

WRF/WPS ユーザーガイド

はじめに

これは管理人がWRF/WPS という気象解析予報モデルプログラム及びその初期化プログラムに関して、色々なトラブルに見舞われつつインストール&使用していった時の覚え書きです。まあソースからのコンパイル&インストールがほとんどなので、だいたいのUNIX系OSなら大丈夫かと。

現在の利用環境は以下の通り。

- * Vine Linux 4.2
- * Intel C/C++ Compiler 10.1
- * Intel Fortran Compiler 10.1

注意

このページでは、コマンドラインで実行するコード、ファイルに記述するコードなどが書かれています。コマンドラインで実行するコードについては、#で始まる場合はルート権限での実行とし、\$で始まる場合はユーザ権限で実行します。ファイルに記述するコードはbashの文法に則って記述します。コンパイラのインストールを除き、root権限でインストールするプログラムは/usr/local以下のエンタリにインストールしていきますが、後で混乱を避けるために専用ディレクトリを作成し、その中にインストールしていく方式をとっています。ユーザ権限でインストールするプログラムはユーザのホームディレクトリ直下にインストールします。ダウンロードしたファイルに対して作業するときは、/tmpで行います。ファイルの編集等はviで行っていますが、geditやemacs等々の普段使われているエディタで勿論OKです。適宜読み替えて下さい。

目次

WRF/WPS をインストール (pp.3)

- * 各種インストール前準備
- * Intel C/C++ Compiler 10.1
- * Intel Fortran Compiler 10.1
- * NetCDF 3.6.2
- * Szip 2.1
- * HDF 4.2r3
- * Udunits 1.12.4
- * G2clib 1.0.5
- * GrADS 2.0.a3
- * WRF 2.2.1
- * WPS 2.2.1
- * WRF2GrADS 2.0

WRF/WPS を使う (pp.18)

- * WPS
- * WRF
- * WRF2GrADS

参考情報：旧バージョンのインストール (pp.27)

- * Intel C/C++ Compiler 9.1
- * Intel Fortran Compiler 9.1
- * NCARG 4.4.1
- * NCL 4.4.1
- * NCL/NCARG 5.0.0
- * GrADS 1.9b4
- * WRFSI (WRF Ver2.1.2 以前に対応)
- * WRF Version 2.1.2

各種インストール前準備

はじめに、WRF/WPS を使用するためのプログラムをインストールするために、開発ツールやライブラリが必要となる。以降のインストールにおける依存性を解決するソフトウェアを apt-get でインストールしておく。

まずソースのコンパイルに必要なプログラムのインストール、その確認をしておく。gcc の関連するデベロッパが入っていないと icc の出力チェックでエラーが発生するという報告があるので、適宜インストール。

```
# apt-get install gcc gcc-*
# apt-get install readline readline-devel
# apt-get install gd gd-devel
```

HDF をインストールするために構文解析器生成プログラムをインストールする必要がある。byacc はパブリックドメイン yacc 構文解析器生成プログラム、bison は GNU 汎用構文解析器生成プログラム、flex はスキャナ（テキストパターン認識器）生成ツールである。各システムで若干名称が違う可能性があるため、該当する構文解析器系のプログラムを確認すること。

```
# apt-get install byacc
# apt-get install bison
# apt-get install flex
```

画像関係の開発用ライブラリその他を導入する。これらも適宜確認、インストールする。

```
# apt-get install libjpeg libjpeg-devel
# apt-get install libpng libpng-devel
# apt-get install zlib zlib-devel
# apt-get install freeglut freeglut-devel
```

g2clib のインストールには jasper(JPEG-2000)が必須となるが、jasper、jasper-devel とともに Vine Linux では apt から入手できる。しかし、ソースファイル中に tmpnam 関数を使用している箇所があり、これは mkstemp 関数を使うよう推奨されている。このため、jasper をインストールする際にはこの修正を行ってからインストールする必要がある。apt で jasper のソースを入手する。

```
# apt-get source jasper
```

apt-get の source オプションでダウンロードするのは SRPM というソース用 RPM である。これを通常の RPM のようにインストールすると、RPM ルート以下にソースファイルが展開される。SRPM インストール後は元のファイルを削除しても構わない。現時点でのバージョンは 1.900.1-0v12.1 である。

```
# rpm -ivh jasper-1.900.1-0v12.1.src.rpm
# rm -f jasper-1.900.1-0v12.1.src.rpm
```

RPM ルートに該当するディレクトリに移動する。RPM ルートは環境によって変化するが、概ね次の二つのどちらかである。

```
# cd /usr/src/{OS名}/SOURCES
// または、
# cd /root/rpm/SOURCES
```

ソースファイルを確認し、該当ソースの含まれる zip ファイルを解凍。

```
# unzip jasper-1.900.1.zip
# cd jasper-1.900.1
```

該当箇所を修正。

```
# vi src/libjasper/base/jas_stream.c
// ----- line: 368 -----
mkstemp(obj->pathname);    // tmpnam 関数を mkstemp 関数へリネーム
```

zip で圧縮し直し、rpm にまとめる。

```
# cd ../
# rm -f jasper-1.900.1.zip
# zip -r9 jasper-1.900.1.zip jasper-1.900.1/
# cd ../SPECS
# rpmbuild -bb jasper.spec
```

作成された rpm を、デベロッパも一緒にインストール。

```
# cd ../RPMS/i386
# rpm -ivh jasper-*.rpm
```

Intel C/C++ Compiler 10.1

まずはコンパイラのインストール。Intel の非商用ライセンス版コンパイラを取得。Free Non-Commercial Download の Compiler から C/C++ Compiler をダウンロードする。英語で非商用目的の使用かどうか、サポートを得ることはできないことへの了解が求められるが、どちらも Yes。登録画面に移り、メールアドレスと所在国を入力。そのアドレスにライセンスファイルが送られてくる。（ここでは/tmp 以下に一時保存）

次にソースファイルを/tmp 以下に解凍し、作成されたディレクトリに入る。

```
# cd /tmp
# tar zxvf l_cc_p_10.1.015.tar.gz
# cd l_cc_p_10.1.015
```

通常ならここでそのままインストールプログラムを実行すればいいが、Vine Linux の場合はそのまま実行するとインストール途中にエラーが発生しインストールできないので、インストールする際に修正しなければならない箇所がある。osdetect.sh の 218 行目の記述を修正する。9.1 まで問題のあった data/install_cc.sh は修正されているのに……。

```
# vi osdetect.sh

// ----- line: 218 -----
CMD=`type -p $c 2>&1` ; // type のオプション(P)を追加
```

これでインストールが可能になる。インストールプログラムを実行。

```
# ./install.sh
```

1→2 の順に選択し、ライセンスファイルのあるアドレスを記述する。/tmp/NCOM_L_CMP_CPP_*.lic 等。

標準インストールでデバッガをインストールしても良いが、カスタムインストールで本体だけをインストールしても構わない。インストール先はデフォルトの/opt としておく。

環境設定のパスを通す。

```
# vi ~/.bashrc

// ----- Add to End of File -----
### Intel C/C++ Compiler
./opt/intel/cc/10.1.015/bin/iccvars.sh
```

環境設定を有効にし、バージョンチェック、簡易なテストプログラムのコンパイル等が成功したら終了。

```
# ~/.bashrc
# icc -v
Version 10.1 // バージョンのチェック
```

