Mathematical functions

Trigonometric functions

`sin`(x, /[, out, where, casting, order, ...])	Trigonometric sine, element-wise.
`cos`(x, /[, out, where, casting, order, ...])	Cosine element-wise.
`tan`(x, /[, out, where, casting, order, ...])	Compute tangent element-wise.
`arcsin`(x, /[, out, where, casting, order, ...])	Inverse sine, element-wise.
`asin`(x, /[, out, where, casting, order, ...])	Inverse sine, element-wise.
`arccos`(x, /[, out, where, casting, order, ...])	Trigonometric inverse cosine, element-wise.
`acos`(x, /[, out, where, casting, order, ...])	Trigonometric inverse cosine, element-wise.
`arctan`(x, /[, out, where, casting, order, ...])	Trigonometric inverse tangent, element-wise.
`atan`(x, /[, out, where, casting, order, ...])	Trigonometric inverse tangent, element-wise.
`hypot`(x1, x2, /[, out, where, casting, ...])	Given the "legs" of a right triangle, return its hypotenuse.
`arctan2`(x1, x2, /[, out, where, casting, ...])	Element-wise arc tangent of `x1/x2` choosing the quadrant correctly.
`atan2`(x1, x2, /[, out, where, casting, ...])	Element-wise arc tangent of `x1/x2` choosing the quadrant correctly.
`degrees`(x, /[, out, where, casting, order, ...])	Convert angles from radians to degrees.
`radians`(x, /[, out, where, casting, order, ...])	Convert angles from degrees to radians.
`unwrap`(p[, discont, axis, period])	Unwrap by taking the complement of large deltas with respect to the period.
`deg2rad`(x, /[, out, where, casting, order, ...])	Convert angles from degrees to radians.

`rad2deg`(x, /[, out, where, casting, order, ...])	Convert angles from radians to degrees.

Hyperbolic functions

`sinh`(x, /[, out, where, casting, order, ...])	Hyperbolic sine, element-wise.
`cosh`(x, /[, out, where, casting, order, ...])	Hyperbolic cosine, element-wise.
`tanh`(x, /[, out, where, casting, order, ...])	Compute hyperbolic tangent element-wise.
`arcsinh`(x, /[, out, where, casting, order, ...])	Inverse hyperbolic sine element-wise.
`asinh`(x, /[, out, where, casting, order, ...])	Inverse hyperbolic sine element-wise.
`arccosh`(x, /[, out, where, casting, order, ...])	Inverse hyperbolic cosine, element-wise.
`acosh`(x, /[, out, where, casting, order, ...])	Inverse hyperbolic cosine, element-wise.
`arctanh`(x, /[, out, where, casting, order, ...])	Inverse hyperbolic tangent element-wise.
`atanh`(x, /[, out, where, casting, order, ...])	Inverse hyperbolic tangent element-wise.

Rounding

`round`(a[, decimals, out])	Evenly round to the given number of decimals.
`around`(a[, decimals, out])	Round an array to the given number of decimals.
`rint`(x, /[, out, where, casting, order, ...])	Round elements of the array to the nearest integer.
`fix`(x[, out])	Round to nearest integer towards zero.
`floor`(x, /[, out, where, casting, order, ...])	Return the floor of the input, element-wise.
`ceil`(x, /[, out, where, casting, order, ...])	Return the ceiling of the input, element-wise.

`trunc`(x, /[, out, where, casting, order, ...])	Return the truncated value of the input, element-wise.

