Environmental Modelling & Software, **23**(10), 1338–1342, doi:10.1016/j.envsoft.2008.03.004.

Updated: Wed 17th Sept, 2008, 21:49

Analysis of Self-describing Gridded Geoscience Data with netCDF Operators (NCO)

by Charles S. Zender*

Department of Earth System Science — zender@uci.edu

University of California, Irvine — Voice: (949) 824-2987

Irvine, CA 92697-3100, USA — Fax: (949) 824-3874

Abstract

The netCDF Operator (NCO) software facilitates manipulation and analysis of gridded geoscience data stored in the self-describing netCDF format. NCO is optimized to efficiently analyze large multi-dimensional datasets spanning many files. Researchers and data centers often use NCO to analyze and serve observed and modeled geoscience data including satellite observations and weather, air quality, and climate forecasts. NCO's functionality includes shared memory threading, a message-passing interface, network transparency, and an interpreted language parser. NCO treats data files as a high level data type whose contents may be simultaneously manipulated by a single command. Institutions and data portals often use NCO for middleware, to hyperslab and aggregate dataset requests, while scientific researchers use NCO to perform three general functions: arithmetic operations, data permutation and compression, and metadata editing. We describe NCO's design philosophy and primary features, illustrate techniques to solve common geoscience and environmental data analysis problems, and suggest ways to design gridded datasets that can ease their subsequent analysis.

Keywords: geoscience; time-series; data analysis; arithmetic; metadata; model design; netCDF;

*Corresponding author

Software Availability

Name of software: netCDF Operators (NCO)

Developer: Charles S. Zender

Operating system: All

Programming languages: C/C99/C++

Availability and cost: Freely available at http://nco.sf.net

1 Introduction

Gridded geoscience model and sensor datasets present an interesting set of challenges for researchers and the data portals that serve them (*Foster et al.*, 2002). Many geoscience disciplines have transitioned or are transitioning from data-poor and simulation-poor to data-rich and simulation-rich (*NRC*, 2001). A software ecosystem has evolved to help researchers exploit this transition with fast data discovery, aggregation, analysis, and dissemination techniques (e.g., *Domenico et al.*, 2002; *Cornillon et al.*, 2003). In this ecosystem are the netCDF Operators (NCO)—software for manipulation and analysis of gridded geoscience data stored in the self-describing netCDF format. NCO is used in several niches in geoscience data analysis workflow (*Woolf et al.*, 2003), because its functionality is independent of and complementary to data discovery, aggregation, and dissemination.

The netCDF Operators have evolved over the past decade to serve research the needs of individual researchers and data-centers for fast, flexible tools to help manage netCDF-format datasets. The NCO User's Guide (*Zender*, 2008) documents NCO's functionality and calling conventions. *Zender and Mangalam* (2007) describe the core NCO arithmetic algorithms and their theoretical and measured scaling with dataset size and structure. This paper describes NCO's design philosophy and primary features, illustrates techniques to solve common geoscience and environmental data analysis problems, and suggests ways to design gridded datasets that can ease their subsequent analysis.

We will demonstrate the NCO paradigm and features by applying them to frequently occurring geoscience data reduction problems taken from the field of climate data analysis. The reader will see that these problems are generic to disciplines where large gridded datasets are regularly produced and analyzed. Modern weather, climate, and remote sensing research often requires identical analyses of hundreds of variables in thousands of files. Traditional analysis approaches that use low-level, compiled languages and most high level, interpreted languages fail to scale well to this problem space (*Wang et al.*, 2007). Re-coding compiled or interpreted data analysis scripts to act on new variables and new datasets is tedious and non-productive when it requires, for example, manually changing variable names and loop counters even when the underlying analysis (such as averaging) does not change.

NCO helps solve this problem by using the self-describing capability of the netCDF data format (*Rew and Davis*, 1990) and POSIX shells (*Newham and Rosenblatt*, 1998) to define a specific analysis of a generic type without user intervention. This flexibility is important to geoscience researchers who often analyze and inter-compare gridded datasets in an open-ended fashion, creating unique analysis workflows through trial and error. For the same reasons, many data-portals use NCO to fulfill the unpredictable hyperslab requests issued by users on their WWW front-ends, e.g., the NCAR Community Data Portal (CDP; https://cdp.ucar.edu), and the NOAA Climate Diagnostics Center (CDC; http://www.cdc.noaa.gov/PublicData). NCO is middleware in that it processes datasets in netCDF format, generated by models or retrieval procedures, to new netCDF datasets, more suitable for graphical display, dissemination, or numerical analysis.

