A Quick Guide for the pbdNCDF4 Package (Ver. 0.1-3)

Pragneshkumar Patel[1], George Ostrouchov[1,2], Wei-Chen Chen[3], Drew Schmidt[1], David Pierce[4]

[1]National Institute for Computational Sciences,
University of Tennessee,
Knoxville, TN, USA

[2]Computer Science and Mathematics Division,
Oak Ridge National Laboratory,
Oak Ridge, TN, USA

[3]pbdR Core Team

[4]Climate Research Division,
Scripps Institution of Oceanography,
UC San Diego,
San Diago, CA, USA

Contents

Acknowledgement iii

1. Introduction 1

1.1. System Requirements 1
1.2. Installation 2
1.3. A Quick Example 2

2. Non-Standard System Installations 3

2.1. Special Path for NetCDF4 3
2.2. Parallel HDF5 and NetCDF4 4

3. Collective I/O of pbdNCDF4 4

4. Windows System 5

4.1. Install from Binary 6
4.2. Build from Source 6
4.3. Detail Steps 7

References 8

This publication was typeset using LaTeX.

Acknowledgement

Patel, Ostrouchov, and Schmidt were supported in part by the project "NICS Remote Data Analysis and Visualization Center" funded by the Office of Cyberinfrastructure of the U.S. National Science Foundation under Award No. ARRA-NSF-OCI-0906324 for NICS-RDAV center. Ostrouchov and Chen were supported in part by the project "Visual Data Exploration and Analysis of Ultra-large Climate Data" funded by U.S. DOE Office of Science under Contract No. DE-AC05-00OR22725.

Chen was supported in part by the Department of Ecology and Evolutionary Biology at the University of Tennessee, Knoxville, and a grant from the National Science Foundation (MCB-1120370.)

This work used resources of National Institute for Computational Sciences at the University of Tennessee, Knoxville, which is supported by the Office of Cyberinfrastructure of the U.S. National Science Foundation under Award No. ARRA-NSF-OCI-0906324 for NICS-RDAV center. This work also used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725. This work used resources of the Newton HPC Program at the University of Tennessee, Knoxville.

We also thank Brian D. Ripley, Kurt Hornik, Uwe Ligges, and Simon Urbanek from the R Core Team for discussing package release issues and helping us solve portability problems on different platforms.

Warning: The findings and conclusions in this article have not been formally disseminated by the U.S. Department of Energy and should not be construed to represent any determination or policy of University, Agency and National Laboratory.

This document is written to explain the main functions of **pbdNCDF4** (Ostrouchov *et al.* 2012), version 0.1-3. Every effort will be made to ensure future versions are consistent with these instructions, but features in later versions may not be explained in this document.

Information about the functionality of this package, and any changes in future versions can be found on website: `http://r-pbd.org/`.

1. Introduction

The **pbdNCDF4** package is an interface to the parallel NetCDF4 library (NetCDF Group 2008) hosted by the Unidata program at the University Corporation for Atmospheric Research (UCAR). The package also requires the parallel HDF5 library (HDF Group 2000-2010) originally developed at the National Center for Supercomputing Applications. Full details about the system requirements and installation instructions for **pbdNCDF4** are provided in the following section.

The **pbdNCDF4** package is fully based on, and incorporates, the **ncdf4** package, version 1.8 by David Pierce (Pierce 2012).

Warning: We have success for installing and passing tests by using the following combinations:

- HDF5-1.8.5 and NetCDF-4.1.1,
- HDF5-1.8.8 and NetCDF-4.2.0, or
- HDF5-1.8.11 and NetCDF-4.3.0.

Other versions may have parallel I/O problems even installation is correct.

1.1. System Requirements

Before installing the **pbdNCDF4** package, there is some software which must first be installed on the user's system. Additionally, **pbdNCDF4** requires the **pbdMPI** (Chen *et al.* 2012a) package, which itself requires some of the same dependencies that **pbdNCDF4** requires, so you are encouraged to install **pbdMPI** first. See the **pbdMPI** vignette (Chen *et al.* 2012b) for full details on how to install it and its dependencies.

In order to install the **pbdNCDF4** package, you must first have system installations of:

1. An MPI library (openmpi, mpich2, ...). This is also required by **pbdMPI**.
2. HDF5 (version 1.8.5) `http://www.hdfgroup.org/HDF5`
3. NetCDF4 (version 4.1.1) `http://www.unidata.ucar.edu/software/netcdf/`

Ideally, the user needs to install the parallel versions of the HDF5 and NetCDF4 libraries, and in such case, these libraries should be compiled with MPI. The **pbdNCDF4** package functions

with serial installations of these libraries, but then only serial reading is possible, obviously. In the event that the parallel versions of the HDF5 and NetCDF4 libraries are not available, **pbdNCDF4** acts as **ncdf4**. See Section 3 for details.

