3B.4

HIGH PERFORMANCE DISTRIBUTED DATA REDUCTION AND ANALYSIS WITH THE NETCDF OPERATORS (NCO)

Charles S. Zender* and Daniel L. Wang
University of California, Irvine

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*, 2006a) fully documents NCO's functionality and calling conventions. *Zender* (2006b) describes NCO's design philosophy, primary features, relation to other geoscience data analysis software, and future plans. *Zender and Mangalam* (2007) describe the core NCO arithmetic algorithms and their theoretical and measured scaling with dataset size and structure. Current research provides NCO with advanced parallel computing techniques at two distinct levels. At the low level, all NCO arithmetic operators are parallelized to throughput on shared memory and distributed memory clusters (Zender et al. 2007, manuscript in preparation). High level analysis scripts of multiple NCO commands benefit from out new dependency-analysis engine which automatically detects and parallelizes basic blocks, returning intermediate files only as needed (Wang et al., 2006, manuscript in preparation). This extended abstract summarizes novel features in NCO's design, arithmetic algorithms, and low- and high-level parallelization.

2. DESIGN

Traditional scientific 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 five 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 command-line *globbing* of multiple files, NCO effectively subsumes two problematic loops (loops over files and over variables) of large scale data-processing 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* 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.

* *Corresponding author address:* Charles S. Zender, Dept. of Earth System Science, University of California, Irvine, Irvine, CA 92697-3100; e-mail: zender@uci.edu.

Table 1: Operator Summary

Command	Name (primary functionality)	Type[1]	MFO[2]	Par.[3]
ncap/ncap2	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		✓

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). 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 and middleware. The NCO User's Guide (*Zender*, 2006a) fully 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. ALGORITHMS

Improving data statistics and reducing noise by averaging are important analysis techniques because they help reveal fundamental patterns. Many other useful operations share with averaging the property that the result is of lesser rank (has fewer dimensions or axes) than the input. First we illustrate a typical arithmetic-intensive NCO averaging operation, then we describe the flexible, high performance algorithm which carries it out.

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

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

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

3.1 *Averaging*

The netCDF weighted averager ncwa implements a useful set of such rank-reduction operations with optional masks and weights, e.g.,

```
ncwa 1986.nc 1986_txyz.nc
ncwa -a x,y -w area -B "ocean == 1" \
     1986.nc 1986_ocean_avg.nc
ncwa -a x,y -w area -d lat,-30.,30. \
     1986.nc 1986_tropics_avg.nc
ncwa -y min -B "land == 1" \
     1986.nc 1986_land_min.nc
```

The first command averages all variables to scalars. The second command determines ocean-wide averages by averaging all variables along their x and y axes, weighting each datum by the value of the corresponding element of area, and including only data whose corresponding ocean flag is true. Such masks are a convenient way of identifying irregular regions (like ocean basins). Rectangular regions are easily specified using hyperslabs, as in the third example. The fourth command determines the minimum of all variables over land.

The record averager ncra swiftly reduces an arbitrary number of datasets along their record dimension (usually the time axis). The output is the time-mean of (usually) spatially varying data,

```
ncra 1986_*.nc ~/1986.nc
ncra -n 274,3,1 1986_060.nc ~/1986_0311.nc
ncra -y min 1986_*.nc ~/1986.nc
```

The first command time-averages all datasets beginning with "1986_". The second command specifies the input files using a shorthand convention for 274 files whose last three digits before the suffix increase by 1 each from the template filename. If the three digits represent day-of-year, then the output is the the March-November average. With the same assumption, the last command finds stores the annual minimum daily values for 1986.

Table 2: File Geometries

	Satellite	GCM
Max. Rank R	2	4
Variables	8^4	128^5
Time	—	8
Level	—	32
Latitude	2160	128
Longitude	4320	256
Mean[6] $\tilde{D}$	3055	55
Elements N [#]	75×10^6	285×10^6
Total Size [MB]	299	1143

As can be seen from these examples, the "averagers" (`ncwa`, `ncra`, `ncea`) are misnamed because they perform many non-linear operations as well, e.g., total, minimum, maximum, root-mean-square. A more accurate but less familiar name would be "reducers", since their fundamental purpose is to reduce a dataset's rank.