Geoscience researchers use many toolkits besides NCO to analyze large volumes of gridded data. These include the Climate Data Analysis Tools (CDAT) (*Fiorino and Williams*, 2002), the Climate Data Operators (CDO; http://www.mpimet.mpg.de/fileadmin/software/cdo), the Grid Analysis and Display System (GrADS; http://www.iges.org/grads/grads.html), the Interactive Data Language (IDL; http://www.ittvis.com/idl), MATLAB (http://www.mathworks.com), and the NCAR Command Language (NCL; http://www.ncl.ucar.edu). Of these toolkits, CDO is closest to NCO in that both use command line operators constructed to perform chain-able operations like tradi-

tional UNIX filters. Unlike NCO and CDO, the CDAT, GrADS, IDL, MATLAB, and NCL toolkits support comprehensive integrated visualization capabilities, but their design is not optimized for batch-driven operations on large numbers of files.

2 Design Philosophy

Traditional geoscience data processing works with an intra-file paradigm where users open one or a few files to read and manipulate one or a few variables at a time. The intra-file paradigm works well in cases where all the pertinent data reside in a few files, and the processing of each variable is unique and requires hand-coding. In large geoscience applications data storage requirements may dictate that relevant data be spread over multiple files. Level one satellite data, for example, are often stored in a file-per-day or file-per-orbit format. Data produced by geophysical time-stepping models is usually output every time-step or as a series of time-averages. Climate models usually archive data once per simulated day or month in multi-year or multi-century simulations. NCO supports an inter-file paradigm for situations where the intra-file paradigm is unwieldy.

NCO abides by guidelines that have proven their value when processing large numbers of geophysical datasets:

1. Files behave as an elemental data unit. Unless specifically requested otherwise, NCO applies the same operation to all variables (or attributes) in a file. Manipulating (e.g., adding, subtracting) entire geophysical states as represented by the collection of variables in a file is as easy as manipulating a single variable in a traditional data analysis language. When the "process all variables" paradigm is combined with UNIX filename *globbing* (expanding a file name pattern containing wildcard characters into a set of specific file names), NCO effectively subsumes two common loops (loops over files and over variables) of geoscience and environmental data-analysis into one command.
2. Files processed sequentially are usually homogeneous. NCO assumes the structure of each file (i.e., the fields present and their dimensions) are identical to the structure of the first

file in the sequence. NCO allows the record dimension (usually time) length and number of variables to change between files, but not the ranks of variables.

3. An audit trail that tracks data provenance and processing history is desirable for both the data analyst and their colleagues who receive the processed data. For analysis involving multi-file sequences, the metadata in the first file, along with a list of the other files, adequately preserves the processing history. By convention, NCO keeps this information in the `history` attribute (*Rew et al.*, 2005).
4. There is value in maintaining the distinctions and associations between *dimensions*, *coordinates*, and *variables* (*Rew and Davis*, 1990) during data analysis. Unless otherwise specified, NCO automatically attaches coordinate data (i.e., dimension values) to variables it transfers.
5. Tools should treat data as generically as possible, and impose no software limitations on data dimensionality, size, type, or ordering.

This design philosophy allows users to remain relatively ignorant details of file and variable names, field geometry, and NCO itself.

3 Operators

NCO partially fulfills the netCDF designers' original vision for a follow-on set of generic data operators (*Rew and Davis*, 1990). Presently NCO includes twelve utilities built from a common library (Table 1).

[Table 1 about here.]

Operator names are acronyms for their functionality, prefixed with "nc" to indicate their relationship to netCDF. The twelve operators typically read netCDF files as input, perform some manipulations, then write netCDF files as output. In this sense the operators are filters, or middleware. The NCO User's Guide (*Zender*, 2008) documents the functionality and calling conventions for all operators.

The primary purpose of the arithmetic operators is to alter existing or create new data. The other operators, called metadata operators, manipulate metadata or re-arrange (but do not alter) data. The arithmetic operators can be quite computationally intensive, in contrast to the metadata operators which are mostly I/O-dominated. The amount of data processed varies strongly by operator type. The multi-file operators (MFOs) are the most data-intensive. Often they are applied to entire data-streams.

