Elementary function

From Wikipedia, the free encyclopedia

For the complexity class, see ELEMENTARY. For the logical system, see Elementary function arithmetic.

In mathematics, an **elementary function** is a function of a single variable composed of particular simple functions.

Elementary functions are typically defined as a sum, product, and/or composition of finitely many polynomials, rational functions, trigonometric and exponential functions, and their inverse functions (including arcsin, log, $x^{1/n}$).[1]

Elementary functions were introduced by Joseph Liouville in a series of papers from 1833 to 1841.[2][3][4] An algebraic treatment of elementary functions was started by Joseph Fels Ritt in the 1930s.[5]

Basic examples [edit]

The elementary functions (of x) include:

- Constant functions: 2, π, e, etc.
- Powers of x: x, x^2, x^3, etc.
- Roots of x: $\sqrt{x}$, $\sqrt[3]{x}$, etc.
- Exponential functions: e^x
- Logarithms: $\log x$
- Trigonometric functions: $\sin x$, $\cos x$, $\tan x$, etc.
- Inverse trigonometric functions: $\arcsin x$, $\arccos x$, etc.
- Hyperbolic functions: $\sinh x$, $\cosh x$, etc.
- Inverse hyperbolic functions: $\operatorname{arsinh} x$, $\operatorname{arcosh} x$, etc.
- All functions obtained by adding, subtracting, multiplying or dividing any of the previous functions[6]
- All functions obtained by composing previously listed functions

Some elementary functions, such as roots, logarithms, or inverse trigonometric functions, are not entire functions and may be multivalued.

Composite examples [edit]

Examples of elementary functions include:

- Addition, e.g. (x+1)
- Multiplication, e.g. ($2x$)
- Polynomial functions
- $\frac{e^{\tan x}}{1+x^2}\sin\left(\sqrt{1+(\ln x)^2}\right)$
- $-i\ln(x+i\sqrt{1-x^2})$

The last function is equal to $\arccos x$, the inverse cosine, in the entire complex plane.

All monomials, polynomials and rational functions are elementary. Also, the absolute value function, for real x, is also elementary as it can be expressed as the composition of a power and root of x: $|x| = \sqrt{x^2}$.

$x \times x = x^2$
$x \times x \times x = x^3$

^2x
^3x

$^2 4 = 2$
$^3 8 = 2, \quad \leftarrow \quad 2\times2\times2 = 2^3 = 8$

$\sqrt[2]{4} = 2$
$\sqrt[3]{8} = 2$

Exponential function

From Wikipedia, the free encyclopedia

In mathematics, an **exponential function** is a function of the form

$$f(x) = ab^x,$$

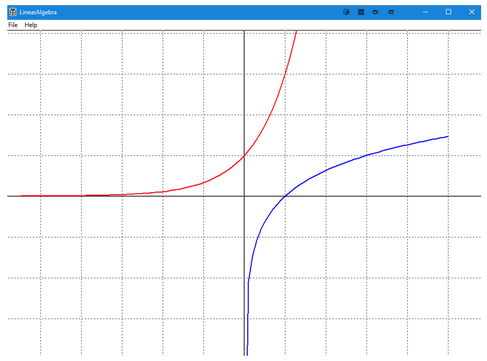
2^x
3^x
10^x

$_2x$
$_3x$
$_{10}x$

$_2 8 = 3, \quad \leftarrow \quad 2\times2\times2 = 2^3 = 8$
$_3 9 = 2$
$_{10} 10000 = 4$

$\log_2 8 = 3$
$\log_3 9 = 2$
$\log_{10} 10000 = 4$

$\log_2(2^x) = x$
$\log_2(2^3) = 3$
$2^{\log_2 8} = x$
$2^{\log_2 8} = 8$



Logarithm

From Wikipedia, the free encyclopedia

In mathematics, the **logarithm** is the inverse function to exponentiation. That means the logarithm of a given number x is the exponent to which another fixed number, the *base* b, must be raised, to produce that number x. In the simplest case, the logarithm counts the number of occurrences of the same factor in repeated multiplication; e.g., since $1000 = 10 \times 10 \times 10 = 10^3$, the "logarithm base 10" of 1000 is 3, or $\log_{10}(1000) = 3$. The logarithm of x to *base* b is denoted as $\log_b(x)$, or without parentheses, $\log_b x$, or even without the explicit base, $\log x$, when no confusion is possible, or when the base does not matter such as in big O notation.

