

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

by Charles S. Zender\*

Department of Earth System Science

[zender@uci.edu](mailto: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

## 16 **Software Availability**

17 Name of software: netCDF Operators (NCO)

18 Developer: Charles S. Zender

19 Operating system: All

20 Programming languages: C/C99/C++

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

## 22 **1 Introduction**

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

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

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

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

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

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

## 71 **2 Design Philosophy**

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

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

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

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

95 3. An audit trail that tracks data provenance and processing history is desirable for both the data  
96 analyst and their colleagues who receive the processed data. For analysis involving multi-file  
97 sequences, the metadata in the first file, along with a list of the other files, adequately pre-  
98 serves the processing history. By convention, NCO keeps this information in the `history`  
99 attribute ([Rew et al., 2005](#)).

100 4. There is value in maintaining the distinctions and associations between *dimensions*, *coordi-*  
101 *nates*, and *variables* ([Rew and Davis, 1990](#)) during data analysis. Unless otherwise specified,  
102 NCO automatically attaches coordinate data (i.e., dimension values) to variables it transfers.

103 5. Tools should treat data as generically as possible, and impose no software limitations on data  
104 dimensionality, size, type, or ordering.

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

### 107 **3 Operators**

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

111 [Table 1 about here.]

112 Operator names are acronyms for their functionality, prefixed with “nc” to indicate their relation-  
113 ship to netCDF. The twelve operators typically read netCDF files as input, perform some manip-  
114 ulations, then write netCDF files as output. In this sense the operators are filters, or middleware.  
115 The NCO User's Guide ([Zender, 2008](#)) documents the functionality and calling conventions for all  
116 operators.

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

### 123 **3.1 Arithmetic Operators**

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

### 130 **3.2 Metadata Operators**

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

### 134 **3.3 Metadata Conventions**

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

139 [Table 2 about here.]

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

146 NCO correctly handles the ARM time offset convention by comparing hyperslab specifications  
147 for the `time` coordinate to the sum of the `base_time` and `time_offset` values. This permits,  
148 for example, maintaining a double precision time coordinate without sacrificing the first eight  
149 digits of precision to store the Julian Day. NCO uses the UDUnits library to translate hyperslab  
150 coordinates specified in “user” units, to “storage” units as indicated by the `units` attribute. *Zender*  
151 (*2008*) describes the supported metadata conventions.

## 152 **3.4 Parallelism**

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

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

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

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

### 172 **3.5 Network Transparency**

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

176 [Table 3 about here.]

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

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

### 186 **3.6 An Integrated Example and its Analysis**

187 The NCO operators each perform rather simple tasks so it is worthwhile to see how these com-  
188 mands can be linked together to perform more sophisticated analyses. It is possible to use a combi-  
189 nation of NCO operations to compute variances and standard deviations of fields stored in a single

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

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

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

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

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

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

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

## 226 4 Future Plans

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

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

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

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

## 257 **Acknowledgments**

258 H. Butowsky and R. Peterson generously contributed their time to NCO. R. Ziemlinski and two  
259 anonymous reviewers provided helpful comments on this manuscript. Unidata staff developed the  
260 netCDF software, and kindly answered my questions and accommodated my visits. This material  
261 is based upon work supported by the National Science Foundation under Grants ATM-0231380  
262 and IIS-0431203. Download this manuscript from [http://dust.ess.uci.edu/ppr/ppr\\_Zen08.pdf](http://dust.ess.uci.edu/ppr/ppr_Zen08.pdf).

## 263 **Bibliography**

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

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

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

274 Foster, I., et al. (2002), The Earth System Grid II: Turning climate datasets into commu-  
275 nity resources, in *Proceedings of the 18th International Conference on Interactive Informa-*  
276 *tion and Processing Systems for Meteorology*, American Meteorological Society, AMS Press,  
277 Boston, MA, January 11–15, Seattle, WA. [1](#), [4](#)

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

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

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

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

287 Newham, C., and B. Rosenblatt (1998), *Learning the bash Shell*, second ed., 318 pp., O’Reilly,  
288 Sebastopol, CA. [1](#)

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

291 OpenMP (2005), *OpenMP Application Program Interface, Version 2.5*, OpenMP Architecture Re-  
292 view Board, <http://www.openmp.org>. 3.4

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

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

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

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

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

305 Wang, D. L., C. S. Zender, and S. F. Jenks (2007), Server-side parallel data reduction and analysis,  
306 in *Advances in Grid and Pervasive Computing, Second International Conference, GPC 2007*,  
307 *IEEE Lecture Notes in Computer Science*, vol. 4459, edited by C. Cérin and K.-C. Li, pp. 744–  
308 750, Springer-Verlag, Berlin/Heidelberg. 1

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

311 Zender, C. S. (2008), NCO User's Guide, version 3.9.4, <http://nco.sf.net/nco.pdf>. 1,  
312 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 <sup>a</sup>	MFO <sup>b</sup>	Par. <sup>c</sup>
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		✓

<sup>a</sup>Operator type: “A” and “M” indicate arithmetic and metadata operators, respectively.

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

<sup>c</sup>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	<i>Rew et al. (2005)</i>
Time offsets	ARM <sup>a</sup>	<a href="http://www.arm.gov/-data/time.stm">http://www.arm.gov/- data/time.stm</a>
Coordinates	CF <sup>b</sup>	<i>Gregory (2003)</i>
Units translation	UDUnits	<a href="http://www.unidata.-ucar.edu/packages/-udunits">http://www.unidata.- ucar.edu/packages/- udunits</a>

<sup>a</sup>US Department of Energy Atmospheric Radiation Measurement (ARM) Program.

<sup>b</sup>Climate and Forecast conventions

**Table 3: Cross-Network File Access**

Protocol	Command <sup>a</sup>	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

<sup>a</sup>Underlying command or protocol NCO uses to access remote data.