3.1 Arithmetic Operators

Arithmetic operators (`ncap`, `ncbo`, `ncea`, `ncflint`, `ncra`, and `ncwa`) are distinguished from metadata operators by their use of floating point arithmetic. The arithmetic operators take individual algorithms (e.g., averaging, broadcasting) from a common library and re-combine them for a specific purpose such as averaging a series of files (*Zender and Mangalam*, 2007). The exception is `ncap`, an interpreted language processor that computes derived fields from algebraic scripts containing standard functions (e.g., `sin, cos, pow`) of arbitrary complexity.

3.2 Metadata Operators

Metadata operators (`ncatted`, `ncecat`, `ncks`, `ncrcat`, `ncpdq`, and `ncrename`) alter only program metadata, and perform no floating point arithmetic. Metadata alteration includes changing attributes, names, dimension sizes, and dimension ordering.

3.3 Metadata Conventions

The netCDF data structure abstraction includes only dimensions, variables, and attributes (*Rew and Davis*, 1990). Metadata conventions extend the potential functionality of this abstraction by assigning special meaning to agreed-upon variables and attributes. NCO supports many metadata conventions, including those in Table 2.

[Table 2 about here.]

The netCDF authors introduced three of the most important metadata conventions NCO supports (*Rew et al.*, 2005). First, all operators support the History convention by appending their date-stamped invocation command line in the `history` global attribute. Second, arithmetic operators all support missing data by ignoring values equal to the value of the `missing_value` attribute. Third, all arithmetic operators work well with packed data, and two operators (`ncap` and `ncpdq`) can pack data themselves.

NCO correctly handles the ARM time offset convention by comparing hyperslab specifications for the `time` coordinate to the sum of the `base_time` and `time_offset` values. This permits, for example, maintaining a double precision time coordinate without sacrificing the first eight digits of precision to store the Julian Day. NCO uses the UDUnits library to translate hyperslab coordinates specified in "user" units, to "storage" units as indicated by the `units` attribute. *Zender* (2008) describes the supported metadata conventions.

3.4 Parallelism

As indicated in Table 1, all arithmetic operators support Shared Memory Parallelism (SMP) and distributed parallelism. These parallelisms are implemented and controlled with standard OpenMP http://www.openmp.org and Message-Passing Interface (MPI) (*Snir et al.*, 1998) techniques respectively. Currently the OpenMP and MPI parallelism operate exclusively, and "hybrid" (OpenMP threads within MPI processes) parallelism is not supported.

The arithmetic operators (except `ncap`) are parallelized (operate independently) over the loop of variables in the current file. `ncap` performs a dependency analysis on the input script and then parallelizes the execution over independent groups of statements (called "basic blocks" in compiler terminology). The operators automatically utilize SMP parallelism when compiled with an OpenMP-compliant compiler. The SMP parallelism increases operator throughput when the number of arithmetic operations per thread is large enough to compensate for the cost of spawning the threads. The operators will spawn pre-set optimal numbers of threads which the user may override with the `OMP_NUM_THREADS` environment variable (*OpenMP*, 2005) or with the `-t` switch, e.g.,

`ncwa -t 4 in.nc out.nc`.

MPI versions of the parallelized arithmetic operators begin with with `mp` (e.g., `mpncbo`). The variables in the current file are distributed over the available MPI processes. NCO takes advantage of the parallelism permitted by the current netCDF3 library—multiple simultaneous file-reads and a single file-write at a time. Extending and adding parallelism to NCO's I/O is an area of current research.

3.5 Network Transparency

Geoscience researchers are increasingly interested in inter-comparing their results with those stored at geographically disparate sites. NCO supports a number of mechanisms to access files stored across networks (Table 3).

[Table 3 about here.]

NCO synchronously copies remote files to the local file system as necessary. This copying always extends the elapsed time to completion relative to comparable analysis of local datasets. Nevertheless, such copying is often acceptable and even desirable for unmonitored "batch" data analysis or operational data analysis which utilizes NCO in continual scripts.

