Understanding Logarithms and the methods and tools for calculating and using them

Jose Portillo

Abstract: Choosing $e^y = x$ as the implicit equation to solve for $\ln(x)$ simplifies derivatives, reduces computational overhead, and improves the stability and efficiency of root-finding algorithms such as Newton-Raphson. Furthermore, base-10 logarithms can be obtained simply by scaling $\ln(x)$ making the natural logarithm the preferred starting point in most numerical applications. This easy was written with a little help from an AI GPT making this a use case of human subject using the AI as copilot for math syntax checking and LaTeX formatting.

0. A Feynman Perspective on Basic Algebra

To discuss this subject we start in the middle. We suppose that we already know what integers are, what zero is, and what it means to increase a number by one unit. You may say, "That is not in the middle!" But it is the middle from a mathematical standpoint, because we could go even further back and describe the theory of sets in order to derive some of these properties of integers. But we are not going in that direction, the direction of mathematical philosophy and mathematical logic, but rather in the other direction, from the assumption that we know what integers are and we know how to count.

If we start with a certain number a, an integer, and we count successively one unit b times, the number we arrive at we call $a + b$, and that defines addition of integers.

Once we have defined addition, then we can consider this: if we start with nothing and add a to it, b times in succession, we call the result multiplication of integers; we call it $b \times a$.

Now we can also have a succession of multiplications: if we start with 1 and multiply by a, b times in succession, we call that raising to a power: a^b.

Now as a consequence of these definitions it can be easily shown that all of the following relationships are true:

(a) $a + b = b + a$

(b) $a + (b + c) = (a + b) + c$

(c) $ab = ba$

(d) $a(b + c) = ab + ac$

(e) $(ab)c = a(bc)$

(f) $(ab)^c = a^c \cdot b^c$

(g) $a^b \cdot a^c = a^{b+c}$

(h) $(a^b)^c = a^{(bc)}$

(i) $a + 0 = a$

(j) $a \cdot 1 = a$

(k) $a^1 = a$

These results are well known and we shall not belabor the point, we merely list them. Of course, 1 and 0 have special properties; for example, $a + 0 = a$, $a \cdot 1 = a$, and $a^1 = a$.

In this discussion we must also assume a few other properties like continuity and ordering, which are very hard to define; we will let the rigorous theory do it. Furthermore, it is definitely true that we have written down too many "rules"; some of them may be deducible from the others, but we shall not worry about such matters.

In addition to the direct operations of addition, multiplication, and raising to a power, we have also the inverse operations, which are defined as follows. Let us assume that a and c are given, and that we wish to find what values of b satisfy such equations as $a + b = c$, $ab = c$, $a^b = c$.

If $a + b = c$, b is defined as $c - a$, which is called subtraction. The operation called division is also clear: if $ab = c$, then $b = \frac{c}{a}$ defines division—a solution of the equation $ab = c$ "backwards."

Now if we have a power $b^a = c$ and we ask ourselves, "What is b?," it is called the ath root of c: $b = \sqrt[a]{c}$. For instance, if we ask ourselves the following question, "What integer, raised to the third power, equals 8?," then the answer is called the cube root of 8; it is 2.

Because b^a and a^b are not equal, there are two inverse problems associated with powers, and the other inverse problem would be, "To what power must we raise 2 to get 8?" This is called taking the *logarithm*. If $a^b = c$, we write $b = \log_a c$.

The fact that it has a cumbersome notation relative to the others does not mean that it is any less elementary,

at least applied to integers, than the other processes. Although logarithms come late in an algebra class, in practice they are, of course, just as simple as roots; they are just a different kind of solution of an algebraic equation.

1. The Concept of Slope

A *Linear Equation* is an algebraic equation

$$y = mx + b$$

where m is the slope and b is the y-intercept. That is the straight line *standard form.*
There is another form called *Point-Slope*

$$y_2 - y_1 = m(x_2 - x_1)$$

so the slope m passing through points (x_1, y_1) and (x_2, y_2) is defined as:

$$m = \frac{y_2 - y_1}{x_2 - x_1} \tag{1}$$

This measures the rate of change of y with respect to x, that is, how steep the line is.