Intel Fortran Compiler 10.1

C/C++コンパイラ同様、Free Non-Commercial Download から Intel Fortran Compiler をダウンロードする。質問内容は C/C++ と全く同じ。メールアドレスと国を記入し、ライセンスファイルを貰う。

ソースファイルを /tmp 以下に解凍し、作成されたディレクトリに入る。

```
# cd /tmp
# tar zxvf l_fc_c_9.1.037.tar.gz
# cd l_fc_c_9.1.037
```

Fortran コンパイラのインストールでも C/C++ コンパイラと同様のインストールエラーが発生する。修正箇所は全く同じ。osdetect.sh の 218 行目を修正する。

```
# vi osdetect.sh

// ----- line: 218 -----
CMD=`type -pP $c 2>&1` ; // type のオプション(P)を追加
```

インストールプログラムを実行。

```
# ./install.sh
```

1→2 の順に選択し、ライセンスファイルのあるアドレスを記述する。/tmp/NCOM_L_CMP_FOR_*.lic 等。

標準インストールでデバッガをインストールしても良いが、カスタムインストールで本体だけをインストールしても構わない。インストール先はデフォルトの/opt としておく。

環境設定のパスを通す。

```
# vi ~/.bashrc

// ----- Add to End of File -----
### Intel Fortran Compiler
./opt/intel/fc/10.1.015/bin/ifortvars.sh
```

環境設定を有効にし、テストが成功したら終了。

```
# ~/.bashrc
# ifort -v
Version 10.1 // バージョンのチェック
```

NetCDF 3.6.2

NetCDFはNetwork Common Data Formの略である。WRFその他のコンパイルにはこのライブラリが必須となるので、早めにインストールしておく。NetCDF Downloadsにアクセスし、最新バージョンをダウンロードする。解凍後、ソースディレクトリへ移動。

```
# cd /tmp
# tar zxvf netcdf.tar.gz
# cd netcdf-3.6.2
```

ソースをコンパイルする。インストール用スクリプト（ここではinstallというファイル名とする）をエディタで記述する。

```
# vi install

// ----- Edit New File -----
#!/bin/bash

export CC=icc
export CXX=icpc
export FC=ifort
export F90=ifort
export CFLAGS="-O2"
export FFLAGS="-O -mp"
export CPPFLAGS="-DNDEBUG -DpgiFortran"

./configure --prefix=/usr/local/netcdf-3.6.2
make test
make install
```

インストール先のディレクトリを作成し、インストール用スクリプトを実行する。

```
# mkdir -p /usr/local/netcdf-3.6.2
# sh install
```

インストール完了後、環境設定を記述する。

```
# vi ~/.bashrc

// ----- Add to End of File -----
### NetCDF
export NETCDF="/usr/local/netcdf-3.6.2"
export NCHOME="${NETCDF}"
export NETCDFHOME="${NETCDF}"
export PATH="${NETCDF}/bin:${PATH}"
export LD_LIBRARY_PATH="${NETCDF}/lib:${LD_LIBRARY_PATH}"
export MANPATH="${NETCDF}/man:${MANPATH}"
```

環境設定後、設定を有効にする。

```
# . ~/.bashrc
```

Szip 2.1

SZIP は HDF、GrADS のオプションインストールの時に必要となる。次の HDF と併せてインストールしておく。ダウンロードは HDF の FTP サイトから可能。

```
# cd /tmp
# tar zxvf szip-2.1.tar.gz
# cd szip-2.1
```

インストール用スクリプトを記述。

```
# vi install

// ----- Edit New File -----
#!/bin/bash

export CC=icc
export CXX=icpc
export FC=ifort
export F90=ifort
export CFLAGS="-O2"
export FFLAGS="-O -mp"

./configure --prefix=/usr/local/szip-2.1
make
make install
```

インストール先ディレクトリを作成し、スクリプトを実行する。

```
# mkdir -p /usr/local/szip-2.1
# sh install
```

次に環境設定を行う。

```
# vi ~/.bashrc

// ----- Add to End of File -----
### Szip
export SZIP="/usr/local/szip-2.1"
export LD_LIBRARY_PATH="${SZIP}/lib:${LD_LIBRARY_PATH}"
```

設定後、設定を有効にする。

```
# . ~/.bashrc
```

HDF 4.2r3

HDFは Hierarchical Data Format という NetCDF と同様データフォーマットライブラリの一つである。GrADS のオプションで HDF のファイルを取扱うようにする時に必要。HDF4 と HDF5 があるが、現時点では HDF4 のライブラリを必要とする。

HDF Home Page から HDF4 のダウンロードページに行き、Linux 用のソースファイルをダウンロードする。解凍後、ソースディレクトリに移る。

```
# cd /tmp
# tar zxvf HDF4.2r3.tar.gz
# cd HDF4.2r3
```

インストール用スクリプトを記述する。

```
# vi install

// ----- Edit New File -----
#!/bin/bash

export CC=icc
export CXX=icpc
export F77=ifort
export FC=ifort

./configure --prefix=/usr/local/hdf-4.2r3 --with-szlib=/usr/local/szip-2.1 --disable-netcdf
make
make install
```

インストール先ディレクトリを作成し、スクリプトを実行する。

```
# mkdir -p /usr/local/hdf-4.2r3
# sh install
```

インストール後、bin 以下に nc*実行ファイルが置かれるが、これは netcdf のインストールで既にインストール済みである。パスの通し方によっては後々 nc*を実行したときに netcdf のものではなくこちらが実行されてしまい、上手く動かなくなる。old ディレクトリを作成し、実行ファイルを移しておく。

```
# cd /usr/local/hdf-4.2r3
# mkdir old
# mv bin/nc* old
```

また、次の ncarg インストール時に HDF のインクルードファイルの書き換えが必要になる。これをバックアップし、修正しておく。

```
# cp include/hdfi.h old
# vi include/hdfi.h

// ----- line: 962-965 -----
// rewrite
#include <sys/types.h>
#include <sys/stat.h>
#include <io.h>
#include <conio.h>
// to
```

```
#include <sys/types.h>
#include <sys/stat.h>
#include <sys/io.h>
#include <ncurses.h>
```

次に環境設定を行う。

```
# vi ~/.bashrc

// ----- Add to End of File -----
### HDF
export HDF="/usr/local/hdf-4.2r3"
export PATH="${HDF}/bin:${PATH}"
export LD_LIBRARY_PATH="${HDF}/lib:${LD_LIBRARY_PATH}"
```

設定後、設定を有効にする。

```
# . ~/.bashrc
```

Udunits 1.12.4

GrADS のインストール時にこのライブラリが必要となる。ダウンロードは Unidata から最新バージョンを選択。解凍し、ディレクトリへ移動。

```
# cd /tmp
# tar zxvf udunits-1.12.4.tar.Z
# cd udunits-1.12.4/src
```

インストール用スクリプトを記述。

```
# vi install

// ----- Edit New File -----
#!/bin/bash

export CC=icc
export CXX=icpc
export FC=ifort
export F90=ifort
export CFLAGS="-O2"
export FFLAGS="-O"
export CPPFLAGS="-DNDEBUG -DpgiFortran"

./configure --prefix=/usr/local/udunits-1.12.4
make
make install
```

インストール先ディレクトリを作成し、スクリプトを実行する。

```
# mkdir -p /usr/local/udunits-1.12.4
# sh install
```

次に環境設定を行う。

```
### ~/.bashrc

### Udunits
export UDUNITS="/usr/local/udunits-1.12.4"
export PATH="${UDUNITS}/bin:${PATH}"
export LD_LIBRARY_PATH="${UDUNITS}/lib:${LD_LIBRARY_PATH}"
```