Sums, products, differences

`prod`(a[, axis, dtype, out, keepdims, ...])	Return the product of array elements over a given axis.
`sum`(a[, axis, dtype, out, keepdims, ...])	Sum of array elements over a given axis.
`nanprod`(a[, axis, dtype, out, keepdims, ...])	Return the product of array elements over a given axis treating Not a Numbers (NaNs) as ones.
`nansum`(a[, axis, dtype, out, keepdims, ...])	Return the sum of array elements over a given axis treating Not a Numbers (NaNs) as zero.
`cumulative_sum`(x, /, *[, axis, dtype, out, ...])	Return the cumulative sum of the elements along a given axis.
`cumulative_prod`(x, /, *[, axis, dtype, out, ...])	Return the cumulative product of elements along a given axis.
`cumprod`(a[, axis, dtype, out])	Return the cumulative product of elements along a given axis.
`cumsum`(a[, axis, dtype, out])	Return the cumulative sum of the elements along a given axis.
`nancumprod`(a[, axis, dtype, out])	Return the cumulative product of array elements over a given axis treating Not a Numbers (NaNs) as one.
`nancumsum`(a[, axis, dtype, out])	Return the cumulative sum of array elements over a given axis treating Not a Numbers (NaNs) as zero.

`diff`(a[, n, axis, prepend, append])	Calculate the n-th discrete difference along the given axis.
`ediff1d`(ary[, to_end, to_begin])	The differences between consecutive elements of an array.
`gradient`(f, *varargs[, axis, edge_order])	Return the gradient of an N-dimensional array.
`cross`(a, b[, axisa, axisb, axisc, axis])	Return the cross product of two (arrays of) vectors.
`trapezoid`(y[, x, dx, axis])	Integrate along the given axis using the composite trapezoidal rule.

Exponents and logarithms

`exp`(x, /[, out, where, casting, order, ...])	Calculate the exponential of all elements in the input array.
`expm1`(x, /[, out, where, casting, order, ...])	Calculate `exp(x) - 1` for all elements in the array.
`exp2`(x, /[, out, where, casting, order, ...])	Calculate *2**p* for all *p* in the input array.
`log`(x, /[, out, where, casting, order, ...])	Natural logarithm, element-wise.
`log10`(x, /[, out, where, casting, order, ...])	Return the base 10 logarithm of the input array, element-wise.
`log2`(x, /[, out, where, casting, order, ...])	Base-2 logarithm of *x*.
`log1p`(x, /[, out, where, casting, order, ...])	Return the natural logarithm of one plus the input array, element-wise.
`logaddexp`(x1, x2, /[, out, where, casting, ...])	Logarithm of the sum of exponentiations of the inputs.

`logaddexp2`(x1, x2, /[, out, where, casting, ...])	Logarithm of the sum of exponentiations of the inputs in base-2.

Other special functions

`i0`(x)	Modified Bessel function of the first kind, order 0.
`sinc`(x)	Return the normalized sinc function.

Floating point routines

`signbit`(x, /[, out, where, casting, order, ...])	Returns element-wise True where signbit is set (less than zero).
`copysign`(x1, x2, /[, out, where, casting, ...])	Change the sign of x1 to that of x2, element-wise.
`frexp`(x[, out1, out2], / [[, out, where, ...])	Decompose the elements of x into mantissa and twos exponent.
`ldexp`(x1, x2, /[, out, where, casting, ...])	Returns x1 * 2**x2, element-wise.
`nextafter`(x1, x2, /[, out, where, casting, ...])	Return the next floating-point value after x1 towards x2, element-wise.
`spacing`(x, /[, out, where, casting, order, ...])	Return the distance between x and the nearest adjacent number.

Rational routines

`lcm`(x1, x2, /[, out, where, casting, order, ...])	Returns the lowest common multiple of `|x1|` and `|x2|`

`gcd`(x1, x2, /[, out, where, casting, order, ...])	Returns the greatest common divisor of `|x1|` and `|x2|`