3.2 *Optimizations*

NCO is designed for gridded geophysical data that obey

$$N \gg D \gg R > 1 \tag{1}$$

where N is the size (number of elements) of a variable, D is the length of a dimension (e.g., spatial or temporal), and R is the rank (number of dimensions) in the variable. In our experience, this assumption applies to most gridded data generated or stored on high performance computers. The exact size N of a rank R variable is

$$N \equiv \prod_{k=1}^{k=R} D_k \tag{2}$$

Arithmetic requires maximum predictable numbers F of floating point operations and I of integer operations. *Zender and Mangalam* (2007) analyze the costs of performing arithmetic (e.g., averaging) datasets with different ranks ranging from unstructured (i.e., $R = 1$) to structure typical of time-varying geoscience datasets (i.e., $R = 4$).

Many steps contribute significantly to the elapsed time required to complete a relatively simple arithmetic procedure such as averaging. These include I/O, byte-swapping, broadcasting (e.g., conforming scalars to multi-dimensional variables), collection (e.g., accessing discontinuous hyperslabs in memory), and weighting. In some geoscience disciplines, weighting data (e.g., by grid-cell area) prior to averaging is ubiquitous. We tested NCO on datasets representative of typical Satellite and Intergovernmental Panel on Climate Change (IPCC) GCM simulations (e.g., *Cubasch and Meehl*, 2001). Table 2 summarizes the geometries in the Satellite and GCM datasets tested. The Satellite and GCM datasets

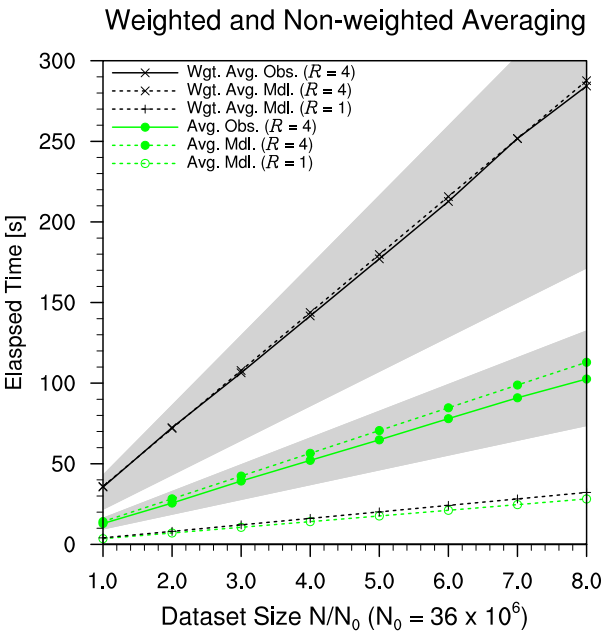

Fig. 1: Observed (solid) and predicted (dashed) elapsed time to perform weighted and non-weighted averages on N element GCM-geometry ($R = 4$) and unstructured ($R = 1$) datasets. Grey areas indicate prediction range for rank $R = 2$–5 datasets. Horizontal axis scaled to units of $N_0 = 36 \times 10^6$ elements.

are examples of ranks $R = 2$ and 4 geometries, respectively.

Interestingly, weighted averaging takes about about three times longer than non-weighted averaging when no optimizations are applied (Figure 1).

NCO's optimized arithmetic algorithms take advantage of run-time opportunities including data storage order (i.e., dataset shape), and isomorphisms between current and previous operations which allow re-use of cached intermediate variables. The most rapidly varying (MRV) algorithm is a storage-order optimization that NCO applies to input hyperslabs stored in the same order required to average it. The weight re-use (WRU) algorithm eliminates all but the first broadcast of averaging weights for a set of identically shaped variables.

The MRV and WRU optimizations significantly reduce the total integer operations required during

[4] All eight variables are rank $R = 2$.

[5] Eight variables are scalars ($R = 0$), eight variables contain the time dimension only ($R = 1$), sixteen variables contain only latitude and longitude ($R = 2$), sixty-four variables have time, latitude, and longitude ($R = 3$), thirty-two variables have contain all four dimensions ($R = 4$).

[6] Mean dimension size (weighted by variable size and number).