2. Limits and the Derivative (The Origin of h)

To understand how a function changes at a single point, we use the concept of a **limit**.
Let $f(x)$ be a real-valued function. Suppose we want to study how it behaves near a point a. Consider a second point a small distance away: $a + h$, where h is a small increment (positive or negative).
We define the difference quotient:

$$\frac{f(a+h) - f(a)}{h}$$

This represents the average rate of change of f over the interval from a to $a + h$.
As we let $h \to 0$, the average rate approaches the *instantaneous rate of change* the derivative of f at a:

$$f'(a) = \lim_{h \to 0} \frac{f(a+h) - f(a)}{h} \tag{2}$$

This is the slope of the tangent line to the graph of f at the point $x = a$.

3. Tangent Line Approximation (Linearization)

We begin with the familiar formula for the slope m of a straight line passing through two points (x_1, y_1) and (x_2, y_2):

$$m = \frac{y_2 - y_1}{x_2 - x_1}$$

If the graph represents a function $y = f(x)$, then we can interpret this slope as:

$$m = \frac{f(x_2) - f(x_1)}{x_2 - x_1} \tag{3}$$

This is the **average rate of change** of the function between x_1 and x_2, corresponding to the slope of the **secant line** through those points.
To find the rate of change at a **single point**, we let $x_2 = x_1 + h$ and take the limit as $h \to 0$:

$$f'(x_1) = \lim_{h \to 0} \frac{f(x_1 + h) - f(x_1)}{h} \tag{4}$$

This limit gives us the **instantaneous rate of change** of f at x_1, which is the slope of the **tangent line** to the curve at that point.
Using this, the equation of the tangent line at a point x_n becomes:

$$f(x) - f(x_n) = f'(x_n)(x - x_n)$$

Solving for $f(x)$, we obtain the **linear approximation** of the function:

$$f(x) \approx f(x_n) + f'(x_n)(x - x_n) \tag{5}$$

This is conceptually the same as the line equation $y = mx+c$, but now the slope m is the derivative $f'(x_n)$, and the line passes through the point $(x_n, f(x_n))$. In other words, this is the *infinitesimal version* of the straight line formula, tailor-made for smooth functions.

4. The Newton-Raphson Method

> "Let's pretend the curve is a straight line locally."

Suppose we want to find a root of the equation:

$$f(x) = 0$$

We can approximate the function $f(x)$ using a Taylor series expansion centered at a point x_n:

$$f(x) = f(x_n) + f'(x_n)(x - x_n) + \frac{1}{2} f''(x_n)(x - x_n)^2 + \frac{1}{3!} f'''(x_n)(x - x_n)^3 + \dots \tag{6}$$

We begin with an initial guess x_0 and use the tangent line at $x = x_n$ to approximate the curve.
From Section 3, the tangent line at x_n is:

$$f(x) \approx f(x_n) + f'(x_n)(x - x_n)$$

We now solve the approximation $f(x) \approx 0$ to find where the tangent line crosses the x-axis:

$$0 = f(x_n) + f'(x_n)(x_{n+1} - x_n)$$

Solving for x_{n+1}:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \qquad (n = 1, 2, \dots) \tag{7}$$

This is the **Newton-Raphson iteration formula**, which gives a refined estimate of the root. The method is based on the idea that locally, a smooth function can be approximated by its tangent line — turning the nonlinear root-finding problem into a linear one at each step. If the function is sufficiently smooth and the initial guess is close enough to a true root, this iteration converges rapidly.

5. Applying to $e^y = x$

Why Choose $e^y = x$ Over $10^y = x$ for Logarithm Computation?