1.2. Installation

The remaining assumes that **pbdMPI** is installed correctly. If **pbdMPI** is not yet installed, see the **pbdMPI** vignette for installation details. We also assume `nc-config`, a NetCDF4 utility which provides information about installation of NetCDF4 library, is in the user's `PATH`. See Section 2.1 for non-default installation if `nc-config` is not in the user's `PATH`.

Users can download **pbdNCDF4** from CRAN at http://cran.r-project.org, and the installation can be done with the following commands from the shell:

Shell Command

```
tar zxvf pbdNCDF4_0.1-0.tar.gz
R CMD INSTALL pbdNCDF4 --configure-args="--enable-parallel"
```

Note that without the flag `-configure-args="-enable-parallel"`, **pbdNCDF4** compiles with NCDF4 and HDF5 in serial. In this case, it is essentially the same as the **ncdf4** package. The extra parallel-enable R functions to **ncdf4** such as `nc_create_par`, `nc_open_par`, and `nc_var_par_access` behave as their serial counterparts. The same R code with **pbdNCDF4** can run either serial or parallel depending on the configuration used. See Section 3 for details.

1.3. A Quick Example

Users can get started quickly with **pbdNCDF4** by learning from the following two examples. Issued from the shell:

Shell Command

```
### Under command mode, run the demo with 2 processors by
### (Use Rscript.exe for windows system)

mpiexec -np 2 Rscript -e "demo(ncwrite_par,'pbdNCDF4',ask=F,echo=F)"
mpiexec -np 2 Rscript -e "demo(ncread_par,'pbdNCDF4',ask=F,echo=F)"
ncdump test_par.nc
```

The examples first write a file `test_par.nc`, then read it back in and print the results.

In this example, each processor writes a 4×5 column-major matrix (an 8×5 matrix in total). If the demos run successfully, the user can see the following output:

Output of `ncread`

```
COMM.RANK = 0
n = 8 p = 5
COMM.RANK = 0
     [,1] [,2] [,3] [,4] [,5]
[1,]    2    3    4    5    6
[2,]    2    3    4    5    6
[3,]    2    3    4    5    6
[4,]    2    3    4    5    6
```

```
COMM.RANK = 1
     [,1] [,2] [,3] [,4] [,5]
[1,]    3    4    5    6    7
[2,]    3    4    5    6    7
[3,]    3    4    5    6    7
[4,]    3    4    5    6    7
```

Note that NetCDF4 stores data in row-major fashion as in the C programming language. Above the data is represented in column-major fashion, as is used by R. Also, users can use the shell command `ncdump` (as in the third demo), a NetCDF4 utility, to show the exact contents of the `test_par.nc` file.

Output of `ncdump`

```
netcdf test_par {
dimensions:
        rows = 8 ;
        columns = 5 ;
variables:
        int rows(rows) ;
                rows:units = "number" ;
                rows:long_name = "rows" ;
        int columns(columns) ;
                columns:units = "number" ;
                columns:long_name = "columns" ;
        int testMatrix(columns, rows) ;
                testMatrix:units = "count" ;
                testMatrix:_FillValue = -1 ;
data:

 rows = 1, 2, 3, 4, 5, 6, 7, 8 ;

 columns = 1, 2, 3, 4, 5 ;

 testMatrix =
  2, 2, 2, 2, 3, 3, 3, 3,
  3, 3, 3, 3, 4, 4, 4, 4,
  4, 4, 4, 4, 5, 5, 5, 5,
  5, 5, 5, 5, 6, 6, 6, 6,
  6, 6, 6, 6, 7, 7, 7, 7 ;
}
```

Above, data are shown in the C, or row-major way.

2. Non-Standard System Installations

In this section we discuss a non-default method of installation for **pbdNCDF4**, as well as compiling the HDF5 and NetCDF4 libraries in parallel.

2.1. Special Path for NetCDF4

As the with the **ncdf4** package, **pbdNCDF4** allows special configuration for `nc-config` via the flag `--with-nc_config`. For example, we might issue the command:

Shell Command

```
R CMD INSTALL pbdNCDF4 \
  --configure-args="--with-nc-config=/usr/local/netcdf4/bin"
```

which specifies that `nc-config` is in the directory `/usr/local/netcdf4/bin`.

2.2. Parallel HDF5 and NetCDF4

Here we assume that the MPI headers are in `/usr/include/mpi` and the MPI libraries are in `/usr/lib`.