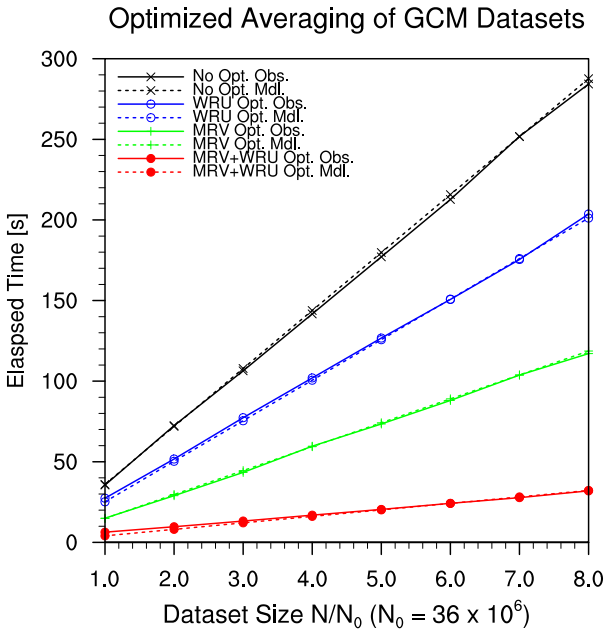

Fig. 2: Observed (solid) and predicted (dashed) elapsed times to perform a weighted average of an N element GCM-geometry ($R = 4$) dataset with and without WRU and MRV and both optimizations. Un-optimized curves same as Figure 1.

weighted averaging

$$I_1 \approx N[34R + 8R_w + 25 + W + (W+11)N_A^{-1}] + B \quad (3a)$$

$$I_2 \approx N[28R + 0 + 23 + W + (W+11)N_A^{-1}] + B \quad (3b)$$

$$I_3 \approx N[6R + 8R_w + 17 + W + (W+11)N_A^{-1}] + B \quad (3c)$$

$$I_4 \approx N[0 + 0 + 15 + W + (W+11)N_A^{-1}] + B \quad (3d)$$

where I_1 is the operation count required without optimizations, and I_2, I_3, and I_4, include the WRU, MRV, and both optimizations, respectively. Here R_w is the rank of the weight (e.g., gridcell area), W is the data wordsize (i.e., four or eight-bytes), and N_A and N_w are the products of the sizes of the averaged and weight dimensions, and $B = (W+2)N_w$.

The MRV dimension optimization boosts performance even more (Figure 2). The worst case times to perform weighted averages occur when neither the WRU nor the MRV optimizations apply or, equivalently, if the averaging software implements "brute force" techniques rather than the WRU and/or MRV optimizations.

The WRU optimization alone reduces the elapsed time to average typical IPCC-style GCM datasets from 285 s to 200 s. This is a throughput increase of about 40%. However, not all datasets are as amenable to WRU as IPCC-style GCM-datasets. For instance, averaging variables which alternated in shape rather than averaging groups of identically shaped-variables (as we did) could significantly degrade the WRU benefits.

The MRV optimization alone increases GCM dataset averaging throughput by about 140%. Figure 2 shows that MRV causes impressive gains when averaging over all dimensions (i.e., $R_A = R$, where R_A is the rank of the averaging space). In fact, MRV optimization is nearly as effective for partial averages ($R_A < R$, not shown). The MRV improvement is distinct from and in addition to the acceleration which hardware-based caching provides for accessing contiguous non-strided datasets, such as MRV averages.

The MRV and WRU optimizations are often applicable in tandem. The MRV and WRU combination reduces the elapsed time to average the GCM dataset from 285 s to 30 s. This order-of-magnitude throughput increase agrees well with the ratio I_1/I_4 (3) for typical GCM datasets which have $R = 4$ and $R_w = 1$. The combination of MRV and WRU optimizations shifts the I/O time for weighted averaging of the GCM dataset from $< 10\%$ to about 50%.

4. SMP AND SPMD PARALLELIZATION

As indicated in Table 1, all arithmetic operators except `ncap` support Shared Memory Parallelism (SMP) and distributed parallelism. These parallelisms are implemented and controlled with standard OpenMP (*OpenMP*, 2005) 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 are parallelized (operate independently) over the loop of variables in the current file. 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.

Compute-intensive operators (`ncwa` and `ncpdq`) benefit most from threading. The greatest increases in throughput due to threading occur on large dataset where each thread performs millions or more floating point operations. Otherwise, the system overhead of setting up threads tends to outweigh the theoretical speed enhancements due to SMP parallelism. SMP parallelism does not currently scale well beyond four threads for these operators. Removing I/O bottlenecks is a high priority (see below), as is parallelizing the next generation arithmetic processor, `ncap2`.