設定後、設定を有効にする。

```
# . ~/.bashrc
```

G2clib 1.0.5

GrADS で grib2 ファイルを扱うために必要なライブラリ。ダウンロードは NCEP の GRIB2 ページから選択。

```
# cd /tmp
# tar xvf g2clib-1.0.5.tar
```

makefile のコンパイラ設定を修正。

```
# cd /tmp/g2clib-1.0.5
# vi makefile

// ----- line: 24-26 -----
CC=icc
LIB=libgrib2c.a
ARFLAGS=
```

make でコンパイル。成功すると、libgrib2c.a ファイルが作成される。/usr/local に移動させ、所有者を変更。

```
# make
# cd ..
# mv g2clib-1.0.5 /usr/local
# chown 0:0 -R /usr/local/g2clib-1.0.5
```

次に環境設定を行う。

```
# vi ~/.bashrc

// ----- Add to End of File -----
### G2clib
export G2CLIB="/usr/local/g2clib-1.0.5"
export JASPERLIB="/usr/lib"
export JASPERINC="/usr/include"
```

設定後、設定を有効にする。

```
# . ~/.bashrc
```

GrADS 2.0.a3

正式名称は Grid Analysis and Display System。解析データの可視化ソフトである。

各種リンカファイルに対し、`/usr/lib` にシンボリックリンクを貼る。

```
# cd /usr/lib
# ln -s /usr/local/*/lib/*.a .
# ln -s /usr/local/*/*.a .
```

Downloading GrADS Software からソースファイルをダウンロード。解凍し、ディレクトリへ移動。

```
# cd /tmp
# tar zxvf grads-src-2.0.a3.tar.gz
# cd grads-2.0.a3
```

インストール用スクリプトを記述する。

```
# vi install

// ----- Edit New File -----
#!/bin/bash

export CC=icc
export CXX=icpc
export FC=ifort
export F90=ifort
export CFLAGS="-O2"
export FFLAGS="-O -mp"
export CPPFLAGS="-I/usr/local/netcdf-3.6.2/include -I/usr/local/zip-2.1/include -I/usr/local/hdf-4.2r3/include -I/
usr/local/udunits-1.12.4/include -I/usr/local/g2clib-1.0.5"
export LIBS="-lnetcdf -lsz"

{
./configure --prefix=/usr/local/grads-2.0.a3
make
make install
} >& make-output & tail -f make-output
```

インストール先ディレクトリを作成後、インストール用スクリプトを実行。ドキュメント等をソースディレクトリからコピーする。

```
# mkdir -p /usr/local/grads-2.0.a3
# sh install
# cp -a data doc etc lib m4 /usrlocal/grads-2.0.a3
```

次にスクリプトファイルをダウンロード、インストールする。`wget` を使用し、FTP サイトからデータを取得する。

```
# cd /usr/local/grads-2.0.a3
# mkdir scripts
# cd scripts
# wget ftp://grads.iges.org/grads/scripts/*
```

インストールが完了したら、環境設定を記述しておく。

```
# vi ~/.bashrc

// ----- Add to End of File -----
### GrADS
export GRADS="/usr/local/grads-2.0.a3"
export PATH="${GRADS}/bin:${PATH}"
export GASCRP="${GRADS}/scripts"
export GADDIR="${GRADS}/data"
```

環境設定を有効にしたら終了。

```
# . ~/.bashrc
```

WRF 2.2.1

Weather Research and Forecasting Model（気象解析予報モデル）。気象シミュレーション解析プログラム
の一種。

WRF はユーザー権限でインストールするため、差し支えなければ root の .bashrc を一般ユーザの .bashrc
にコピーする。root と一般ユーザで設定が異なるならば、該当箇所をエディタでコピーすること。ファ
イルを丸ごとコピーした場合は所有者権限の変更も行っておく。

```
# cd /home/{USER_NAME}
# cp /root/.bashrc .
# chown {USER_NAME}:{GROUP_NAME} .bashrc
# su - {USER_NAME}
```

以降、一般ユーザ権限での作業となる。WRF Download から WRF model tar file をダウンロード。mkdir
でホームディレクトリ下に WRF 用のディレクトリを作成する。

```
$ mkdir ~/WRF
$ cd ~/WRF
$ tar zxvf /tmp/WRFV2.2.1.TAR.gz
$ cd WRFV2
```

WRFV2/arch/configure.defaults を編集する。「AMD x86_64 Intel xeon i686 ia32 Xeon Linux, ifort compiler
(single-threaded, no nesting)」の設定を基にする。該当行は line:6810-6937 となる。変更する箇所のみを
記載する。

```
$ vi arch/configure.defaults
// ----- line: 6810-6937 -----
CC      =   icc
CPP     =   ifort -E -free
```

設定の変更が終了したら configure を実行する。画面中設定は 7 を採用。configure が終わったらコンパ
イル。

```
$ ./configure
$ ./compile em_real
```

run ディレクトリに ndown.exe、real.exe、wrf.exe が作成されていればコンパイル成功。

WPS 2.2.1

WRF Preprocessing System の略。WRFSI に代わり、WRF Version2.2 から WPS が前処理プログラムとなる。WRFSI から大きく変更された点は、WPS の前に WRF をインストールする必要があること、GUI だけでなく CUI でも処理が可能になったこと等。ここでは簡単のために、WPS の GUI をインストールしない。

WRF Download のページから WPS をダウンロードする。また地理データとして `geog.tar.gz` もダウンロードすること。WRFSI のものと若干異なるので、WPS 用に新しくインストールする必要がある。

```
$ cd ~/WRF
$ tar zxvf /tmp/WPSV2.2.TAR.gz
$ tar zxvf /tmp/geog.tar.gz
$ mv geog GEOG
$ cd WPS
```

WRF 同様、`arch` ディレクトリに `configure.defaults` があるので該当箇所を修正。変更点のみ記載する。

```
$ vi arch/configure.defaults

// ----- line: 253-265 -----
CC          =      icc
SCC         =      icc
CPP         =      ifort -E -free -C -P
```

`configure` で設定。設定モードは 6 を選択する。設定後、コンパイルを実行。コンパイルには、WPS のルートディレクトリと同階層に "WRFV2" というディレクトリが存在し、その中に WRF がインストールされている必要がある。

```
$ ./configure
$ ./compile
```

`geogrid.exe`、`ungrib.exe`、`metgrid.exe` の 3 つが出力されていればコンパイル完了。

WRF2GrADS 2.0

WRFで出力されたファイルをGrADSで可視化する。この出力がうまくいけば、一通りの解析とその表示が可能になる。WRF Downloadの「WRF Post-Processing Software」から「Convert WRF model output in netCDF to GrADS format」にアクセスし、WRF2GrADSをダウンロードする。解凍後、ディレクトリへ移動。

```
$ cd ~/WRF
$ tar zxvf /tmp/wrf2grads.tar.gz
$ cd WRF2GrADS
```

インストールの設定を行う。Makefileの「linux flag (INTEL)」の箇所を修正。コメントアウトを解除し、ライブラリファイル、インクルードファイルへのパスを修正する。CPPやフラグも修正しておく。

```
$ vi Makefile

// ----- line: 29-34 -----
LIBNETCDF = -L/usr/local/netcdf-3.6.2/lib -lnetcdf -lm
INCLUDE = -I/usr/local/netcdf-3.6.2/include -I./
FC = ifort
FCFLAGS = -C -FR
CPP = ifort -E -free
CPPFLAGS = -I. -C -DRECL1
```