OPeNDAP intercepts netCDF library calls and executes them on the remote file using HTTP access requests (*Cornillon et al.*, 2003). Hence OPeNDAP copies only the requested data across the network. This can lead to a significant speed advantage when the user operates on small subsets of remote files. The widespread support for OPeNDAP among the climate data analysis toolkits mentioned in Section 1 (CDAT, CDO, GrADS, and NCL) is indicative of this advantage.

3.6 An Integrated Example and its Analysis

The NCO operators each perform rather simple tasks so it is worthwhile to see how these commands can be linked together to perform more sophisticated analyses. It is possible to use a combination of NCO operations to compute variances and standard deviations of fields stored in a single

file or across multiple files. Computing the standard deviation of a time-series across multiple files is a four-step procedure:

```
ncrcat in*.nc tmp_1.nc           # Place all input in one file
ncwa -a time tmp_1.nc tmp_2.nc   # Get gridpoint time-mean values
ncbo tmp_1.nc tmp_2.nc tmp_3.nc # Compute gridpoint anomalies
ncra -y rmssdn tmp_3.nc out.nc   # Combine into standard deviation
```

The first step assembles all the data into a single file (this step would be unnecessary if the fields were already stored in a single file). Filename wildcard expansion is used so that exact knowledge (or typing) of input filenames is not required. This may temporarily consume large amounts of disk space. The second step creates the time-mean value of each gridpoint. The user only needs to provide the time coordinate name so that a temporal (rather that, e.g., spatial) standard deviation is calculated. Step three computes each gridpoint anomaly as the difference between the time-varying and the time-mean data. The fourth step finishes by computing the standard deviation from the anomalies.

There is no need for an operator designed specifically to compute multi-file standard deviations since the four commands above can easily be converted to a shell script. NCO tries to solve complex data analysis problems using a small number of fundamental operators to perform common data transformations. Monolithic approaches with large function libraries can accomplish as much and more, yet tend to have steeper learning curves and to require longer scripts than NCO.

The standard deviation procedure above (and similar scripts) works "as is" on unlimited numbers of files stored locally or remotely (Section 3.5) with arbitrary numbers of timesteps in each file. The input files may contain any number of floating point or integer variables with any names and dimensionality, so long as all files have the same variables and non-record dimensions as the first file. The input files may store these variables in any order, packed or unpacked, with or without missing data (Section 3.3). NCO automagically anticipates and handles these and other complicating factors (e.g., exploiting SMP parallelism) transparently to the user. In accord with the design philosophy (Section 2), the user may have little or no knowledge of these details because the op-

erators behave sensibly by default. Like UNIX commands, NCO's power derives from combining elementary operations together.

The performance and scaling of dataset analysis using NCO on input files with the same schema and typical file geometries is assessed in *Zender and Mangalam* (2007). However, users often wish to analyze input files whose schemas differ in ways that NCO does not automatically understand. For example, the name and spatial grid of the temperature field may differ between the model and satellite-derived sources that the user wishes to intercompare. In such cases, use NCO operators (e.g., `ncrename`) and other netCDF toolkits (Section 1) to pre-process (e.g., rename, re-grid) input datasets before commencing inter-file arithmetic.

4 Future Plans

As an Open Source software project (*Raymond*, 1999), NCO will continue to evolve to meet the needs of its authors and most vocal users. We aim for NCO to comply more completely with geoscience metadata standards such as those in Table 2. Typically metadata standards are often easier to define than to implement. Whereas specific applications only need to implement the standard to suit their own purposes, generic applications such as NCO are destined to encounter unforeseen or difficult uses of the standard. Priorities for future NCO support include metadata conventions which define representation of reduced, staggered, and non-rectangular data grids (*Gregory*, 2003).

The institutional support NCO currently receives allows us to also tackle fundamental problems in distributed geoscience data analysis. The current netCDF library restricts file-writes to a single process at a time. Parallel I/O offers potentially dramatic improvements in operator throughput (*Gropp et al.*, 1999). Exploiting this opportunity by extending the NCO arithmetic parallelism, already implemented, through to the I/O layer seems achievable with current and near-future software libraries. Parallel netCDF (pnetCDF) (*Li et al.*, 2003) currently offers an MPI-IO implementation of the netCDF3 format which helps reduce I/O bottlenecks for datasets stored on parallel

file systems. netCDF4 has an HDF5 back end (HDF; http://hdf.ncsa.uiuc.edu). which supports MPI-IO (*Rew et al.*, 2006). We plan to analyze and inter-compare the performance of the shared memory and distributed parallelism on common arithmetic tasks in a future study.