Both the equations

$$10^y = x \quad \text{and} \quad e^y = x$$

implicitly define logarithms of base 10 and base e, respectively:

$$y = \log_{10}(x), \quad y = \ln(x)$$

Let us compare their derivatives, as this plays a crucial role in numerical methods like Newton-Raphson.

Derivative for $10^y = x$: Differentiating both sides with respect to x, applying the chain rule,

$$10^y \ln(10) \cdot \frac{dy}{dx} = 1$$

which yields

$$\frac{dy}{dx} = \frac{1}{10^y \ln(10)} = \frac{1}{x \ln(10)}$$

Note the extra constant factor $\ln(10) \approx 2.3026$ in the denominator.

Derivative for $e^y = x$: Similarly,

$$e^y \cdot \frac{dy}{dx} = 1$$

giving

$$\frac{dy}{dx} = \frac{1}{e^y} = \frac{1}{x} \tag{8}$$

which is a much simpler expression, free of extra multiplicative constants.

Implications for Numerical Methods: When applying iterative methods like Newton-Raphson to find y such that $a^y = x$, the update step depends on both the function and its derivative:

$$y_{n+1} = y_n - \frac{a^{y_n} - x}{a^{y_n} \ln(a)}$$

for $a = 10$, and

$$y_{n+1} = y_n - \frac{e^{y_n} - x}{e^{y_n}} = y_n - 1 + \frac{x}{e^{y_n}} \tag{9}$$

for $a = e$.
The presence of $\ln(10)$ in the denominator complicates computations and can affect numerical stability and performance.
Choosing $e^y = x$ as the implicit equation to solve for $\ln(x)$ simplifies derivatives, reduces computational overhead, and improves the stability and efficiency of root-finding algorithms such as Newton-Raphson.
Furthermore, base-10 logarithms can be obtained simply by scaling:

$$\log_{10}(x) = \frac{\ln(x)}{\ln(10)}$$

making the natural logarithm the preferred starting point in most numerical applications.
We want to find $y = \ln(x)$, so we define:

$$f(y) = e^y - x$$

Then:

$$f'(y) = e^y$$

Using Newton-Raphson:

$$y_{n+1} = y_n - \frac{e^{y_n} - x}{e^{y_n}} = y_n + \frac{x - e^{y_n}}{e^{y_n}} \tag{10}$$

and so

$$y_{n+1} = y_n - (1 - \frac{x}{e^{y_n}}) \tag{11}$$

6. Two methods for calculating e

The constant e is basic for our $e^y = x$ equation and so for $ln(x)$. There are many ways to approximate the number e. Two notable methods are:

6.1. Compound Interest Formula

One of the earliest ways to define e comes from the compound interest formula. The formula for continuously compounded interest is:

$$A = P\left(1 + \frac{r}{n}\right)^{nt}$$

As $n \to \infty$, this expression converges to:

$$\lim_{n\to\infty}\left(1+\frac{1}{n}\right)^n = e$$

This shows how e arises in the context of compound interest.

6.2. Continued Fraction for e

Another fascinating way to represent e is through its continued fraction expansion:

$$e = [2; 1, 2, 1, 1, 4, 1, 1, 6, 1, 1, 8, 1, \dots]$$

$$2 + \cfrac{1}{1 + \cfrac{1}{2 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{4 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{6 + \cfrac{1}{1 + \cfrac{1}{1 + \frac{1}{8}}}}}}}}}}} \tag{12}$$

This continued fraction converges very quickly to the value of e.