また、このままコンパイルすると「fortrtl: severe (193): Run-Time Check Failure. The variable 'module_wrf_to_grads_util_mp_time_calc_\${HOURL}' is being used without being defined」という実行時エラーを起こす。これは初期化されていない変数を使用しているというエラー。詳細は「インテル(R) Visual Fortran コンパイラ 9.1 Windows* 版 リリースノート」の変更点のうちの該当箇所を参照（Linux版のリリースノートでは9.0、9.1ともにそのような変更点は記載されていない）。これを回避するために、module_wrf_to_grads_util.Fの1066-1068行目を修正する。

```
$ vi module_wrf_to_grads_util.F

// ----- line: 1066-1068 -----
// rewrite
hour1=hours
mins1=minutes
elseif ( it == 2) then
// to
elseif ( it == 2) then
hour1=hours
mins1=minutes
```

アルゴリズムやインデントの状態を考慮すると、書き順ミスによる単純なバグ?のようである。

修正が終わったらコンパイルを実行しても良いが、他のソースファイルまで実行可能になっているので色分けをしているときに見づらくなる。属性を変更してからコンパイル。

```
$ chmod 644 *
$ make
```

wrf_to_grads 実行ファイルが作成されていればコンパイル終了。

WPS

WPS を使うには FNL データを必要とする。WRF Download にある「WRF Preprocessing System test data」をダウンロードし、適当なディレクトリを作成して解凍。

```
$ mkdir -p ~/WRF/FNL/sample
$ cd ~/WRF/FNL/sample
$ tar zxvf /tmp/avn_data.tar.gz
$ cd ~/WRF/WPS
```

WPS は「geogrid.exe」→「ungrib.exe」→「metgrid.exe」の順でプログラムを走らせる。これらのプログラムの設定には、namelist.wps を一貫して使用する。

FNL データにアクセスするためには、シンボリックリンクを設定する。シンボリックリンクを貼るための C シェルスクリプトが link_grib.csh として提供されているのでこれを利用する。

```
$ ./link_grib.csh ../FNL/sample/*
```

処理する変数テーブルに対してシンボリックリンクを貼る。ここでは NCEP2 を使用する。

```
$ ln -sf ungrib/Variable_Tables/Vtable.NCEP2 Vtable
```

namelist.wps を以下のように編集する。初期設定からの変更部分を赤色で記載。

```
$ vi namelist.wps
// ----- Edit All Line -----
&share
  wrf_core = 'ARW',
  max_dom = 1,
  start_date = '2000-01-24_12:00:00',
  end_date = '2000-01-25_00:00:00',
  interval_seconds = 21600,
  io_form_geogrid = 2,
/

&geogrid
  parent_id = 1,
  parent_grid_ratio = 1,
  i_parent_start = 1,
  j_parent_start = 1,
  e_we = 74,
  e_sn = 61,
  geog_data_res = '10m',
  dx = 30000,
  dy = 30000,
  map_proj = 'lambert',
  ref_lat = 34.83,
  ref_lon = -81.03,
  truelat1 = 30.0,
  truelat2 = 60.0,
  stand_lon = -90.0,
  geog_data_path = './GEOG'
/

&ungrib
```

```
out_format = 'WPS',
prefix = 'NCEP2',
/

&metgrid
fg_name = 'NCEP2',
io_form_metgrid = 2,
/

&mod_levs
press_pa = 201300 , 200100 , 100000 ,
           95000 , 90000 ,
           85000 , 80000 ,
           75000 , 70000 ,
           65000 , 60000 ,
           55000 , 50000 ,
           45000 , 40000 ,
           35000 , 30000 ,
           25000 , 20000 ,
           15000 , 10000 ,
           5000 , 1000
/
```

編集が終了したら、順序通りにプログラムを走らせる。

```
$ ./geogrid.exe
$ ./ungrib.exe
$ ./metgrid.exe
```

出力に成功したら、met_em.*.nc というファイル群が作成される。WRFではこのファイル群を使用する。

WRF

WRFはrunディレクトリでプログラムを実行する。ディレクトリに移動し、WPSで作成したファイルをこのディレクトリに置く。

```
$ cd ~/WRF/WRFV2/run
$ mv ~/WRF/WPS/met_em* .
```

WRFの設定はnamelist.inputで行う。namelist.inputに関する詳しい情報は、README.namelistを参照すること。以下、設定ファイルの情報を記す。初期設定からの変更点は赤で記載。

```
$ vi namelist.input

// ----- Edit All Line -----
&time_control
run_days           = 0,
run_hours          = 12,
run_minutes        = 0,
run_seconds        = 0,
start_year         = 2000, 2000, 2000,
start_month        = 01, 01, 01,
start_day          = 24, 24, 24,
start_hour         = 12, 12, 12,
start_minute       = 00, 00, 00,
start_second       = 00, 00, 00,
end_year           = 2000, 2000, 2000,
end_month          = 01, 01, 01,
end_day            = 25, 25, 25,
end_hour           = 00, 12, 12,
end_minute         = 00, 00, 00,
end_second         = 00, 00, 00,
interval_seconds   = 21600
input_from_file    = .true.,.false.,.false.,
history_interval   = 180, 60, 60,
frames_per_outfile = 1000, 1000, 1000,
restart            = .false.,
restart_interval    = 5000,
io_form_history     = 2
io_form_restart     = 2
io_form_input       = 2
io_form_boundary    = 2
debug_level        = 0
/

&domains
time_step          = 180,
time_step_fract_num = 0,
time_step_fract_den = 1,
max_dom            = 1,
s_we               = 1, 1, 1,
e_we               = 74, 112, 94,
s_sn               = 1, 1, 1,
e_sn               = 61, 97, 91,
s_vert             = 1, 1, 1,
e_vert             = 27, 28, 28,
```

```

num_metgrid_levels      = 27
dx                      = 30000, 3333, 1111,
dy                      = 30000, 3333, 1111,
grid_id                 = 1,  2,  3,
parent_id               = 0,  1,  2,
i_parent_start          = 0,  30, 30,
j_parent_start          = 0,  20, 30,
parent_grid_ratio       = 1,  3,  3,
parent_time_step_ratio  = 1,  3,  3,
feedback                = 1,
smooth_option           = 0
/

&physics
mp_physics              = 3,  3,  3,
ra_lw_physics           = 1,  1,  1,
ra_sw_physics           = 1,  1,  1,
radt                    = 30, 30, 30,
sf_sfclay_physics      = 1,  1,  1,
sf_surface_physics     = 1,  1,  1,
bl_pbl_physics         = 1,  1,  1,
bldt                    = 0,  0,  0,
cu_physics              = 1,  1,  0,
cudt                    = 5,  5,  5,
isfflx                  = 1,
ifsnow                  = 0,
icloud                  = 1,
surface_input_source    = 1,
num_soil_layers         = 5,
ucmcall                 = 0,
mp_zero_out             = 0,
maxiens                 = 1,
maxens                  = 3,
maxens2                 = 3,
maxens3                 = 16,
ensdim                  = 144,
/

&fdda
/

&dynamics
w_damping               = 0,
diff_opt                = 1,
km_opt                  = 4,
diff_6th_opt           = 0,
diff_6th_factor        = 0.12,
base_temp               = 290.
damp_opt                = 0,
zdamp                  = 5000., 5000., 5000.,
dampcoef                = 0.01, 0.01, 0.01
khdif                  = 0,  0,  0,
kvdif                  = 0,  0,  0,
non_hydrostatic         = .true., .true., .true.,
pd_moist                = .false., .false., .false.,