We can install HDF5 with parallel I/O via:

Compile parallel HDF5

```
./configure \
    --prefix=/usr/local/hdf5 \
    --enable-parallel \
    --enable-shared \
    CC="mpicc -g" \
    CFLAGS="-fPIC -I/usr/include/mpi" \
    CPPFLAGS="-fPIC -I/usr/include/mpi" \
    LDFLAGS="-L/usr/lib -lmpi"
make
make install
```

For parallel NetCDF4, suppose that the parallel HDF5 library is installed in `/usr/local/hdf5`. Then we can install NetCDF4 with parallel I/O via:

Compile parallel NetCDF4

```
./configure \
    --prefix=/usr/local/netcdf4 \
    --enable-netcdf4 \
    --enable-shared \
    CC="mpicc -g" \
    CFLAGS="-fPIC -I/usr/include/mpi -I/usr/local/hdf5/include" \
    CPPFLAGS="-fPIC -I/usr/include/mpi -I/usr/local/hdf5/include" \
    LDFLAGS="-L/usr/lib -lmpi -L/usr/local/hdf5/lib -lhdf5"
make
make install
```

3. Collective I/O of pbdNCDF4

There are mainly three parallel functions added to **pbdNCDF4** over **ncdf4**, which enable collective I/O. Namely, we include the R functions `nc_create_par()`, `nc_open_par()`, and `nc_var_par_access()`.

- `nc_create_par()` is similar to `nc_create()` in **ncdf4**, but creates a NetCDF4 file with parallel format.
- `nc_open_par()` is similar to `nc_open()` in **ncdf4**, but is specifically for files in the parallel format.
- `nc_var_par_access()` is used to tell the NetCDF4 library to use collective read or write for the given variable.

By default, `nc_var_par_access()` sets `collective=TRUE`, which turns on collective read and write for the given variable. If `nc_var_par_access()` is not called or `collective=FALSE` is set, then the independent method will be used.

If parallel versions of HDF5 and NetCDF4 libraries are not available or `--enable-parallel` is not set when compiling **pbdNCDF4**, then user needs to use the original **ncdf4** functions such as `nc_create` and `nc_open`. For reading in parallel, it will be a little bit slower than collective reading. However, the serial version should not be used for parallel writing unless manual synchronization is used.

We can run the serial example as in the Section 1.3 next without parallel read and write capabilities. Suppose serial HDF5 and NetCDF4 are compiled and **pbdNCDF4** is installed without the enable parallel flag `--enable-parallel`.

Shell Command

```
### Under command mode, run the demo with 2 processors by
### (Use Rscript.exe for windows system)

mpiexec -np 2 Rscript -e "demo(ncwrite_ser,'pbdNCDF4',ask=F,echo=F)"
mpiexec -np 2 Rscript -e "demo(ncread_ser,'pbdNCDF4',ask=F,echo=F)"
```

The first `demo()` uses two processors and independently writes to the file `test_ser.nc` in order to generate the same matrix, while the second `demo` reads back in and print the matrix in serial. The outputs are exactly the same as the parallel version. Note that the first demo `ncwrite_ser` provides an example of manual synchronization, see the source code for details.

4. Windows System

Windows system has serial HDF5 and NetCDF4 libraries available in both 32- and 64-bit systems. The pre-built (**netCDF-C**) libraries can be downloaded from http://www.unidata.ucar.edu/software/netcdf/docs/winbin.html. For example, the stable release (**netCDF-C 4.3.2**) has

- 32-bit library at http://www.unidata.ucar.edu/netcdf/win_netcdf/netCDF4.3.2-NC4-32.exe, and
- 64 bit library at http://www.unidata.ucar.edu/netcdf/win_netcdf/netCDF4.3.2-NC4-64.exe.

Note that both pre-built libraries contain HDF5 and NetCDF4 serial versions and can be directly linked and loaded with **pbdNCDF4** inside R.

All required `*.dll` files are via `dyn.load()` in `.onLoad()` at run time, and via `dyn.unload()` in `.onUnload()` when quiting the package.

With a few click on both `*.exe` files, we suggest users to install at the system directory

- `C:/Program Files (x86)/netCDF 4.3.2/` for the 32-bit library, and
- `C:/Program Files/netCDF 4.3.2/` for the 64-bit library.

Note that the default for 32-bit may be the same as 64-bit, so do the change manually since they can not be merged in general.

4.1. Install from Binary

The binary packages of **pbdNCDF4** are available on the website: "Programming with Big Data in R" at http://r-pbd.org/. The binary can be installed by

Shell Command

```
R CMD INSTALL pbdNCDF4_0.1-3.zip
```

As in Unix systems, one can start quickly with **pbdNCDF4** by learning from the following demos. There are two basic examples in serial.