Examples [edit]

- $\log_2 16 = 4$, since $2^4 = 2\times2\times2\times2 = 16$.
- Logarithms can also be negative: $\log_2 \frac{1}{2} = -1 \quad$ since $\quad 2^{-1} = \frac{1}{2^1} = \frac{1}{2}$.
- $\log_{10} 150$ is approximately 2.176, which lies between 2 and 3, just as 150 lies between $10^2 = 100$ and $10^3 = 1000$.
- For any base b, $\log_b b = 1$ and $\log_b 1 = 0$, since $b^1 = b$ and $b^0 = 1$, respectively.

Product, quotient, power, and root [edit]

The logarithm of a product is the sum of the logarithms of the numbers being multiplied; the logarithm of the ratio of two numbers is the difference of the logarithms. The logarithm of the p-th power of a number is p times the logarithm of the number itself; the logarithm of a p-th root is the logarithm of the number divided by p. The following table lists these identities with examples. Each of the identities can be derived after substitution of the logarithm definitions $x = b^{\log_b x}$ or $y = b^{\log_b y}$ in the left hand sides.[1]

	Formula	Example
Product	$\log_b(xy) = \log_b x + \log_b y$	$\log_3 243 = \log_3(9\cdot27) = \log_3 9 + \log_3 27 = 2+3 = 5$
Quotient	$\log_b \frac{x}{y} = \log_b x - \log_b y$	$\log_2 16 = \log_2 \frac{64}{4} = \log_2 64 - \log_2 4 = 6-2 = 4$
Power	$\log_b(x^p) = p\log_b x$	$\log_2 64 = \log_2(2^6) = 6\log_2 2 = 6$
Root	$\log_b \sqrt[p]{x} = \frac{\log_b x}{p}$	$\log_{10}\sqrt{1000} = \frac{1}{2}\log_{10} 1000 = \frac{3}{2} = 1.5$

Change of base [edit]

The logarithm $\log_b x$ can be computed from the logarithms of x and b with respect to an arbitrary base k using the following formula:

$$\log_b x = \frac{\log_k x}{\log_k b}.$$

Derivation of the conversion factor between logarithms of arbitrary base [hide]

Starting from the defining identity

$$x = b^{\log_b x}$$

we can apply $\log_k$ to both sides of this equation, to get

$$\log_k x = \log_k\left(b^{\log_b x}\right) = \log_b x \cdot \log_k b.$$

Solving for $\log_b x$ yields:

$$\log_b x = \frac{\log_k x}{\log_k b},$$

showing the conversion factor from given $\log_k$-values to their corresponding $\log_b$-values to be $(\log_k b)^{-1}$.

Implementation: Draw Exponential Function and Log Function

std::pow, std::powf, std::powl

Defined in header <cmath>

float	pow (float base, float exp);	(1)	
float	powf(float base, float exp);		(since C++11)
double	pow (double base, double exp);	(2)	
long double	pow (long double base, long double exp);	(3)	
long double	powl(long double base, long double exp);		(since C++11)
float	pow (float base, int iexp);	(4)	(until C++11)
double	pow (double base, int iexp);	(5)	(until C++11)
long double	pow (long double base, int iexp);	(6)	(until C++11)
Promoted	pow (Arithmetic1 base, Arithmetic2 exp);	(7)	(since C++11)

```
double ExpFunction(double base_, double x)
{
    return std::pow(base_, x);
}
```

std::log10, std::log10f, std::log10l

Defined in header <cmath>

float	log10 (float arg);	(1)	
float	log10f(float arg);		(since C++11)
double	log10 (double arg);	(2)	
long double	log10 (long double arg);	(3)	
long double	log10l(long double arg);		(since C++11)
double	log10 (IntegralType arg);	(4)	(since C++11)

```
double LogFunction(double base_, double x)
{
    return std::Log10(x) / std::Log10(base_);
}
```

