# netcdf (for reading in nc)

This document describes how to use the Python netCDF4 library to create, open, modify, and write NetCDF (Network Common Data Form) files. This file type is common in satellite datasets because of its versatility, its ability to be used with parallel I/O, and because it is a self-describing format. A dataset’s metadata can be stored in the same file as the data itself, reducing errors in interpretation. NetCDF filenames usually bear the .nc suffix.

A NetCDF file contains three basic parts: the data itself (stored in Variables), the size of data (stored as Dimensions), and information about the data (aka metadata, stored as Attributes). Here we will talk about how to read, edit, and write NetCDF files.

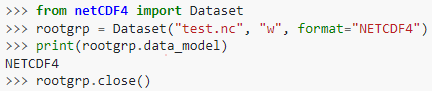
See the [netCDF4 API documentation (unidata.github.io)](https://unidata.github.io/netcdf4-python/#creatingopeningclosing-a-netcdf-file) for a full list of functions/methods in the netCDF4 Python library.

ALL INFORMATION IS FROM THE SOURCE BELOW:

*NETCDF4*. netCDF4 API documentation. (n.d.). Retrieved May 17, 2022, from https://unidata.github.io/netcdf4-python/#creatingopeningclosing-a-netcdf-file

## Creating/opening netCDF

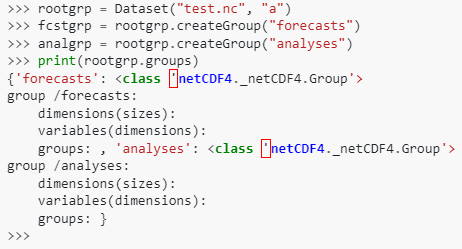
To create a NetCDF file from Python, you simply call the [Dataset](https://unidata.github.io/netcdf4-python/#Dataset) constructor. This is also the method used to open an existing NetCDF file. If the file is open for write access (mode='w', 'r+' or 'a'), you may write any type of data including new dimensions, groups, variables and attributes. If you only need to read data, an existing NetCDF file can be opened with read access (mode=’r’). Do not forget to close opened files when you are finished with them!



Note that by default, writing a file with a name that already exists will overwrite (“clobber”) the existing file. Be careful when naming your files not to accidentally overwrite things you want to save. You can also include an optional argument (clobber=False) to change this behaviour.

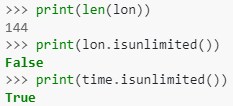
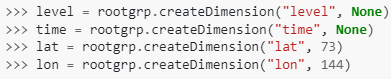
## Groups

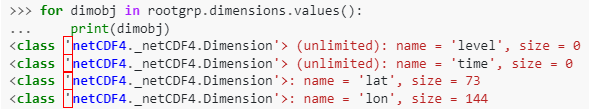
The NetCDF format allows data to be organized within the file, using Groups. To create [Group](https://unidata.github.io/netcdf4-python/#Group) instances, use the [Dataset.createGroup](https://unidata.github.io/netcdf4-python/#Dataset.createGroup) method of a [Dataset](https://unidata.github.io/netcdf4-python/#Dataset) or [Group](https://unidata.github.io/netcdf4-python/#Group) instance (note that you can nest a Group within a Group!). [Dataset.createGroup](https://unidata.github.io/netcdf4-python/#Dataset.createGroup) takes a single argument, a Python string containing the name of the new Group. The new [Group](https://unidata.github.io/netcdf4-python/#Group) instances contained within the root group can be accessed by name using the Groups dictionary attribute of the [Dataset](https://unidata.github.io/netcdf4-python/#Dataset) instance.



## Dimensions

Dimensions refer to the size of the data arrays that are stored in the file. A single Variable can have several Dimensions; also, a single Dimension can be used by multiple Variables. A Dimension is created using the [Dataset.createDimension](https://unidata.github.io/netcdf4-python/#Dataset.createDimension) method of a [Dataset](https://unidata.github.io/netcdf4-python/#Dataset) or [Group](https://unidata.github.io/netcdf4-python/#Group) instance. A Python string is used to set the name of the Dimension, and an integer value is used to set the size. To create an unlimited Dimension (a Dimension that can be appended to), the size value is set to None or 0.





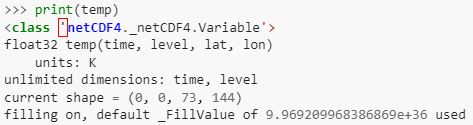
When we create a new NetCDF file, the Dimensions must be created before any Variables. As we will see in the next section, the Variable constructor relies on a list of Dimensions to reserve space in memory.

## Variables

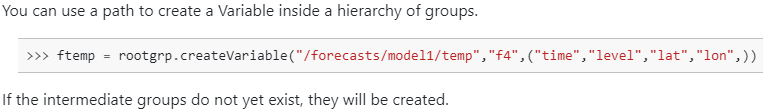
In a NetCDF file, data is stored within Variables. These Variables can be thought of as multi-dimensional arrays, and a single Variable contains data of only one type. A Group or Dataset may contain multiple Variables that store different data types (i.e., you could have a string Variable and an integer Variable in the same file). To create a NetCDF Variable within Python, use the [Dataset.createVariable](https://unidata.github.io/netcdf4-python/#Dataset.createVariable) method of a [Dataset](https://unidata.github.io/netcdf4-python/#Dataset) or [Group](https://unidata.github.io/netcdf4-python/#Group) instance. The [Dataset.createVariable](https://unidata.github.io/netcdf4-python/#Dataset.createVariable) method has two mandatory arguments, the Variable name (a Python string), and the Variable datatype. The Variable's dimensions are given by a tuple containing the Dimension names (defined previously with [Dataset.createDimension](https://unidata.github.io/netcdf4-python/#Dataset.createDimension)). To create a scalar Variable, simply leave out the Dimensions keyword.