Shell Command

```
demo(ncwrite_ser,'pbdNCDF4',ask=F,echo=F)
demo(ncread_ser,'pbdNCDF4',ask=F,echo=F)
```

4.2. Build from Source

Installation of **pbdNCDF4** in windows system requires users to set environment variables `NETCDF4_ROOT_32` and `NETCDF4_ROOT_64` for both pre-built libraries. By default, we suggest

Shell Command

```
### Under command mode, or save in a batch file.
SET NETCDF4_ROOT_32=C:\Program Files (x86)\netCDF 4.3.2\
SET NETCDF4_ROOT_64=C:\Program Files\netCDF 4.3.2\
SET NETCDF4_ROOT=%NETCDF4_ROOT_64%

SET R_HOME=C:\Program Files\R\R-3.0.1\
SET RTOOLS=C:\Rtools\bin\
SET MINGW=C:\Rtools\gcc-4.6.3\bin\

SET PATH=%R_HOME%;%R_HOME%bin\;%RTOOLS%;%MINGW%;%PATH%
SET PATH=%NETCDF4_ROOT%bin\;%PATH%
```

Here, we use 64-bit system for testing and set the `bin` directory to `PATH`, then some utilities such as `ncdump.exe` can be tested and used in **pbdNCDF4**.

With a correct `PATH`, one can use the R commands to install/build the **pbdNCDF4**:

Shell Command

```
### Under command mode, build and install the binary.
tar zxvf pbdNCDF4_0.1-3.tar.gz
R CMD INSTALL --build pbdNCDF4
R CMD INSTALL pbdNCDF4_0.1-3.zip
```

4.3. Detail Steps

The steps of building **pbdNCDF4** in Windwos are described next:

1. At configure time, I take `NETCDF4_ROOT_32` and `NETCDF4_ROOT_64` from environment variables, then set `NETCDF4_ROOT` according to architecture.

2. Before compile time, I check if the directories exists. If not, then I will set `NETCDF4_LINKED = FALSE` and feed a fake `a.c` to build a fake library in order to pass `R CMD check`.

3. At compile time, I save `NETCDF4_ROOT_32`, `NETCDF4_ROOT_64`, and `NETCDF4_ROOT` in `pbdNCDF4/etc/i386/Makeconf` and `pbdNCDF4/etc/x64/Makeconf`.

4. At installation time, both files will be copied to

   `C:/Program Files/R/R-3.0.1/library/pbdNCDF4/...`,

   with built `*.dll` and others. Then, `pbdNCDF4/R/windows/zzz.R` will be copied to `pbdNCDF4/R/zzz.R` to finish compiling R codes.

5. At load time, `.onLoad()` of `pbdNCDF4/R/windows/zzz.R` will obtain `NETCDF4_ROOT_32`, `NETCDF4_ROOT_64`, and `NETCDFR_ROOT` from system environment variables via `Sys.getenv()` first.

6. If environment variables were unavailable, `.onLoad()` will obtain them from corresponding `Makeconf`.

7. At run time, if none of `*.dll` were available or `NETCDF4_LINKED = FALSE`, then I would cast warnings. Note that any **pbdNCDF4** function evokes `.C()` function may cause crashes in this case.

In order to load **pbdNCDF4** correctly in windows, there are several ways can solve the problem:

- There are at least two ways to avoid dynamic file loading problems if any. Either change the `Makeconf` files to the personal/new installation of **netCDF-C** or change environment variables.

- There are also at least two ways to set environment variables in windows, either from system "Control Panel" or use batch file.

- Further, inside R, `Sys.setenv()` can be used to set environment variables in run time.

References

Chen WC, Ostrouchov G, Schmidt D, Patel P, Yu H (2012a). "pbdMPI: Programming with Big Data – Interface to MPI." R Package, URL http://cran.r-project.org/package=pbdMPI.

Chen WC, Ostrouchov G, Schmidt D, Patel P, Yu H (2012b). "A Quick Guide for the pbdMPI package." R Vignette, URL http://cran.r-project.org/package=pbdMPI.

HDF Group (2000-2010). "Hierarchical data format version 5." Software package, URL http://www.hdfgroup.org/HDF5.

NetCDF Group (2008). "Network Common Data Form." Software package, URL http://www.unidata.ucar.edu/software/netcdf/.

Ostrouchov G, Patel P, Chen WC, Schmidt D, Pierce D (2012). "pbdNCDF4: Programming with Big Data – Interface to Parallel Unidata NetCDF4 Format Data Files." R Package, URL http://cran.r-project.org/package=pbdNCDF4.

Pierce D (2012). "ncdf4: Interface to Unidata netCDF (version 4 or earlier) format data files." R package, URL http://CRAN.R-project.org/package=ncdf4.