Differentiation of a Exponential Function

$$\frac{d}{dx}a^x = \lim_{h\to0}\frac{a^{x+h}-a^x}{h}$$
$$\lim_{h\to0}\frac{a^x a^h - a^x}{h}$$
$$a^x \lim_{h\to0}\frac{a^h-1}{h}$$
$$f'(a^x) = ca^x$$

$$f'(a) = \lim_{h\to0}\frac{f(a+h)-f(a)}{h}.$$

- Elementary Number
  $0, 1, i, e, \pi$
- Elementary Function

Function Type	Differentiation	Inverse	Differentiation
Power: $x^2,\ x^n$	$2x^1, nx^{n-1}$	Root: $\sqrt{x}$	$\frac{1}{2\sqrt{x}}$
Exponential: e^x	e^x	Log: $\ln x$	$\frac{1}{x}$
Trigonometric: $\cos(x), \sin(x)$	$-\sin(x), \cos(x)$	Inverse Trigonometric: $acos(x), asin(x)$	...

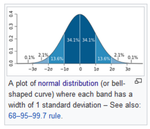
Standard deviation

From Wikipedia, the free encyclopedia

For other uses, see Standard deviation (disambiguation).

In statistics, the **standard deviation** is a measure of the amount of variation or dispersion of a set of values.[1] A low standard deviation indicates that the values tend to be close to the mean (also called the expected value) of the set, while a high standard deviation indicates that the values are spread out over a wider range.

Standard deviation may be abbreviated **SD**, and is most commonly represented in mathematical texts and equations by the lower case Greek letter sigma **σ**, for the population standard deviation, or the Latin letter s, for the sample standard deviation.[2]

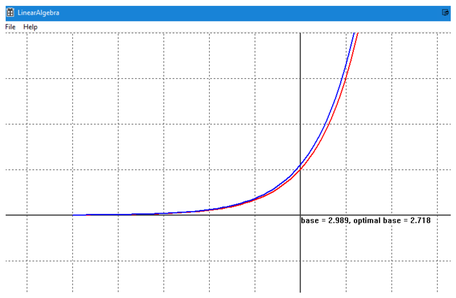


A plot of normal distribution (or bell-shaped curve) where each band has a width of 1 standard deviation – See also: 68–95–99.7 rule.

$$s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2},$$[2][7]

where $\{x_1, x_2, \ldots, x_N\}$ are the observed values of the sample items, $\bar{x}$ is the mean value of these observations, and N is the number of observations in the sample.

```
double GetStdDeviation( double base_, double beginX, double endX, double xstep )
{
    std::vector<double>     vecDiff;

    double x = beginX;
    double ydiff;
    double N = 0;
    while(x < endX)
    {
        ydiff = ExpFunction( base_, x ) - SymmetricDifference2( &ExpFunction, base_, x );
        x += xstep;
        N += 1;
        vecDiff.push_back( ydiff );
    }//while

    double sum = 0;
    for(const double diff : vecDiff)
    {
        sum += ( diff * diff );
    }

    return sqrt( sum / ( N - 1 ) );
}


    static double minStdDevSoFar = DBL_MAX;
    static double optimalBaseSoFar = 0;
    double stddev = GetStdDeviation( base_, xbegin, xend, xstep );
    if(stddev < minStdDevSoFar)
    {
        optimalBaseSoFar = base_;
        minStdDevSoFar = stddev;
    }

    double xText = KVectorUtil::g_screenCoordinate.origin.x + 1;
    double yText = KVectorUtil::g_screenCoordinate.origin.y + 1;

    char buffer[80];
    sprintf_s( buffer, "base = %1.3f, optimal base = %1.3f", base_, optimalBaseSoFar );
    ::TextOutA(hdc, (int)xText, (int)yText, buffer, strlen(buffer));
```



e (mathematical constant)

From Wikipedia, the free encyclopedia

"Euler's number" redirects here. For other uses, see List of things named after Leonhard Euler § Numbers.
"E (number)" redirects here. For the codes representing food additives, see E number.