Gridded netCDF data accessible to NCO via its OPeNDAP capabilities include the Earth System Grid (*Foster et al.*, 2002) and the multi-model database used by the Intergovernmental Panel on Climate Change (IPCC) to write its fourth climate assessment report (*IPCC*, 2007). The IPCC mandated that models adhere to the netCDF format and to many of the metadata conventions illustrated in this paper. More than 250 peer-reviewed scientific publications have used the IPCC datasets as a result of this forethought, coordination, and open access (`http://www-pcmdi.llnl.gov/ipcc/subproject_publications.php`). The widespread use of these internationally shared climate data demonstrates the potential for producers and users of other environmental modeling software to leverage their models and data. By understanding the data analysis practices and principles illustrated in this paper, environmental scientists can learn to create and manipulate gridded datasets which are easily shared with and used by their international colleagues.

Acknowledgments

H. Butowsky and R. Peterson generously contributed their time to NCO. R. Ziemlinski and two anonymous reviewers provided helpful comments on this manuscript. Unidata staff developed the netCDF software, and kindly answered my questions and accommodated my visits. This material is based upon work supported by the National Science Foundation under Grants ATM-0231380 and IIS-0431203. Download this manuscript from http://dust.ess.uci.edu/ppr/ppr_Zen08.pdf.

Bibliography

Cornillon, P., J. Gallagher, and T. Sgouros (2003), OPeNDAP: Accessing data in a distributed heterogeneous environment, *Data Science Journal*, *2*, 164–174. 1, 3.5

Domenico, B., J. Caron, E. Davis, R. Kambic, and S. Nativi (2002), Thematic Real-time Environmental Distributed Data Services (THREDDS): Incorporating interactive analysis tools into NSDL, *Journal of Digital Information*, *2*(4), art. #114. 1

Fiorino, M., and D. Williams (2002), The PCMDI Climate Data Analysis Tools (CDAT)—an open system approach to the implementation of a model diagnosis infrastructure, in *Proceedings of the 18th International Conference on Interactive Information and Processing Systems for Meteorology*, p. J3.22, American Meteorological Society, AMS Press, Boston, MA, January 11–15, Seattle, WA. 1

Foster, I., et al. (2002), The Earth System Grid II: Turning climate datasets into community resources, in *Proceedings of the 18th International Conference on Interactive Information and Processing Systems for Meteorology*, American Meteorological Society, AMS Press, Boston, MA, January 11–15, Seattle, WA. 1, 4

Gregory, J. (2003), The CF metadata standard, *CLIVAR Exchanges*, *8*(4), 4. 4, 2

Gropp, W., E. Lusk, and R. Thakur (1999), *Using MPI-2: Advanced Features of the Message-Passing Interface*, 382 pp., MIT Press, Cambridge, MA. 4

IPCC (2007), Climate change 2007: The physical science basis. Contribution of Working Group I to the fourth assessment report of the Intergovernmental Panel on Climate Change, p. 996, Cambridge Univ. Press, Cambridge, UK, and New York, NY, USA. 4

Li, J., et al. (2003), Parallel netCDF: A high-performance scientific I/O interface, in *Proceedings of the 2003 ACM/IEEE Conference on Supercomputing*, pp. 39–49, Association for Computing Machinery, IEEE Computer Society, Washington, DC, USA, November 15–21, Phoenix, AZ. 4

Newham, C., and B. Rosenblatt (1998), *Learning the bash Shell*, second ed., 318 pp., O'Reilly, Sebastopol, CA. 1

NRC (2001), *Grand Challenges in Environmental Sciences*, 96 pp., National Research Council, National Academy Press, Washington, DC. 1

OpenMP (2005), *OpenMP Application Program Interface, Version 2.5*, OpenMP Architecture Review Board, http://www.openmp.org. 3.4

Raymond, E. S. (1999), *The Cathedral & the Bazaar*, O'Reilly Inc., Sebastopol, CA. 4

Rew, R., and G. Davis (1990), NetCDF: an interface for scientific data access, *IEEE Comput. Graph. Appl.*, *10*(4), 76–82, doi:10.1109/38.56302. 1, 4, 3, 3.3