7. Newton-Raphson method example: Compute $\ln(5)$

Let $x = 5$, and start with $y_0 = 1.5$.

$$\begin{aligned}
y_0 &= 1.5 \\
e^{y_0} &\approx 4.4817 \\
y_1 &= 1.5 + \frac{5 - 4.4817}{4.4817} \approx 1.6156 \\
e^{y_1} &\approx 5.0318 \\
y_2 &= 1.6156 + \frac{5 - 5.0318}{5.0318} \approx 1.6093 \\
e^{y_2} &\approx 5.0001 \\
y_3 &= 1.6093 + \frac{5 - 5.0001}{5.0001} \approx 1.60928
\end{aligned}$$

So:

$$\ln(5) \approx 1.60928$$

8. Convert to Base-10 Logarithm

Using the identity:

$$\log_{10}(x) = \frac{\ln(x)}{\ln(10)}$$

And since $\ln(10) \approx 2.30259$, we find:

$$\log_{10}(5) \approx \frac{1.60928}{2.30259} \approx 0.69897$$

9. Python implementation

```
def const_e():
   return 2 + 1/(1 + 1/(2 + 1/(1 + 1/
        (1 + 1/(4 + 1/(1 + 1/
          (1 + 1/(6 + 1/(1 + 1/(1 + 1/(8)))))))))))

def logNeper(x):
   Yn = 0
   for i in range(0, 200):
      Yn = Yn - (1 - (x/(const\_e()**Yn)))
   return Yn

def logBase10(x):
   return logNeper(x) / logNeper(10)

print(f"log(3) = {logBase10(3)}")
```

The Newton-Raphson method is grounded in the most fundamental ideas of calculus:

- The slope (derivative) represents instantaneous change.
- Linear approximation gives us a way to predict nearby function values.
- The Newton-Raphson algorithm uses the tangent line to iteratively home in on a root.

With a good initial guess and a differentiable function, this method converges rapidly and reliably.

Appendix A

A slide rule is an analog device used for calculation that exploits the concept of logarithms. The key idea behind the slide rule is that logarithmic scales can transform multiplication and division operations into addition and subtraction. We'll explore how the logarithmic transformation used in slide rules can be viewed as an isomorphism, preserving mathematical structure.
The most important scales are:

- **L scale**: A linear scale of $\log_{10}(x)$.
- **D scale**: A scale that maps real numbers using their base-10 logarithms.

Step-by-Step Construction

a. Define the Length

Assume the rule is 10 inches long, which equals 254 mm. We will map:

$$x = 1 \to \log_{10}(1) = 0 \Rightarrow \text{Position} = 0 \text{ mm}$$
$$x = 10 \to \log_{10}(10) = 1 \Rightarrow \text{Position} = 254 \text{ mm}$$

b. Logarithmic Table from 1 to 10

x	$\log_{10}(x)$	x	$\log_{10}(x)$
1.00	0.000	1.01	0.004
1.02	0.009	1.03	0.013
1.04	0.018	1.05	0.022
1.06	0.026	1.07	0.031
1.08	0.034	1.09	0.037
1.10	0.041	1.20	0.079
1.30	0.114	1.40	0.146
1.50	0.176	1.60	0.204
1.70	0.230	1.80	0.255
1.90	0.278	2.00	0.301
2.10	0.323	2.20	0.342
2.30	0.361	2.40	0.380
2.50	0.398	2.60	0.414
2.70	0.431	2.80	0.447
2.90	0.463	3.00	0.477
3.10	0.491	3.20	0.505
3.30	0.519	3.40	0.531
3.50	0.544	3.60	0.556
3.70	0.568	3.80	0.579
3.90	0.591	4.00	0.602
4.10	0.612	4.20	0.623
4.30	0.633	4.40	0.643
4.50	0.653	4.60	0.663
4.70	0.672	4.80	0.681
4.90	0.690	5.00	0.699
5.10	0.707	5.20	0.716
5.30	0.724	5.40	0.732
5.50	0.740	5.60	0.748
5.70	0.755	5.80	0.762
5.90	0.769	6.00	0.778
6.10	0.784	6.20	0.791
6.30	0.798	6.40	0.804
6.50	0.810	6.60	0.816
6.70	0.822	6.80	0.828
6.90	0.834	7.00	0.845
7.10	0.851	7.20	0.857
7.30	0.863	7.40	0.869
7.50	0.875	7.60	0.881
7.70	0.886	7.80	0.892
7.90	0.897	8.00	0.903
8.10	0.908	8.20	0.913
8.30	0.918	8.40	0.923
8.50	0.928	8.60	0.933
8.70	0.938	8.80	0.943
8.90	0.948	9.00	0.954
9.10	0.959	9.20	0.964
9.30	0.969	9.40	0.974
9.50	0.979	9.60	0.984
9.70	0.989	9.80	0.994
9.90	0.999	10.00	1.000

