(../../index.html)

Scipy.org (http://scipy.org/)   Docs (http://docs.scipy.org/)

NumPy v1.10 Manual (../../index.html)   NumPy Reference (../index.html)

Routines (../routines.html)   Input and output (../routines.io.html)

index (../../genindex.html)   next (numpy.genfromtxt.html)   previous (numpy.loadtxt.html)

numpy.savetxt

numpy.savetxt(fname, X, fmt='%.18e', delimiter=' ', newline='\n', header='', footer='', comments='# ') (http://github.com   [source] /numpy/numpy/blob/v1.10.1/numpy/lib/npyio.py#L967-L1166)

Save an array to a text file.

ndarray

Previous topic

numpy.loadtxt (numpy.loadtxt.htm

Next topic

numpy.genfromtxt (numpy.genfromtxt.

Parameters: **fname** : filename or file handle

> If the filename ends in `.gz`, the file is automatically saved in compressed gzip format. loadtxt (numpy.loadtxt.html#numpy.loadtxt) understands gzipped files transparently.

X : array_like

> Data to be saved to a text file.

fmt : str or sequence of strs, optional

> A single format (%10.5f), a sequence of formats, or a multi-format string, e.g. 'Iteration %d – %10.5f', in which case delimiter is ignored. For complex X, the legal options for fmt are:
>
> > a. a single specifier, fmt='%.4e', resulting in numbers formatted
> >
> > > like ' (%s+%sj)' % (fmt, fmt)
> >
> > b. a full string specifying every real and imaginary part, e.g.
> >
> > > ' %.4e %+.4j %.4e %+.4j %.4e %+.4j' for 3 columns
> >
> > c. a list of specifiers, one per column - in this case, the real
> >
> > > and imaginary part must have separate specifiers, e.g. ['%.3e + %.3ej', '(%.15e%+.15ej)'] for 2 columns

delimiter : str, optional

> String or character separating columns.

newline : str, optional

> String or character separating lines.
> New in version 1.5.0.

header : str, optional

> String that will be written at the beginning of the file.
> New in version 1.7.0.

footer : str, optional

> String that will be written at the end of the file.
> New in version 1.7.0.

comments : str, optional

> String that will be prepended to the `header` and `footer` strings, to mark them as comments. Default: '# ', as expected by e.g. `numpy.loadtxt`.
> New in version 1.7.0.

See also:

save (numpy.save.html#numpy.save)   Save an array to a
> binary file in NumPy `.npy` format

savez (numpy.savez.html#numpy.savez)   Save several
> arrays into an uncompressed `.npz` archive

savez_compressed
(numpy.savez_compressed.html#numpy.savez_compressed)
Save several arrays into a compressed `.npz` archive

Notes

Further explanation of the fmt parameter
(`%[flag]width[.precision]specifier`):

flags:
> `-` : left justify
>
> `+` : Forces to precede result with + or -.
>
> `0` : Left pad the number with zeros instead of space (see width).

width:
> Minimum number of characters to be printed. The value is not truncated if it has more characters.

precision:
> - For integer specifiers (eg. `d`, `i`, `o`, `x`), the minimum number of digits.
> - For `e`, `E` and `f` specifiers, the number of digits to print after the decimal point.
> - For `g` and `G`, the maximum number of significant digits.
> - For `s`, the maximum number of characters.

specifiers:
> `c` : character
>
> `d` or `i` : signed decimal integer
>
> `e` or `E` : scientific notation with `e` or `E`.
>
> `f` : decimal floating point
>
> `g`, `G` : use the shorter of `e`, `E` or `f`
>
> `o` : signed octal
>
> `s` : string of characters
>
> `u` : unsigned decimal integer
>
> `x`, `X` : unsigned hexadecimal integer

This explanation of `fmt` is not complete, for an exhaustive specification see [R280].

References

[R280]   (1, 2) Format Specification Mini-Language (http://docs.python.org/library/string.html#format-specification-mini-language), Python Documentation.

Examples

>>>

```
>>> x = y = z = np.arange(0.0, 5.0, 1.0)
>>> np.savetxt('test.out', x, delimiter=',')    # X is an array
>>> np.savetxt('test.out', (x, y, z))    # x, y, z equal sized 1D a
rrays
>>> np.savetxt('test.out', x, fmt='%1.4e')    # use exponential
 notation
```

.out

```
>>> type(x)
<type 'numpy.ndarray'>
```

>>> x = y = z = np.arange(0.0, 5.0, 1.0)
>>> np.savetxt('test.out', x, delimiter=',')   # X is an array
>>> np.savetxt('test.out', (x, y, z))   # x, y, z equal sized 1D a
rrays
>>> np.savetxt('test.out', x, fmt='%1.4e')   # use exponential
 notation