```

```

pd_scalar          = .false., .false., .false.,
/

&bdy_control
spec_bdy_width     = 5,
spec_zone          = 1,
relax_zone         = 4,
specified          = .true., .false.,.false.,
nested             = .false., .true., .true.,
/

&grib2
/

&namelist_quilt
nio_tasks_per_group = 0,
nio_groups = 1,
/

```

設定が完了したら、初期値・境界値データを作成する。Vine Linux ではスタックサイズに 8192KB 制限があるので、real.exe を実行する前にスタックサイズを無制限にしておく。

```

$ ulimit -s unlimited
$ ./real.exe

```

初期値・境界値データが wrfbdy、wrfinput というファイルで作成される。この後、wrf.exe を実行してシミュレーションを行う。

```

$ ulimit -s unlimited
$ ./wrf.exe

```

計算範囲やタイムステップによって実行時間は大きく変化する。シミュレーションが終了すると、wrfout ファイルが作成される。

WRF2GrADS

WRFで作成されたファイルはNetCDFのファイル形式をとっている。この出力ファイルをGrADSで表示できるように変換する。WRFで作成したファイルをWRF2GrADSディレクトリに移動する。

```
$ cd ~/WRF/WRF2GrADS
$ mv ~/WRF/WRFV2/run/wrfout* .
```

GrADS変換プログラムを走らせるために、control_fileを編集する。

```
$ vi control_file

// ----- Edit All Line -----
-2          ! number of times to put in GrADS file, negative means ignore the times
2000-01-24_12:00:00
2000-01-24_18:00:00
2000-01-25_00:00:00
end_of_time_list

                ! 3D variable list for GrADS file
                ! indent one space to skip
U            ! U Component of wind
V            ! V Component of wind
  UMET       ! U Component of wind - rotated (diagnostic)
  VMET       ! V Component of wind - rotated (diagnostic)
W            ! W Component of wind
  THETA      ! Theta
  TK         ! Temperature in K
TC           ! Temperature in C
  TKE        ! TURBULENCE KINETIC ENERGY
P            ! Pressure (hPa)
Z            ! Height (m)
  QVAPOR     ! Vapor
  QCLOUD     ! Cloud Water
  QRAIN      ! Rain Water
  QICE       ! -
  QSNOW      ! -
  QGRAUP     ! -
  TKE_MYJ    ! TKE FROM MELLOR-YAMADA-JANJIC
PB           ! BASE STATE PRESSURE AT HALF LEVEL
TD           ! Dewpoint Temperature (diagnostic)
RH           ! Relative Humidity (diagnostic)
                ! Soil variables
TSLB        ! SOIL TEMPERATURE
SMOIS       ! SOIL MOISTURE
                ! below a list of fields from SI static and input files
LANDUSEF    ! -
SOILCTOP    ! -
SOILCBOT    ! -
SPECHUMD    ! -
GREEN12M    ! -
ALBDO12M    ! -
end_of_3dvar_list
ACSNOM      ! ACCUMULATED MELTED SNOW
ACSNOW      ! ACCUMULATED SNOW
AKHS        ! SFC EXCH COEFF FOR HEAT
```

AKMS ! SFC EXCH COEFF FOR MOMENTUM
 CANWAT ! CANOPY WATER
 GLW ! DOWNWARD LONG WAVE FLUX AT GROUND SURFACE
 GSW ! DOWNWARD SHORT WAVE FLUX AT GROUND SURFACE
 HFX ! UPWARD HEAT FLUX AT THE SURFACE
 HGT ! Terrain Height
 IVGTYP ! VEGETATION TYPE
 ISLTYP ! SOIL TYPE
 LU_INDEX ! LAND USE CATEGORY
 MAPFAC_M ! Map scale factor on mass grid
 MU ! Perturbation dry air mass in column
 MUB ! base state dry air mass in column
 MU0 ! initial dry mass in column
 Q2 ! QV at 2 M
 QFX ! UPWARD MOISTURE FLUX AT THE SURFACE
 RAINC ! ACCUMULATED TOTAL CUMULUS PRECIPITATION
 RAINCV ! TIME-STEP CUMULUS PRECIPITATION
 RAINNC ! ACCUMULATED TOTAL GRID SCALE PRECIPITATION
 SFROFF ! SURFACE RUNOFF
 slvl ! sea level pressure
 SMSTAV ! MOISTURE VARIBILITY
 SNOW ! SNOW WATER EQUIVALENT
 SNOWC ! FLAG INDICATING SNOW COVERAGE (1 FOR SNOW COVER)
 SST ! SEA SURFACE TEMPERATURE
 T2 ! TEMP at 2 M
 TH2 ! POT TEMP at 2 M
 TMN ! SOIL TEMPERATURE AT LOWER BOUNDARY
 TSK ! SURFACE SKIN TEMPERATURE
 U10 ! U at 10 M
 U10M ! U at 10 M - rotated
 V10 ! V at 10 M
 V10M ! V at 10 M - rotated
 UDROFF ! UNDERGROUND RUNOFF
 VEGFRA ! VEGETATION FRACTION
 WEASD ! WATER EQUIVALENT OF ACCUMULATED SNOW
 XLAT ! LATITUDE, SOUTH IS NEGATIVE
 XLONG ! LONGITUDE, WEST IS NEGATIVE
 XLAND ! LAND MASK (1 FOR LAND, 2 FOR WATER)
 ! below a list of fields from SI static and input files
 ALBBCK ! -
 LANDMASK ! -
 PMSL ! -
 SLOPECAT ! -
 SHDMAX ! -
 SHDMIN ! -
 SNOALB ! -
 SOILHGT ! -
 ST000010 ! -
 ST010040 ! -
 ST040100 ! -
 ST100200 ! -
 SM000010 ! -
 SM010040 ! -
 SM040100 ! -
 SM100200 ! -

```

TOPOSTDV      ! -
TOPOSLPX      ! -
TOPOSLPY      ! -
XICE          ! -
end_of_2dvar_list

                ! All list of files to read here
                ! Indent not to read
                ! Full path OK

/DATA/wrfstatic_d01
/DATA/wrfstatic_d02
/DATA/wrf_real_input_em.d01.2000-01-24_12:00:00
/DATA/wrf_real_input_em.d02.2000-01-24_12:00:00
/DATA/real/wrfinput_d01

./wrfout_d01_2000-01-24_12:00:00

/DATA/b_wave/wrfout_d01_0001-01-01_00:00:00
/DATA/grav2d_x/wrfout_d01_0001-01-01_00:00:00
/DATA/hill2d_x/wrfout_d01_0001-01-01_00:00:00
/DATA/quarter_ss/wrfout_d01_0001-01-01_00:00:00
/DATA/squall2d_x/wrfout_d01_0001-01-01_00:00:00
/DATA/squall2d_y/wrfout_d01_0001-01-01_00:00:00
end_of_file_list

                ! Now we check to see what to do with the data
real          ! real (input/output) / ideal / static
1            ! 0=no map background in grads, 1=map background in grads
-1          ! specify grads vertical grid
                ! 0=cartesian,
                ! -1=interp to z from lowest h
                ! 1 list levels (either height in km, or pressure in mb)
1000.0
950.0
900.0
850.0
800.0
750.0
700.0
650.0
600.0
550.0
500.0
450.0
400.0
350.0
300.0
250.0
200.0
150.0
100.0