Arithmetic operations

`add`(x1, x2, /[, out, where, casting, order, ...])	Add arguments element-wise.
`reciprocal`(x, /[, out, where, casting, ...])	Return the reciprocal of the argument, element-wise.
`positive`(x, /[, out, where, casting, order, ...])	Numerical positive, element-wise.
`negative`(x, /[, out, where, casting, order, ...])	Numerical negative, element-wise.
`multiply`(x1, x2, /[, out, where, casting, ...])	Multiply arguments element-wise.
`divide`(x1, x2, /[, out, where, casting, ...])	Divide arguments element-wise.
`power`(x1, x2, /[, out, where, casting, ...])	First array elements raised to powers from second array, element-wise.
`pow`(x1, x2, /[, out, where, casting, order, ...])	First array elements raised to powers from second array, element-wise.
`subtract`(x1, x2, /[, out, where, casting, ...])	Subtract arguments, element-wise.
`true_divide`(x1, x2, /[, out, where, ...])	Divide arguments element-wise.
`floor_divide`(x1, x2, /[, out, where, ...])	Return the largest integer smaller or equal to the division of the inputs.
`float_power`(x1, x2, /[, out, where, ...])	First array elements raised to powers from second array, element-wise.
`fmod`(x1, x2, /[, out, where, casting, ...])	Returns the element-wise remainder of division.
`mod`(x1, x2, /[, out, where, casting, order, ...])	Returns the element-wise remainder of division.

`modf`(x[, out1, out2], / [[, out, where, ...])	Return the fractional and integral parts of an array, element-wise.
`remainder`(x1, x2, /[, out, where, casting, ...])	Returns the element-wise remainder of division.
`divmod`(x1, x2[, out1, out2], / [[, out, ...])	Return element-wise quotient and remainder simultaneously.

Handling complex numbers

`angle`(z[, deg])	Return the angle of the complex argument.
`real`(val)	Return the real part of the complex argument.
`imag`(val)	Return the imaginary part of the complex argument.
`conj`(x, /[, out, where, casting, order, ...])	Return the complex conjugate, element-wise.
`conjugate`(x, /[, out, where, casting, ...])	Return the complex conjugate, element-wise.

Extrema finding

`maximum`(x1, x2, /[, out, where, casting, ...])	Element-wise maximum of array elements.
`max`(a[, axis, out, keepdims, initial, where])	Return the maximum of an array or maximum along an axis.
`amax`(a[, axis, out, keepdims, initial, where])	Return the maximum of an array or maximum along an axis.
`fmax`(x1, x2, /[, out, where, casting, ...])	Element-wise maximum of array elements.

`nanmax`(a[, axis, out, keepdims, initial, where])	Return the maximum of an array or maximum along an axis, ignoring any NaNs.
`minimum`(x1, x2, /[, out, where, casting, ...])	Element-wise minimum of array elements.
`min`(a[, axis, out, keepdims, initial, where])	Return the minimum of an array or minimum along an axis.
`amin`(a[, axis, out, keepdims, initial, where])	Return the minimum of an array or minimum along an axis.
`fmin`(x1, x2, /[, out, where, casting, ...])	Element-wise minimum of array elements.
`nanmin`(a[, axis, out, keepdims, initial, where])	Return minimum of an array or minimum along an axis, ignoring any NaNs.

Miscellaneous

`convolve`(a, v[, mode])	Returns the discrete, linear convolution of two one-dimensional sequences.
`clip`(a[, a_min, a_max, out, min, max])	Clip (limit) the values in an array.
`sqrt`(x, /[, out, where, casting, order, ...])	Return the non-negative square-root of an array, element-wise.
`cbrt`(x, /[, out, where, casting, order, ...])	Return the cube-root of an array, element-wise.
`square`(x, /[, out, where, casting, order, ...])	Return the element-wise square of the input.
`absolute`(x, /[, out, where, casting, order, ...])	Calculate the absolute value element-wise.
`fabs`(x, /[, out, where, casting, order, ...])	Compute the absolute values element-wise.
`sign`(x, /[, out, where, casting, order, ...])	Returns an element-wise indication of the sign of a number.
`heaviside`(x1, x2, /[, out, where, casting, ...])	Compute the Heaviside step function.

`nan_to_num`(x[, copy, nan, posinf, neginf])	Replace NaN with zero and infinity with large finite numbers (default behaviour) or with the numbers defined by the user using the `nan`, *posinf* and/or *neginf* keywords.
`real_if_close`(a[, tol])	If input is complex with all imaginary parts close to zero, return real parts.