Rew, R., G. Davis, S. Emmerson, and H. Davies (2005), *The NetCDF Users' Guide, Version 3.6.1*, University Corporation for Atmospheric Research, Boulder, CO, `http://www.unidata.ucar.edu/packages/netcdf`. 3, 3.3, 2

Rew, R., E. Hartnett, and J. Caron (2006), NetCDF-4: Software implementing an enhanced data model for the geosciences, in *Proceedings of the 22nd AMS Conference on Interactive Information and Processing Systems for Meteorology*, p. 6.6, American Meteorological Society, AMS Press, Boston, MA. 4

Snir, M., S. Otto, S. Huss-Lederman, D. Walker, and J. Dongarra (1998), *MPI: The Complete Reference. Volume 1, The MPI Core*, second ed., 426 pp., MIT Press, Cambridge, MA. 3.4

Wang, D. L., C. S. Zender, and S. F. Jenks (2007), Server-side parallel data reduction and analysis, in *Advances in Grid and Pervasive Computing, Second International Conference, GPC 2007*, *IEEE Lecture Notes in Computer Science*, vol. 4459, edited by C. Cérin and K.-C. Li, pp. 744–750, Springer-Verlag, Berlin/Heidelberg. 1

Woolf, A., K. Haines, and C. Liu (2003), A web service model for climate data access on the grid, *Int. J. High Perform. Comput. Appl.*, *17*(3), 281–295. 1

Zender, C. S. (2008), NCO User's Guide, version 3.9.4, `http://nco.sf.net/nco.pdf`. 1, 3, 3.3

313 Zender, C. S., and H. J. Mangalam (2007), Scaling properties of common statistical op-
314 erators for gridded datasets, *Int. J. High Perform. Comput. Appl.*, *21*(4), 458–498,
315 doi:10.1177/1094342007083,802. 1, 3.1, 3.6

Table 1: Operator Summary

Command	Name (primary functionality)	Type[a]	MFO[b]	Par.[c]
`ncap`	Arithmetic Processor (algebra, derived fields)	A		✓
`ncatted`	Attribute Editor (change attributes)	M		
`ncbo`	Binary Operator (subtraction, addition ...)	A		✓
`ncea`	Ensemble Averager (means, min/max, ...)	A	✓	✓
`ncecat`	Ensemble Concatenator (join files)	M	✓	✓
`ncflint`	File Interpolator	A		✓
`ncks`	Kitchen Sink (sub-set, hyperslab, ...)	M		
`ncpdq`	Pack Data, Permute Dimensions	A/M		✓
`ncra`	Record Averager (means, min/max, ...)	A	✓	✓
`ncrcat`	Record Concatenator (join time-series)	M	✓	✓
`ncrename`	Renamer (rename any metadata)	M		
`ncwa`	Weighted Averager (average, mask, integrate, ...)	A		✓

[a]Operator type: "A" and "M" indicate arithmetic and metadata operators, respectively.

[b]Multi-file Operators—Operators which process an arbitrarily large number ($N > 2$) of input files.

[c]Operator parallelism. These operators exploit shared memory parallelism (SMP) on OpenMP-compliant platforms, and distributed parallelism with MPI.

Table 2: Supported Metadata Conventions

Purpose	Convention	Reference
History, missing data, packing	netCDF	*Rew et al.* (2005)
Time offsets	ARM[a]	http://www.arm.gov/-data/time.stm
Coordinates	CF[b]	*Gregory* (2003)
Units translation	UDUnits	http://www.unidata.-ucar.edu/packages/-udunits

[a]US Department of Energy Atmospheric Radiation Measurement (ARM) Program.
[b]Climate and Forecast conventions

Table 3: Cross-Network File Access

Protocol	Command[a]	Sample File Specification
File Transfer Protocol	`ftp`	ftp://host/pub/user/data/in.nc
FTP, password-protected	`ftp`	Same as FTP, requires ~/.netrc
NCAR Mass Store	`msrcp`	/USER/data/in.nc or mss:/USER/data/in.nc
Remote Shell	`rcp`	host:/data/in.nc
Secure Shell	`scp`	host:/data/in.nc
Secure FTP	`sftp`	sftp://host:/home/ftp/pub/user/data/in.nc
OPeNDAP	HTTP	http://host/cgi-bin/dods/nph-dods/dodsdata/in.nc

[a]Underlying command or protocol NCO uses to access remote data.