Now, we need all those 1 to 10 numbers and its artificial numbers! So, to construct a table of logarithmic values, we can evaluate $\ln(x)$ for numbers between 1 and 10 (including 0.1, 0.01). This will give us the artificial logarithmic values necessary for use on a slide rule. These values are typically represented as a series of digits (the mantissa) that correspond to each value of $\ln(x)$.
Here is the logarithmic table for numbers from 1 to 10 that will be used to construct the L and D scales. These values represent $\log_{10}(x)$ for values of x from 1 to 10.

These values are key to constructing the logarithmic scale for the slide rule. The distances between the numbers on the scale are proportional to the differences between the logarithms of the numbers. By marking these values along the rule, we can perform multiplications and divisions by simply adding or subtracting distances on the scale, making complex calculations easier and faster.

c. A Briggs-Style Algorithm for Base-10 Logarithms

In *The Art of Computer Programming*, Vol. 1, Donald Knuth describes an algorithm for computing logarithms that is, in his own words, "very similar to the method used for division in computer hardware." More importantly, he notes that this idea goes back to Henry Briggs, who used it (in decimal rather than binary form) to compute his famous logarithm tables published in 1624.
The essence of the method is simple: build up a number x by multiplying small rational corrections slightly greater than 1, and accumulate the logarithms of those corrections. The clever part is that each correction follows a predictable form:

$$\frac{b^k}{b^k - 1}$$

For base-10, this becomes:

$$\frac{10^k}{10^k - 1}$$

Each multiplication by such a fraction nudges the product upward. At each step, we test whether including the factor overshoots our target number x. If not, we apply it and add the corresponding logarithm to our result.
Algorithm Sketch (Decimal Version)
Given a number $x > 1$, initialize:

- $y \leftarrow 0$ (the accumulated logarithm)
- $P \leftarrow 1$ (the running product)

Then, for $k = 1, 2, 3, \ldots$:

1. Compute the correction factor:

$$f_k = \frac{10^k}{10^k - 1}$$

2. Multiply: $P' = P \cdot f_k$
3. If $P' \leq x$, accept the correction:

$$P \leftarrow P', \quad y \leftarrow y + \log_{10}(f_k)$$

4. Otherwise, skip this k and continue.

Example: Approximating $\log_{10}(5)$
Let's approximate $\log_{10}(5)$ by this method.

Step	Fraction f_k	Product P	Log sum y
1	$\frac{10}{9} \approx 1.111111$	1.111111	0.045757
2	$\frac{100}{99} \approx 1.010101$	1.122223	0.050078
3	$\frac{1000}{999} \approx 1.001001$	1.123345	0.050512
4	$\frac{10000}{9999} \approx 1.000100$	1.124468	0.050652
⋮	⋮	⋮	⋮

This sum gradually converges toward $\log_{10}(5) \approx 0.69897$. The more correction terms we include, the more accurate the result.
Henry Briggs computed extensive base-10 logarithm tables using this very method — long before the invention of electronic computing. Knuth's algorithm is simply this process in base-2, adapted to binary arithmetic and suited for computer implementation. It's a beautiful example of how an idea from the 1600s still underpins modern numerical methods.

c. Construct the L Scale

"What we are doing is manipulating lengths and using those lengths to tell us something about numbers. Lengths are not numbers, of course, but in certain ways they follow the same rules that numbers do. Lengths have properties that are analogous to those of numbers in some ways. If we then use lengths to work out, or compute the answers to number problems, we are using the foot rule as an Analog Computer"
— Asimov, Isaac, *An Easy Introduction to the Slide Rule*, 1965