Valid datatype specifiers include:

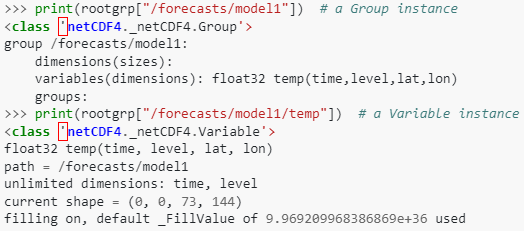
* 'f4' (32-bit floating point),
* 'f8' (64-bit floating point),
* 'i4' (32-bit signed integer),
* 'i2' (16-bit signed integer),
* 'i8' (64-bit signed integer),
* 'i1' (8-bit signed integer),
* 'u1' (8-bit unsigned integer),
* 'u2' (16-bit unsigned integer),
* 'u4' (32-bit unsigned integer),
* 'u8' (64-bit unsigned integer), or
* 'S1' (single-character string).



You can specify a path to create a Variable inside of a hierarchy of Groups:



If the intermediate Groups do not yet exist, they will be created.



## Attributes

Attributes are to data what comments are to good code. They contain information describing the dataset, things like the units on a particular variable or where, when, and by whom the dataset was created. Having good metadata stored in the files themselves saves users time in going back to refer to a manual and cuts down on misinterpretation.

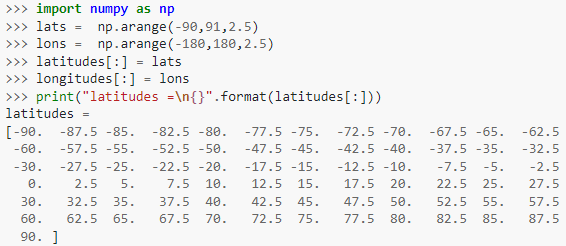
In a NetCDF file, we have global Attributes that describe the entire Dataset or an entire Group. Variable Attributes are specific to a particular Variable. We assign global Attributes by assigning values directly to Dataset or Group instances. Variable Attributes are assigned to instances of Variables. Attributes can be strings, numbers, or sequences.

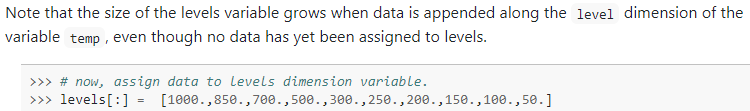
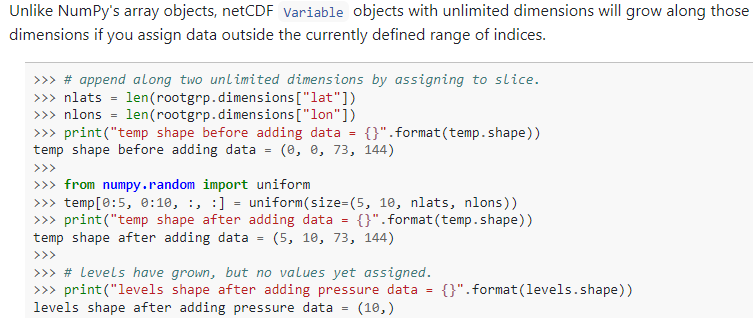


Attributes can be deleted from a Dataset, Group, or Variable instance using the Python del statement (i.e. del grp.foo removes the attribute foo group the group grp).

## Writing data

Writing data from the Python environment into a NetCDF file involves assigning an array of the same size as the Dimensions of a given NetCDF Variable to that Variable. Variables with unlimited Dimensions in one or more directions behave a bit differently, as described below.

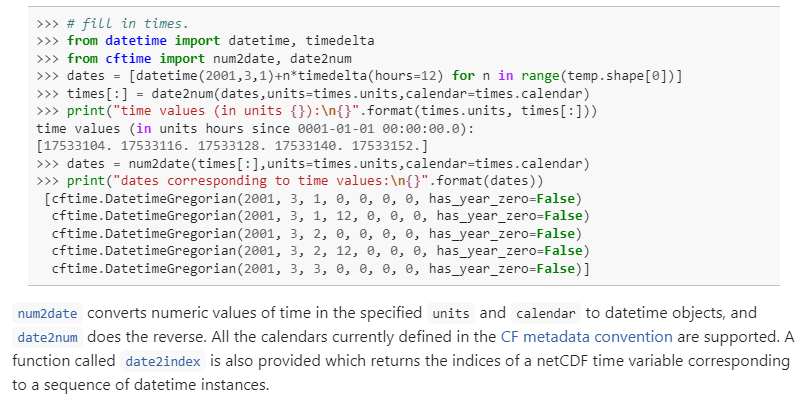




While the actual copying of data into a NetCDF file looks easy here (temp = uniform), recall that in order to get to this point, we first had to create the hierarchy (Datasets, Groups, Dimensions, then Variables and Attributes) into which the data would be stored.

## Time coordinates

It is worthwhile to mention a few things about how times are stored because this comes up often in geospatial datasets. Python has specific libraries dealing with dates and times internally, as shown below.



Very often times are specified as time elapsed from a given date and time (sometimes referred to as an “epoch”). In the example above, the units on the time Variable (stored as an Attribute) are “hours since 0001-01-01 00:00:00.0”. Numerous standard epochs exist so check the metadata for specifics. Unless otherwise specified, times for satellite data are given in UTC (Coordinated Universal Time).