```

編集が終了したら、変換プログラムを走らせる。

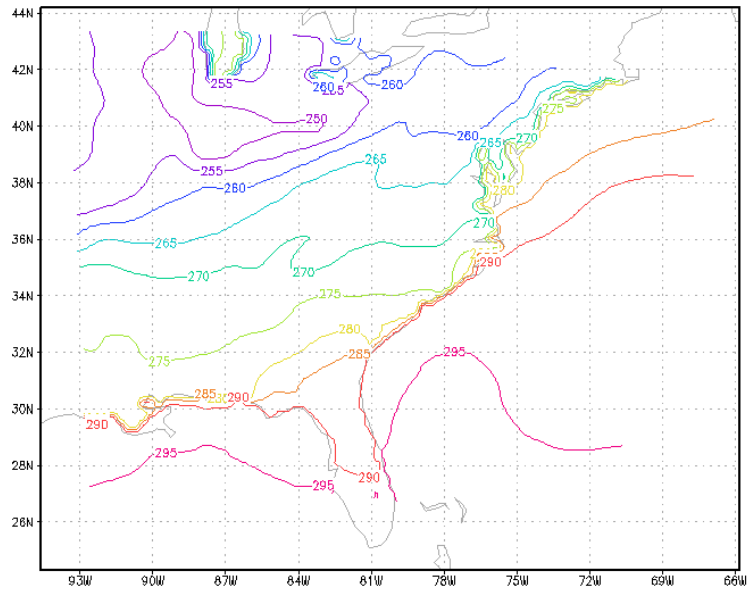
```
$ ./wrf_to_grads control_file sample
```

sample は出力名となる。出力が完了すると、GrADS形式の sample.cti と sample.dat が作成される。

GrADS を起動し、出力を確認する。

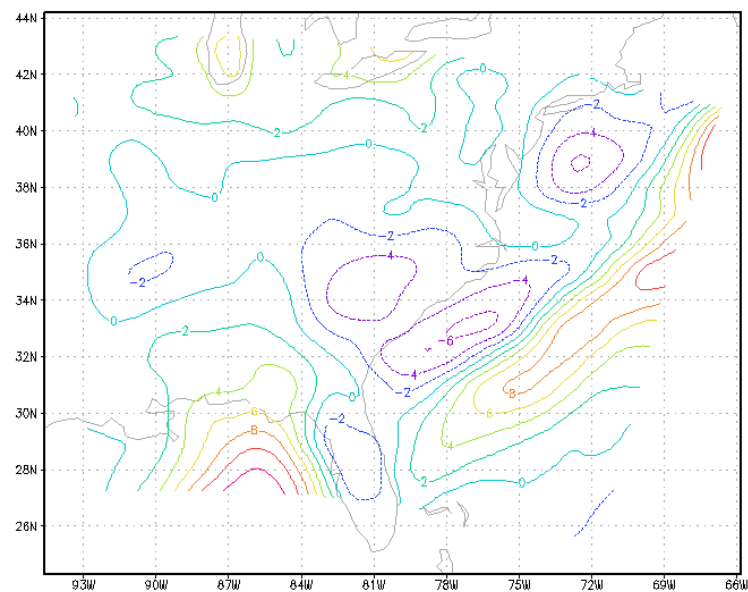
```
$ grads  
ga-> open sample.ctl  
ga-> q file  
ga-> d sst  
ga-> c  
ga-> d u10m
```

設定通りに行うと、以下のような画像が出力される。



GrADS: COLA/IGES

2008-05-22-17:15



GrADS: COLA/IGES

2008-05-22-17:13

Intel C/C++ Compiler 9.1

まずはコンパイラのインストール。Intel の非商用ライセンス版コンパイラを取得。Free Non-Commercial Download の Compiler から C/C++ Compiler をダウンロードする。英語で非商用目的の使用かどうか、サポートを得ることはできないことへの了解が求められるが、どちらも Yes。登録画面に移り、メールアドレスと所在国を入力。そのアドレスにライセンスファイルが送られてくる。ライセンスファイルが送られてきたら、/usr/local 以下にディレクトリを作ってライセンスファイルを置く。

```
# mkdir -p /usr/local/intel/licenses  
# cp /tmp/NCOM_L_CMP_CPP_*.lic /usr/local/intel/licenses
```

次にソースファイルを/tmp 以下に解凍し、作成されたディレクトリに入る。

```
# cd /tmp  
# tar zxvf l_cc_c_9.1.043.tar.gz  
# cd l_cc_c_9.1.043
```

通常ならここでそのままインストールプログラムを実行すればいいが、Vine Linux 3.x の場合はそのまま実行するとインストール途中でエラーが発生しインストールできないので、インストールする際に修正しなければならない箇所がある。data 以下にある install_cc.sh の 1456 行目の記述を修正する。

```
### data/install_cc.sh line:1456  
CMD='type -pP $c 2>&1' ; # type のオプションを追加
```

これでインストールが可能になる。インストールプログラムを実行。

```
# ./install.sh
```

1→2 の順に選択し、ライセンスファイルのあるアドレスを記述する。
/usr/local/intel/licenses/NCOM_L_CMP_CPP_*.lic 等。

標準インストールでデバッグをインストールしても良いが、カスタムインストールで本体だけをインストールしても構わない。インストール先は/usr/local/intel/cc/9.1.043 としておく。

インストールが完了したら、ライセンスファイルへのシンボリックリンクを貼っておく。

```
# cd /usr/local/intel/cc/9.1.043/licenses  
# ln -s /usr/local/intel/licenses/NCOM_L_CMP_CPP_*.lic
```

環境設定のパスを通す。

```
### ~/.bashrc  
  
### Intel C/C++ Compiler 9.1.043  
./usr/local/intel/cc/9.1.043/bin/iccvars.sh
```

環境設定を有効にし、テストが成功したら終了。

```
# . ~/.bashrc
```

Intel Fortran Compiler 9.1

C/C++コンパイラ同様、Free Non-Commercial Download から Intel Fortran Compiler をダウンロードする。質問内容は C/C++ と全く同じ。メールアドレスと国を記入し、ライセンスファイルを貰う。ライセンスファイルは同じ場所に置いておく。

```
# cp /tmp/NCOM_L_CMP_FOR_*.lic /usr/local/intel/licenses
```

ソースファイルを /tmp 以下に解凍し、作成されたディレクトリに入る。

```
# cd /tmp
# tar zxvf l_fc_c_9.1.037.tar.gz
# cd l_fc_c_9.1.037
```

Fortran コンパイラのインストールでも C/C++ コンパイラと同様のインストールエラーが発生する。修正箇所は全く同じ。data 以下の install_fc.sh の 1456 行目を修正する。

```
### data/install_fc.sh line:1456
CMD=`type -pP $c 2>&1` ; # type のオプションを追加
```

インストールプログラムを実行。

```
# ./install.sh
```

1→2 の順に選択し、ライセンスファイルのあるアドレスを記述する。
/usr/local/intel/licenses/NCOM_L_CMP_FOR_*.lic 等。

標準インストールでデバッガをインストールしても良いが、カスタムインストールで本体だけをインストールしても構わない。インストール先は /usr/local/intel/cc/9.1.043 としておく。