To build the L scale:

1. Choose values like 0.01, 0.1, 1, 2, ..., 10.
2. Compute position:

$$\text{Position} = 254 \times \log_{10}(x)$$

3. Mark that position on your physical scale and label it with $\log_{10}(x)$.

Example:

$$\log_{10}(2) \approx 0.301 \Rightarrow \text{Position} \approx 76.5 \text{ mm}$$

In this example 254 is the Constant Factor!

d. Construct the D Scale

For the D scale:

1. Use the same physical positions from the L scale.
2. For each position p, compute:

$$x = 10^{\frac{p}{254}}$$

3. Label that position with the corresponding real number.

The equation $x = 10^{\frac{p}{254}}$ allows us to recover the real number x from a given position p on the L scale.
The logarithmic scale on a slide rule encodes the logarithm of a number such that equal distances on the scale

correspond to equal ratios between numbers. To construct the scale, the logarithms of numbers are spaced along a linear scale, with the distance between the numbers representing the difference in their logarithms. This transformation can be described as a function $T(x) = log(x)$, where x is the input number. Specifically, the transformation is the natural logarithm function:

$$T(x) = log(x)$$

Here, T is a function that maps numbers to their logarithms, and this transformation is critical in how the slide rule performs its calculations.
In our context, the transformation $T(x) = \log(x)$ acts as an isomorphism between multiplication in the positive real numbers, that is, the group $(\mathbb{R}^+, \cdot)$, and addition of distances on the logarithmic scale, which corresponds to the group $(\mathbb{R}, +)$.

This transformation preserves structure because:

- It is **bijective** on $\mathbb{R}^+$: each positive real number has a unique logarithm, and each real number is the logarithm of exactly one positive number.

- It converts products into sums:

$$\log(ab) = \log(a) + \log(b)$$

- It maps the identity of multiplication (which is 1) to the identity of addition (which is 0):

$$\log(1) = 0$$

This is precisely what makes the logarithmic scale on a slide rule so powerful: multiplying two numbers corresponds to sliding and adding their logarithmic distances — a visual and mechanical realization of this isomorphism.
A function is an isomorphism if it is both injective (one-to-one) and surjective (onto). Let's check these properties for the logarithmic transformation:

Injective: The function $log(x)$ is injective because if $log(x_1) = log(x_2)$, then it must be the case that $x_1 = x_2$.

Surjective: The function $log(x)$ is surjective because for every real number y, there exists a positive number $x = e^y$ such that $log(x) = y$.

The logarithmic function $\log : \mathbb{R}^+ \to \mathbb{R}$ is more than just a bijection between sets. It is an isomorphism between two algebraic structures:

- The set $\mathbb{R}^+$ under multiplication, denoted $(\mathbb{R}^+, \cdot)$
- The set $\mathbb{R}$ under addition, denoted $(\mathbb{R}, +)$

This means:

- **Injective**: $\log(a) = \log(b) \Rightarrow a = b$
- **Surjective**: For every $r \in \mathbb{R}$, there exists $x \in \mathbb{R}^+$ such that $\log(x) = r$
- **Structure-preserving**: $\log(ab) = \log(a) + \log(b)$

Therefore, log is a group isomorphism: it maps multiplicative relationships into additive ones while preserving identity and inverse elements.
An isomorphism must also preserve the structure of operations. In the case of the slide rule, the relevant operation is multiplication. The logarithmic transformation preserves this operation because:

$$T(x \cdot y) = log(x \cdot y) = log(x) + log(y) = T(x) + T(y)$$

Thus, the logarithmic transformation preserves the multiplication operation by converting it into addition.

