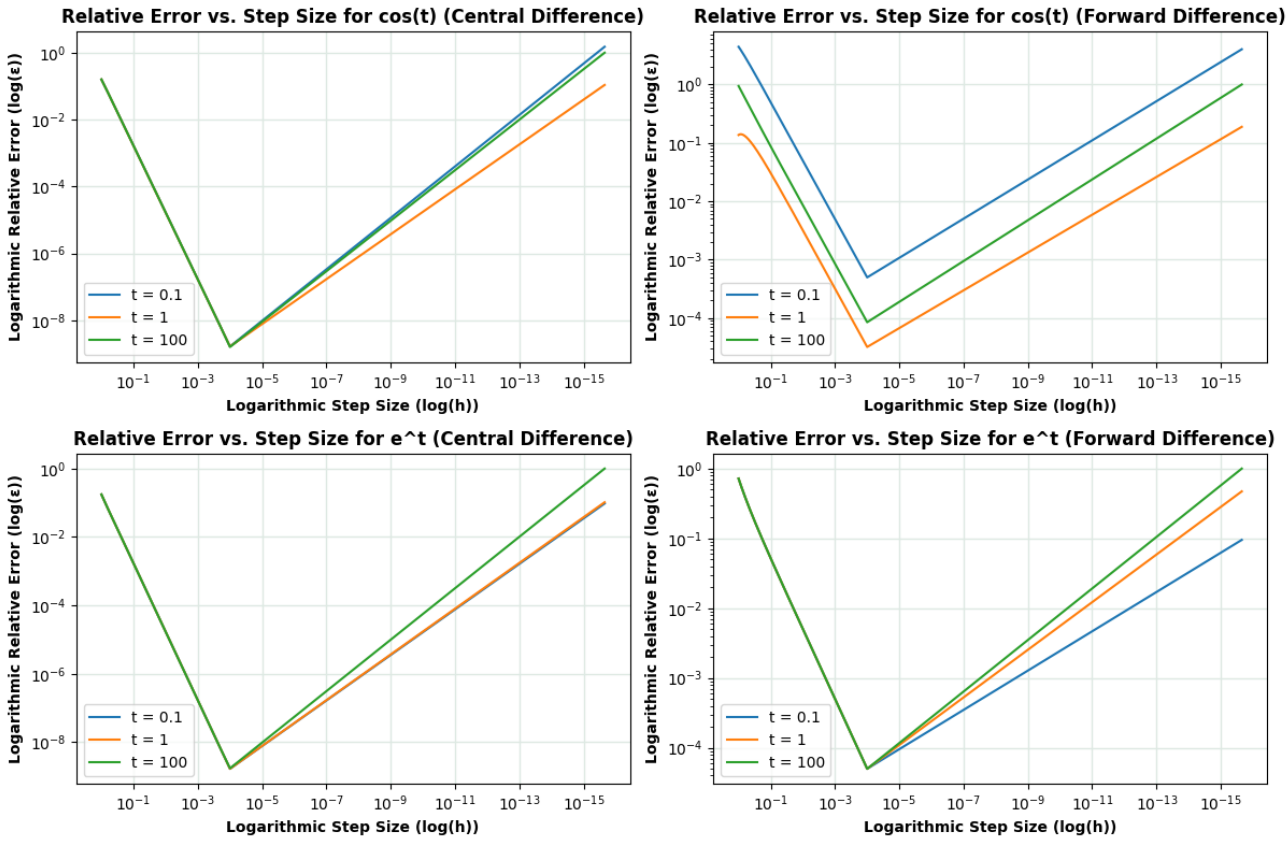

Figure 4: *Relative Error vs Step Size for Central and Forward Difference Methods*

As indicated by the printed values and minima of the figure above; One may not achieve an accuracy for the derivative that is near the machine precision as the computational errors begin to amount as the step size h decreases and result in relative errors comparable to those for very large values of h.

The central difference method behaves equivalently or better than the forward difference method for all examples plotted which is in line with theory as the approximation error is better minimised.

References:

"Computational Problems for Physics" by Landau and Paez, CRC ´Press 2018.