Search the docs ...

scipy
scipy.cluster
scipy.constants
scipy.datasets
scipy.fft
scipy.fftpack
scipy.integrate
scipy.interpolate
scipy.io
scipy.linalg
scipy.misc
scipy.ndimage
scipy.odr
scipy.optimize
scipy.signal
scipy.sparse
scipy.spatial
scipy.special
scipy.stats

scipy.fftpack.dct

`scipy.fftpack.dct(x, type=2, n=None, axis=-1, norm=None, overwrite_x=False)` [source]

Return the Discrete Cosine Transform of arbitrary type sequence x.

Parameters: **x** : ***array_like***

The input array.

type : ***{1, 2, 3, 4}, optional***

Type of the DCT (see Notes). Default type is 2.

n : ***int, optional***

Length of the transform. If `n < x.shape[axis]`, *x* is truncated. If `n > x.shape[axis]`, *x* is zero-padded. The default results in `n = x.shape[axis]`.

axis : ***int, optional***

Axis along which the dct is computed; the default is over the last axis (i.e., `axis=-1`).

norm : ***{None, 'ortho'}, optional***

Normalization mode (see Notes). Default is None.

overwrite_x : ***bool, optional***

If True, the contents of *x* can be destroyed; the default is False.

Returns: **y** : ***ndarray of real***

The transformed input array.

See also

`idct`
Inverse DCT

Notes

For a single dimension array `x`, `dct(x, norm='ortho')` is equal to MATLAB `dct(x)`.

There are, theoretically, 8 types of the DCT, only the first 4 types are implemented in scipy. 'The' DCT generally refers to DCT type 2, and 'the' Inverse DCT generally refers to DCT type 3.

Type I

There are several definitions of the DCT-I; we use the following (for `norm=None`)

$$y_k = x_0 + (-1)^k x_{N-1} + 2 \sum_{n=1}^{N-2} x_n \cos\left(\frac{\pi k n}{N-1}\right)$$

If `norm='ortho'`, `x[0]` and `x[N-1]` are multiplied by a scaling factor of $\sqrt{2}$, and `y[k]` is multiplied by a scaling factor `f`

$$f = \begin{cases} \frac{1}{2}\sqrt{\frac{1}{N-1}} & \text{if } k = 0 \text{ or } N-1, \\ \frac{1}{2}\sqrt{\frac{2}{N-1}} & \text{otherwise} \end{cases}$$

New in version 1.2.0: Orthonormalization in DCT-I.

Note

The DCT-I is only supported for input size > 1.

Type II

There are several definitions of the DCT-II; we use the following (for `norm=None`)

$$y_k = 2 \sum_{n=0}^{N-1} x_n \cos\left(\frac{\pi k(2n+1)}{2N}\right)$$

If `norm='ortho'`, `y[k]` is multiplied by a scaling factor `f`

$$f = \begin{cases} \sqrt{\frac{1}{4N}} & \text{if } k = 0, \\ \sqrt{\frac{1}{2N}} & \text{otherwise} \end{cases}$$

which makes the corresponding matrix of coefficients orthonormal (`O @ O.T = np.eye(N)`).

Type III

There are several definitions, we use the following (for `norm=None`)

$$y_k = x_0 + 2 \sum_{n=1}^{N-1} x_n \cos\left(\frac{\pi(2k+1)n}{2N}\right)$$

or, for `norm='ortho'`

$$y_k = \frac{x_0}{\sqrt{N}} + \sqrt{\frac{2}{N}} \sum_{n=1}^{N-1} x_n \cos\left(\frac{\pi(2k+1)n}{2N}\right)$$

The (unnormalized) DCT-III is the inverse of the (unnormalized) DCT-II, up to a factor *2N*. The orthonormalized DCT-III is exactly the inverse of the orthonormalized DCT-II.

Type IV

There are several definitions of the DCT-IV; we use the following (for `norm=None`)

$$y_k = 2 \sum_{n=0}^{N-1} x_n \cos\left(\frac{\pi(2k+1)(2n+1)}{4N}\right)$$

If `norm='ortho'`, `y[k]` is multiplied by a scaling factor `f`

$$f = \frac{1}{\sqrt{2N}}$$

> ***New in version 1.2.0:*** Support for DCT-IV.

References

[1] ‘A Fast Cosine Transform in One and Two Dimensions’, by J. Makhoul, *IEEE Transactions on acoustics, speech and signal processing* vol. 28(1), pp. 27-34, DOI:10.1109/TASSP.1980.1163351 (1980).

[2] Wikipedia, “Discrete cosine transform”, https://en.wikipedia.org/wiki/Discrete_cosine_transform

Examples

The Type 1 DCT is equivalent to the FFT (though faster) for real, even-symmetrical inputs. The output is also real and even-symmetrical. Half of the FFT input is used to generate half of the FFT output:

```
>>> from scipy.fftpack import fft, dct
>>> import numpy as np
>>> fft(np.array([4., 3., 5., 10., 5., 3.])).real
array([ 30.,  -8.,   6.,  -2.,   6.,  -8.])
>>> dct(np.array([4., 3., 5., 10.]), 1)
array([ 30.,  -8.,   6.,  -2.])
```

Previous scipy.fftpack.irfft
Next scipy.fftpack.idct