e. Plotting L and D scales with PYTHON

Simple Ruler plot experiment.
Itś fractal rather than strictly recursive. The difference lies in the nature of the pattern:
Fractals involve self-similarity at different scales, which is what we see with the ruler markings. Each subdivision has a similar structure, but the sizes change as you zoom in or out.
Recursion would be a process where the pattern is defined in terms of itself (like in the case of recursive functions, or mathematical sequences like the Fibonacci series).
So, while the rulerś pattern has that repeating, self-similar vibe $likeafractal$, it doesnt́ quite follow the recursive rule of generating each part from a smaller instance of the same process.
Fractal is definitely the term here!
Now, lets make a Logarithm scale based on the normal ruler.
Procedure for using the Logarithmic Scale to find the Base-10 Log of a number NN:
Find N on the Logarithmic Scale:
Locate the position of the number N on the logarithmic scale $thebottomruler$.
Since the logarithmic scale is logarithmic, the spacing between the marks shrinks as you move rightward, which corresponds to the powers of 10.
Look at the Corresponding Position on the Linear Ruler:
Once wev́e located the mark for N on the logarithmic scale, weĺl then look directly above it on the linear ruler $thetopruler$.
The corresponding position on the linear ruler will give us a representation of the logarithmic value of the number.

Example:
Lets say we want to find

$$log10(100)$$

On the logarithmic scale =bottom ruler=, locate the mark for 100. The position should be near the mark for 102.
Then, look directly above the 100 mark on the linear ruler =top ruler=. The mark at that position will correspond to 100 on the linear scale, which tells us that

$$log10(100) = 2$$

So, in this example, the logarithmic scale is used to find the number in terms of its logarithmic value, and the linear ruler shows the actual number corresponding to the logarithmic position.
Why it works:
The logarithmic scale represents exponential growth, so each tick corresponds to a power of 10 *i.e.*, $101, 102, 103, etc.$.
The linear ruler represents those values in a straightforward linear fashion.
In essence, wefe using the logarithmic scale to map out numbers exponentially and the linear scale to read them off as actual values. This is the core idea behind how a slide rule works!

```
import numpy as np
import matplotlib.pyplot as plt

def plot_ruler_and_log_scale(length=100, divisions=10):
    fig, axs = plt.subplots(2, 1, figsize=(10, 4))

    ax = axs[0]
    ax.set_xlim(0, length)
    ax.set_ylim(0, 1)
    for i in range(0, length+1, length//divisions):
        ax.plot([i, i], [0, 0.5], color='black', lw=2)

    c = 1
    for i in range(0, length+1, length//(divisions*10)):
        if c == 6:
            ax.plot([i, i], [0, 0.30], color='black', lw=1)
            c = 1
        else:
            ax.plot([i, i], [0, 0.20], color='black', lw=0.5)
        c += 1

    ax.set_xticks(range(0, length+1, length//divisions))
    ax.set_xticklabels([str(i) for i in range(0, divisions+1)])
    ax.set_title('Linear Ruler')

    ax = axs[1]
    ax.set_xlim(1, length)
    ax.set_ylim(0, 1)
    for i in range(1, divisions + 1):
        position = length *
           (np.log10(i) / np.log10(divisions))
        ax.plot([position, position], [0, 0.5],
color='black', lw=2)
    c = 1
    for i in range(1, divisions * 10):
        position = length *
           (np.log10(i / 10) / np.log10(divisions))
        if c == 6:
            ax.plot([position, position], [0, 0.30],
color='black', lw=1)
            c = 1
        else:
            ax.plot([position, position], [0, 0.20],
color='black', lw=0.5)
        c += 1

    ax.set_xticks([length *
              (np.log10(i) / np.log10(divisions))
             for i in range(1, divisions+1)])
    ax.set_xticklabels([str(int(i))
                  for i in range(1, divisions+1)])
    ax.set_title('Logarithmic Scale')

    for ax in axs:
        ax.axis('on')

    plt.tight_layout()
    plt.show()
```

This chunk of code produces

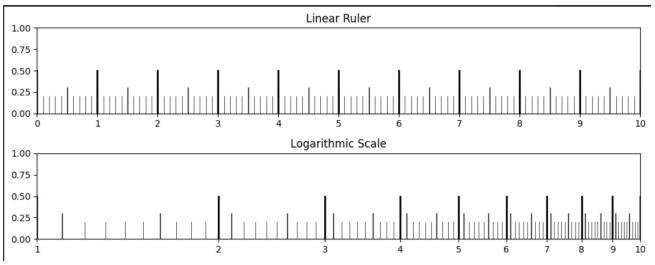

Figure 1: Ruler and logarithmic scale.