The number e, known as Euler's number, is a mathematical constant approximately equal to 2.71828, and can be characterized in many ways. It is the base of the natural logarithm.[1][2][3] It is the limit of $(1 + 1/n)^n$ as n approaches infinity, an expression that arises in the study of compound interest. It can also be calculated as the sum of the infinite series[4][5]

$$e = \sum_{n=0}^{\infty}\frac{1}{n!} = \frac{1}{1} + \frac{1}{1} + \frac{1}{1\cdot2} + \frac{1}{1\cdot2\cdot3} + \cdots$$

It is also the unique positive number a such that the graph of the function $y = a^x$ has unit slope at $x = 0$.[6]

The (natural) exponential function $f(x) = e^x$ is the unique function which is equal to its own derivative, with the initial value $f(0) = 1$ (and hence one may define e as $f(1)$). The natural logarithm, or logarithm to base e, is the inverse function to the natural exponential function. The natural logarithm of a number $k > 1$ can be defined directly as the area under the curve $y = 1/x$ between $x = 1$ and $x = k$, in which case e is the value of k for which this area equals one (see image). There are various other characterizations.

1. The number e is the unique positive real number such that $\frac{d}{dt}e^t = e^t$.
2. The number e is the unique positive real number such that $\frac{d}{dt}\log_e t = \frac{1}{t}$.

The following four characterizations can be proven to be equivalent:

3. The number e is the limit
   $$e = \lim_{n\to\infty}\left(1 + \frac{1}{n}\right)^n$$
   Similarly:
   $$e = \lim_{t\to0}(1+t)^{\frac{1}{t}}$$
4. The number e is the sum of the infinite series
   $$e = \sum_{n=0}^{\infty}\frac{1}{n!} = \frac{1}{0!} + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} + \cdots,$$
   where $n!$ is the factorial of n. (By convention $0! = 1$.)

$e^{\ln 2} = 2, \quad 2^x = \left(e^{\ln 2}\right)^x = e^{x\ln 2}$
$3^x = e^{x\ln 3}$
$a^x = e^{cx}$

$\log_3(2^3) = 3$
$2^{\log_2 8} = 8$
$2^{\log_2 x} = x$

std::exp, std::expf, std::expl

Defined in header <cmath>

float	exp (float arg);	(1)	
float	expf(float arg);		(since C++11)
double	exp (double arg);	(2)	
long double	exp (long double arg);	(3)	
long double	expl(long double arg);		(since C++11)
double	exp (IntegralType arg);	(4)	(since C++11)

Computes e (Euler's number, 2.7182818...) raised to the given power arg

Logarithm

$\log_2 x = lg\,x, \quad binary\log$
$\log_{10} x = \log x, \quad base\,10\log$
$\log_e x = \ln x, \quad natural\log$

std::log2, std::log2f, std::log2l

Defined in header <cmath>

float	log2 (float arg);	(1)	(since C++11)
float	log2f(float arg);		
double	log2 (double arg);	(2)	(since C++11)
long double	log2 (long double arg);	(3)	(since C++11)
long double	log2l(long double arg);		
double	log2 (IntegralType arg);	(4)	(since C++11)

1-3) Computes the binary (base-2) logarithm of arg.

std::log10, std::log10f, std::log10l

Defined in header <cmath>

float	log10 (float arg);	(1)	
float	log10f(float arg);		(since C++11)
double	log10 (double arg);	(2)	
long double	log10 (long double arg);	(3)	
long double	log10l(long double arg);		(since C++11)
double	log10 (IntegralType arg);	(4)	(since C++11)

std::log, std::logf, std::logl

Defined in header <cmath>

float	log (float arg);	(1)	
float	logf(float arg);		(since C++11)
double	log (double arg);	(2)	
long double	log (long double arg);	(3)	
long double	logl(long double arg);		(since C++11)
double	log (IntegralType arg);	(4)	(since C++11)

1-3) Computes the natural (base e) logarithm of arg.