scipy.linalg.solve_banded

scipy.linalg.solve_banded(*l_and_u*, *ab*, *b*, *overwrite_ab=False*, *overwrite_b=False*, *debug=None*, *check_finite=True***)** **[source] (https://github.com/scipy/scipy/blob/v1.5.4/scipy/linalg/basic.py#L361-L471)**

Solve the equation a x = b for x, assuming a is banded matrix.

The matrix a is stored in *ab* using the matrix diagonal ordered form:

```
ab[u + i - j, j] == a[i,j]
```

Example of *ab* (shape of a is (6,6), *u* =1, *l* =2):

```
*    a01  a12  a23  a34  a45
a00  a11  a22  a33  a44  a55
a10  a21  a32  a43  a54   *
a20  a31  a42  a53   *    *
```

Parameters: **(l, u) :** ***(integer, integer)***

Number of non-zero lower and upper diagonals

ab : ***(l + u + 1, M) array_like***

Banded matrix

b : ***(M,) or (M, K) array_like***

Right-hand side

overwrite_ab : ***bool, optional***

Discard data in *ab* (may enhance performance)

overwrite_b : ***bool, optional***

Discard data in *b* (may enhance performance)

check_finite : ***bool, optional***

Whether to check that the input matrices contain only finite numbers. Disabling may give a performance gain, but may result in problems (crashes, non-termination) if the inputs do contain infinities or NaNs.

Returns: **x :** ***(M,) or (M, K) ndarray***

The solution to the system a x = b. Returned shape depends on the shape of *b*.

Examples

Solve the banded system a x = b, where:

```
    [5  2 -1  0  0]       [0]
    [1  4  2 -1  0]       [1]
a = [0  1  3  2 -1]   b = [2]
    [0  0  1  2  2]       [2]
    [0  0  0  1  1]       [3]
```

There is one nonzero diagonal below the main diagonal (l = 1), and two above (u = 2). The diagonal banded form of the matrix is:

```
     [*  * -1 -1 -1]
ab = [*  2  2  2  2]
     [5  4  3  2  1]
     [1  1  1  1  *]
```

>>>

```
>>> from scipy.linalg import solve_banded
>>> ab = np.array([[0,  0, -1, -1, -1],
...                [0,  2,  2,  2,  2],
...                [5,  4,  3,  2,  1],
...                [1,  1,  1,  1,  0]])
>>> b = np.array([0, 1, 2, 2, 3])
>>> x = solve_banded((1, 2), ab, b)
>>> x
array([-2.37288136,  3.93220339, -4.        ,  4.3559322 , -1.3559322 ])
```

Previous topic

scipy.linalg.solve (scipy.linalg.solve.html)

Next topic

scipy.linalg.solveh_banded (scipy.linalg.solveh_banded.html)

Quick search

search