Appendix B

1. Geometric Interpretation: Logarithm as Area Under a Curve

By definition, the natural logarithm can be expressed as the integral

$$\ln(a) = \int_1^a \frac{1}{x}\, dx,$$

which represents the area under the curve $y = \frac{1}{x}$ from 1 to a.
If we increase a by a small amount Δa, the area increases approximately by the area of a thin vertical slice of height $\frac{1}{a}$ and width Δa:

$$\Delta(\ln(a)) \approx \frac{1}{a} \cdot \Delta a.$$

Taking the limit as $\Delta a \to 0$, we get

$$\frac{d}{da} \ln(a) = \lim_{\Delta a \to 0} \frac{\Delta(\ln(a))}{\Delta a} = \frac{1}{a}.$$

2. Calculus Derivation Using the Chain Rule

The natural logarithm is the inverse function of the exponential:

$$b = \ln(a) \quad \Longleftrightarrow \quad e^b = a.$$

Our goal is to find $\frac{db}{da}$.
Differentiate both sides with respect to a. Since $b = b(a)$, e^b is a composition of functions. By the chain rule,

$$\frac{d}{da}e^b = \frac{d}{db}e^b \cdot \frac{db}{da} = e^b \cdot \frac{db}{da}.$$

On the right-hand side,

$$\frac{d}{da}a = 1.$$

Equating both,

$$e^b \cdot \frac{db}{da} = 1 \implies \frac{db}{da} = \frac{1}{e^b}.$$

Substituting back $e^b = a$, we conclude

$$\frac{d}{da}\ln(a) = \frac{1}{a}.$$

3. Limit Definition from First Principles

Starting from the definition of the derivative,

$$\frac{d}{da}\ln(a) = \lim_{h\to 0}\frac{\ln(a+h) - \ln(a)}{h}.$$

Using the logarithm property,

$$= \lim_{h\to 0}\frac{\ln\left(\frac{a+h}{a}\right)}{h} = \lim_{h\to 0}\frac{\ln\left(1+\frac{h}{a}\right)}{h}.$$

Letting $u = \frac{h}{a}$, so $h = au$,

$$= \lim_{u\to 0}\frac{\ln(1+u)}{au} = \frac{1}{a}\lim_{u\to 0}\frac{\ln(1+u)}{u}.$$

The standard limit $\lim_{u\to 0}\frac{\ln(1+u)}{u} = 1$ yields

$$\frac{d}{da}\ln(a) = \frac{1}{a}.$$

Intuition: The Difference Quotient as a Mathematical Microscope

The difference quotient

$$\frac{f(x+h) - f(x)}{h}$$

is like a **mathematical microscope** focused on the behavior of f right near the point x.

How the analogy works:

- h is the zoom knob.
- When h is large, we're looking at a wide area—so you see a rough slope, a big chunk of the curve.
- When we turn the knob and make h smaller, we zoom in closer and closer on the point x.
- The derivative $f'(x)$ is the limit of this zoomed-in slope as $h \to 0$, i.e., when we zoom in infinitely close.

At that infinite zoom, the curve looks like a straight line—the tangent line—and the slope of this line is the derivative.

So in short:

- The formula
  $$\frac{f(x+h) - f(x)}{h}$$
  is the microscope's lens focusing on the function's behavior near x.
- The limit $h \to 0$ is turning the zoom all the way in.

This analogy provides a concrete way to visualize the limit process and the meaning of differentiation.