

 Search the docs ...

[Array objects](#)

[Array API Standard](#)

[Compatibility](#)

[Constants](#)

[Universal functions \(**ufunc**\)](#)

[Routines](#) ^

[Array creation routines](#)

[Array manipulation routines](#)

[Binary operations](#)

[String operations](#)

[C-Types foreign function interface \(**numpy.ctypeslib**\)](#)

[Datetime support functions](#)

[Data type routines](#)

[Mathematical functions with automatic domain](#)

[Floating point error handling](#)

[Discrete Fourier Transform \(**numpy.fft**\)](#)

[Functional programming](#)

[NumPy-specific help functions](#)

[Input and output](#)

[Linear algebra \(**numpy.linalg**\)](#) ^

[numpy.dot](#)

[numpy.linalg.multi_dot](#)

[numpy.vdot](#)

[numpy.inner](#)

[numpy.outer](#)

[numpy.matmul](#)

[numpy.tensordot](#)

[numpy.einsum](#)

[numpy.einsum_path](#)

[numpy.linalg.matrix_power](#)

[numpy.kron](#)

[numpy.linalg.cholesky](#)

[numpy.linalg.qr](#)

[numpy.linalg.svd](#)

[numpy.linalg.eig](#)

[numpy.linalg.eigh](#)

[numpy.linalg.eigvals](#)

[numpy.linalg.eigvalsh](#)

[numpy.linalg.norm](#)

numpy.linalg.inv

[\[source\]](#)

linalg.inv(a)

Compute the (multiplicative) inverse of a matrix.

Given a square matrix *a*, return the matrix *ainv* satisfying `dot(a, ainv) = dot(ainv, a) = eye(a.shape[0])`.

Parameters: *a* : (*..., M, M*) *array_like*
Matrix to be inverted.

Returns: *ainv* : (*..., M, M*) *ndarray or matrix*
(Multiplicative) inverse of the matrix *a*.

Raises: *LinAlgError*
If *a* is not square or inversion fails.

 See also

[scipy.linalg.inv](#)
Similar function in SciPy.

Notes

 *New in version 1.8.0.*

Broadcasting rules apply, see the [numpy.linalg](#) documentation for details.

Examples

```
>>> from numpy.linalg import inv
>>> a = np.array([[1., 2.], [3., 4.]])
>>> ainv = inv(a)
>>> np.allclose(np.dot(a, ainv), np.eye(2))
True
>>> np.allclose(np.dot(ainv, a), np.eye(2))
True
```

If *a* is a matrix object, then the return value is a matrix as well:

```
>>> ainv = inv(np.matrix(a))
>>> ainv
matrix([[ -2. ,  1. ],
        [ 1.5, -0.5]])
```

Inverses of several matrices can be computed at once:

```
>>> a = np.array([[[1., 2.], [3., 4.]], [[1, 3], [3, 5]]])
>>> inv(a)
array([[[-2. ,  1. ],
        [ 1.5, -0.5 ]],
       [[-1.25,  0.75],
        [ 0.75, -0.25]]])
```

© Copyright 2008-2022, NumPy Developers.
Created using [Sphinx](#) 5.3.0.