5. SERVER-SIDE DATA REDUCTION

Low storage costs have driven an explosion in the availability of geoscience datasets, but their usage has been hampered by the cost and time involved in network transfer. To address this, we have implemented a server-side computation engine that allows scripts of NCO commands to be performed at the server. Our system, currently called SSDAP (Server-Side DAP) piggybacks on the Data Access Protocol (DAP) (*Cornillon et al.*, 2003) of a customized OPeNDAP data handler (*OPeNDAP*, 2004). NCO commands are sent through an interface extended from DAP's subsetting facility and processed by a server-side execution engine. Resultant datasets may be retrieved in the same DAP request, or deferred for later retrieval.

Recognizing the wide use of scripting to perform data reduction and analysis, SSDAP is capable of parsing and executing most typical Bourne-shell scripts of NCO commands with only light modifications. Because such scripts typically perform massive reductions, a system that merely transmits script results is capable of more efficient use of network bandwidth. In addition to benefits in data-computation locality, a server-side computation engine intersperses a layer between script submission and execution, allowing compiler techniques such as dataflow analysis to be applied at the script-level. With such techniques, the engine can parallelize execution of independent NCO commands and better exploit server-class hardware. Early tests confirm the potential for increased computational performance. For one 14000+ line script, execution time was reduced from 74 minutes for a bare execution to 45 minutes for compilation and parallel execution on the same dual-core, dual CPU machine. The abstract by Wang et al. in this volume contains more details.

Server-side computation should significantly lower costs of data-intensive computation, increasing the practicality of desktop analysis of remote terascale datasets. With a script-based interface, our implementation hopes to enable access to server-side computation and script-level parallelization for geoscientists.

6. STATUS AND FUTURE WORK

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 CF. 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 will analyze and inter-compare the performance of the shared memory and distributed parallelism on common arithmetic tasks in a future study.

ACKNOWLEDGEMENTS

We thank H. Butowsky and R. Peterson for their NCO contributions, H. Mangalam for help with benchmarks, S. Jenks for stimulating discussions on dataflow parallelization, and R. Rew and the Unidata crew for creating, maintaining, and extending netCDF. This material is based upon work supported by the National Science Foundation under Grants ATM-0231380 and IIS-0431203.

Bibliography

Cornillon, P., J. Gallagher, and T. Sgouros (2003), OPeNDAP: Accessing data in a distributed heterogeneous environment, *Data Science Journal*, *2*, 164–174. 1, 5

Cubasch, U., and G. Meehl (2001), Projections of future climate change, in *Climate Change 2001: The Scientific Basis. Contribution of Working Group I to the Third Assessment Report of the Intergovernmental Panel on Climate Change*, edited by J. T. Houghton, Y. Ding, D. J. Griggs, M. Noguer, P. J. van der Linden, X. Dai, K. Maskell, and C. A. Johnson, chap. 9, pp. 527–578, Cambridge Univ. Press, Cambridge, UK, and New York, NY, USA. 3.2

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

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

Gregory, J. (2003), The CF metadata standard, *CLIVAR Exchanges*, *8*(4), 4. 6

Gropp, W., E. Lusk, and R. Thakur (1999), *Using MPI-2: Advanced Features of the Message-Passing Interface*, 382 pp., MIT Press, Cambridge, MA. 6

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. 6

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

OPeNDAP (2004), Open-source Project for a Network Data Access Protocol, http://www.opendap.org. 5

OpenMP (2005), *OpenMP Application Program Interface, Version 2.5*, OpenMP Architecture Review Board, http://www.openmp.org. 4

Raymond, E. S. (1999), *The Cathedral & the Bazaar*, O'Reilly Inc., Sebastopol, CA. 6

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. 2

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

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. 6

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. 4

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. (2006a), NCO User's Guide, version 3.1.4, http://nco.sf.net/nco.pdf. 1, 2

Zender, C. S. (2006b), netCDF Operators (NCO) for analysis of self-describing gridded geoscience data, *Submitted to Environ. Modell. Softw.*, available from http://dust.ess.uci.edu/ppr/ppr_Zen07.pdf. 1

Zender, C. S., and H. J. Mangalam (2007), Scaling properties of common statistical operators for gridded datasets, *In Press in Int. J. High Perform. Comput. Appl.*, available from http://dust.ess.uci.edu/ppr/ppr_ZeM07.pdf. 1, 3.2