インストールが完了したら、ライセンスファイルへのシンボリックリンクを貼っておく。

```
# cd /usr/local/intel/fc/9.1.037/licenses
# ln -s /usr/local/intel/licenses/NCOM_L_CMP_FOR_*.lic
```

環境設定のパスを通す。

```
### ~/.bashrc
### Intel Fortran Compiler 9.1.037
. /usr/local/intel/fc/9.1.037/bin/ifortvars.sh
```

環境設定を有効にし、テストが成功したら終了。

```
# . ~/.bashrc
```

NCAR Graphics 4.4.1

NCAR Graphics のインストール。まず、HDF インクルードファイルにあった ncurses がインストールされている必要がある。

```
# apt-get install ncurses ncurses-devel
```

Download NCAR Graphics にアクセスし、ページ中程の「I have read and hereby accept the GNU General Public License.」にチェックする。その他は記入しなくても構わない。ページ下の「Submit」でダウンロードページに移るので、そこから最新のソースファイルを取得する。2006年10月現在は4.4.1。解凍後、ソースディレクトリに移る。

```
# cd /tmp
# tar zxvf ncarg-4.4.1.src.tar.gz
# cd ncarg-4.4.1
```

インストールファイルを修正する必要があるので、適宜修正。

```
### ./ncarview/src/lib/libncarg_ras/hdf.c line:67-68
--- from
#include <hdf/hdf.h>
#include <hdf/dfgr.h>
--- to
#include <hdf.h>
#include <dfgr.h>
--- End ---
```

./config/LINUX.INTEL の Fortran コマンド ifc は古いコマンド名。現在の ifort に修正しておく。修正後、LINUX ファイルにコピー。

```
### ./config/LINUX.INTEL line:31
#define FCompiler ifort -mp -Vaxlib
```

```
# cp ./config/LINUX.INTEL ./config/LINUX
```

インストール先ディレクトリを作成後、設定用スクリプトを実行する。インストール先ディレクトリ指定は/usr/local/ncarg-4.4.1/*とする。HDF サポートは Yes にし、ライブラリのサーチパスは「/usr/X11R6/lib /usr/local/hdf-4.2r1/lib」 「/usr/X11R6/include /usr/local/hdf-4.2r1/include」とする。設定後、コンパイル実行。

```
# mkdir /usr/local/ncarg-4.4.1
# ./Configure
# make Everything >& make-output
```

次に高解像度海岸線データをインストールする。Rainer Feistel にアクセスし、ページ中程にある rangs*.zip、gshhs*.zip を全てダウンロードする。インストール先ディレクトリを作成し、インストール。

```
# mkdir /usr/local/ncarg-4.4.1/rangs
# cd /tmp
# for i in 0 1 2 3 4;
> do unzip rangs($i).zip;
> unzip gshhs($i).zip;
> done;
> mv *.rim *.cat *.cel /usr/local/ncarg-4.4.1/rangs
```

インストール後、環境設定を行う。

```
### ~/.bashrc

### NCARG 4.4.1
export NCARG="/usr/local/ncarg-4.4.1"
export NCARG_ROOT="${NCARG}"
export NCARG_BIN="${NCARG}/bin"
export NCARG_LIB="${NCARG}/lib"
export NCARG_INCLUDE="${NCARG}/include"
export NCARG_RANGS="${NCARG}/rangs"
export PATH="${NCARG}/bin:${PATH}"
export LD_LIBRARY_PATH="${NCARG}/lib:${LD_LIBRARY_PATH}"
export MANPATH="${NCARG}/man:${MANPATH}"
```

設定後、設定を有効にする。

```
# . ~/.bashrc
```

NCL

NCL は NCAR Command Language の略。EarthSystem Grid にアクセスし、「Scientific Data Processing and Visualization Software」→「NCAR Command Language (NCL)」→「NCL 4.2.0.a033 Release (latest)」の順にアクセスする。ログイン画面が現れるので、ID とパスワードを入力する。

新規ユーザーの場合：

1. 「request」をクリックして新アカウントを作成する。
2. アカウント申請ページでは「ESG Contact Person:」以外の全項目に記入。「Request」をクリック。
3. website から申請確認メールが届く。メールの指示に従い、メール内の URL にアクセス。
4. 登録完了までしばらく待つ（一晩ほどかかる）。登録完了メールが来たらメールの指示に従う。
5. ログインページにて ID とパスワードを入力。ログインページ下の「I Accept」をクリック。

「NCL 4.2.0.a033 binaries (not OPeNDAP-enabled)」→「NCL 4.2.0.a033 32-bit binary for i686 chips for LINUX (using GNU)」の順にクリックし、ファイルをダウンロードする。ダウンロードしたファイルは/tmp に移動して、解凍する。

```
# cd /tmp
# tar zxvf ncl-4.2.0.a033.Linux_i686.tar.gz -C /usr/local/ncarg-4.4.1
```

以上でインストールが完了。環境設定を記述する。

```
### ~/.bashrc
### NCL 4.2.0
export NCLCOMMAND="${NCARG}/bin/idt"
export NCL_COMMAND="${NCARG}/bin/ncl"
```

設定後、設定を有効にする。

```
# . ~/.bashrc
```

以下、テスト項目。これが正常に動けば、インストールは問題ない。

```
--- NCARG Test ---
# ncargex cpex08
# idt cpex08.ncgm
# ctrans -d X11 cpex08.ncgm

--- NCL Test ---
# ng4ex gsun01n
# ncl gsun01n.ncl
```

NCL/NCARG 5.0.0

NCARG は NCAR Graphics、NCL は NCAR Command Language の略。ダウンロード、インストールともに NCL のページに統合されたため、NCARG をダウンロード、インストールだけでも以下のユーザー登録が必要となる。EarthSystem Grid にアクセスし、「Scientific Data Processing and Visualization Software」→「NCAR Command Language (NCL)」→「NCL 5.0.0 Release」→「NCL 5.0.0 source code」の順にアクセスする。ログイン画面が現れるので、ID とパスワードを入力する。

新規ユーザーの場合：

1. 「request」をクリックして新アカウントを作成する。
2. アカウント申請ページでは「ESG Contact Person:」以外の全項目に記入。「Request」をクリック。
3. website から申請確認メールが届く。メールの指示に従い、メール内の URL にアクセス。
4. 登録完了までしばらく待つ（一晩ほどかかる）。登録完了メールが来たらメールの指示に従う。
5. ログインページにて ID とパスワードを入力。ログインページ下の「I Accept」をクリック。

「NCL/NCAR Graphics 5.0.0 source code」をクリックし、ファイルをダウンロードする。ダウンロードしたファイルは/tmp に移動して、解凍する。

まず、HDF インクルードファイルにあった ncurses がインストールされている必要がある。

```
# apt-get install ncurses ncurses-devel
```

解凍後、ソースディレクトリに移る。

```
# cd /tmp
# tar zxvf ncl_ncarg-5.0.0.src.tar.gz
# cd ncl_ncarg-5.0.0
```

インストールファイルを修正する必要があるので、適宜修正。

```
### ./ncarview/src/lib/libncarg_ras/hdf.c line:67-69
#include <hdf.h>
#include <df.h>
#include <dfgr.h>

### ./ni/src/ncl/NclHDF.c line:26-27
#include <netcdf.h> (順番を入れ替える)
#include <hdf.h>

### HDF-EOS 使用時
### ./ni/src/ncl/NclHDFEOS.c line:26-27
#include <netcdf.h> (順番を入れ替える)
#include <hdf.h>

### ./ni/src/lib/hlu/CnTriMeshRenderer.c line:33-34
#ifdef BuildTRIANGLE
#define REAL double
#include <ncarg/hlu/triangle.h>
#endif
```

公式サイトのトラブルシューティングから、修正された wrapit.c をダウンロードして新しいファイルに入れ替える。

```
# cp /tmp/wrapit.c ./ni/src/mkwrap
# touch ./ni/src/mkwrap/wrapit.c
```

./config/LINUX.INTEL の Fortran コマンド ifc は古いコマンド名。現在の ifort に修正しておく。ライブラリのパスに各種の lib、include を追加。修正後、LINUX ファイルにコピー。

```
### ./config/LINUX.INTEL line:29-34
#define CppCommand 'ifort -E -free'
#define CCompiler  icc -mp
#define FCompiler  ifort -mp -Vaxlib
#define CtoFLibraries "-L/opt/intel/fc/10.1.015/lib" -ICEPCF90 -IF90 -lcxa -lifcore -lcprts -lm
#define CcOptions   -ansi -w -lpng -lsz
#define FcOptions   -cm -w

### ./config/LINUX.INTEL line:38-39
#define ArchRecLibSearch -L/usr/X11R6/lib -L/usr/local/netcdf-3.6.2/lib -L/usr/local/hdf-4.2r3/lib -L/usr/local/udunits-1.12.4/lib -L/usr/local/g2clib-1.0.5
#define ArchRecIncSearch -I/usr/X11R6/include -I/usr/local/netcdf-3.6.2/include -I/usr/local/hdf-4.2r3/include -I/usr/local/udunits-1.12.4/include -I/usr/local/g2clib-1.0.5
```

```
# cp ./config/LINUX.INTEL ./config/LINUX
```

インストール先ディレクトリを作成後、設定用スクリプトを実行する。各種オプションは無論インストール済みのプログラム（HDF/Udunits/G2clib）のみを選択する。インストール先ディレクトリ指定は/usr/local/ncl_ncarg-5.0.0とする。設定後、コンパイル実行。

```
# mkdir -p /usr/local/ncl_ncarg-5.0.0
# ./Configure
# make Everything >& make-output & tail -f make-output
```

次に高解像度海岸線データをインストールする。Rainer Feistel にアクセスし、ページ中程にある rangs*.zip、gshhs*.zip を全てダウンロードする。インストール先ディレクトリを作成し、インストール。

```
# mkdir /usr/local/ncl_ncarg-5.0.0/rangs
# cd /tmp
# for i in 0 1 2 3 4;
> do unzip rangs($i).zip;
> unzip gshhs($i).zip;
> done;
> mv *.rim *.cat *.cel /usr/local/ncl_ncarg-5.0.0/rangs
```

インストール後、環境設定を行う。

```
### ~/.bashrc

### NCL/NCARG
export NCARG="/usr/local/ncl_ncarg-5.0.0"
export NCARG_ROOT="${NCARG}"
export NCARG_BIN="${NCARG}/bin"
export NCARG_LIB="${NCARG}/lib"
export NCARG_INCLUDE="${NCARG}/include"
export NCARG_RANGS="${NCARG}/rangs"
export NCLCOMMAND="${NCARG}/bin/idt"
export NCL_COMMAND="${NCARG}/bin/ncl"
export PATH="${NCARG}/bin:${PATH}"
export LD_LIBRARY_PATH="${NCARG}/lib:${LD_LIBRARY_PATH}"
export MANPATH="${NCARG}/man:${MANPATH}"
```

設定後、設定を有効にする。

```
# ~/.bashrc
```

GrADS 1.9b4

正式名称は Grid Analysis and Display System。解析データの可視化ソフトである。

Downloading GrADS Software からソースファイルをダウンロード。解凍し、ディレクトリへ移動。

```
# cd /tmp
# tar zxvf grads-src-1.9b4.tar.gz
# cd grads-1.9b4
```

ソースのコンパイルに必要なプログラムのインストール、その確認をしておく。

```
# apt-get install readline readline-devel
# apt-get install gd gd-devel
```

適宜、ソースファイルを修正する。

```
### ./src/gxhpng.c line:380

int gdCompareInt(const void *a, const void *b)
{
    return (*(const int *)a) - (*(const int *)b);
}

### ./src/gxhpng.c line:461
    qsort(im->polyInts, ints, sizeof(int), (void *) gdCompareInt);
```

各種リンカファイルに対し、/usr/lib にシンボリックリンクを貼る。

```
# cd /usr/lib
# ln -s /usr/local/*/lib/*.a .
# ln -s /usr/local/*/*.a .
# cd /tmp/grads-1.9b4
```

インストール用スクリプトを記述する。

```
#!/bin/bash

export CC=icc
export CXX=icpc
export FC=ifort
export F90=ifort
export CFLAGS="-O2"
export FFLAGS="-O -mp"
export CPPFLAGS="-I/usr/local/netcdf-3.6.2/include -I/usr/local/szip-2.1/include -I/usr/local/hdf-4.2r3/include -I/
usr/local/udunits-1.12.4/include -I/usr/local/g2clib-1.0.5"
export LIBS="-lnetcdf -lsz"
export SUPPLIBS="/usr"

{
./configure --prefix=/usr/local/grads-1.9b4
make
make install
} >& make-output & tail -f make-output
```

インストール先ディレクトリを作成後、インストール用スクリプトを実行。ドキュメント等をソースディレクトリからコピーする。

```
# mkdir -p /usr/local/grads-1.9b4
# sh install
# cp -a data doc etc lib /usr/local/grads-1.9b4
```

次にスクリプトファイルをダウンロード、インストールする。wget を使用し、FTP サイトからデータを取得する。

```
# cd /usr/local/grads-1.9b4
# mkdir scripts
# cd scripts
# wget ftp://grads.iges.org/grads/scripts/*
```

インストールが完了したら、環境設定を記述しておく。

```
### ~/.bashrc

### GrADS
export GRADS="/usr/local/grads-1.9b4"
export PATH="${GRADS}/bin:${PATH}"
export GASCRP="${GRADS}/scripts"
export GADDIR="${GRADS}/data"
```

環境設定を有効にしたら終了。

```
# . ~/.bashrc
```

WRFSI

※ WRF Version2.2 から WRFSI の代わりに WPS が前処理プログラムとして使われるようになっている。

WRFSI は次にインストールする WRF の初期化処理プログラム (Standard Initialization : SI) である。WRFSI からダウンロードする。研究開発用の ARW 型と、現業用の NMM 型があるが、ここでは ARW を選択。また、ダウンロードページから地理データ (Geog Data) にアクセスするスクリプトファイルもダウンロードしておく。

WRFSI は Perl-Tk を使って GUI を提供している。2006 年 10 月現在の Vine Linux 3.2 の場合、この Perl-Tk のバージョンは 804.026 であるが、このバージョンでは GUI がうまく動かない。バージョン 800.025 が必要になる。このバージョンの Perl-Tk があればアンインストールしておき、独自に Perl-Tk をインストールする。SourceForge.net から該当バージョンのソースファイルをダウンロード。解凍し、ディレクトリへ移動する。

```
# apt-get remove perl-Tk
# cd /tmp
# tar jxvf perl-Tk-800.025-2-src.tar.bz2
# tar jxvf Tk-800.025.tar.bz2
# cd Tk-800.025
```

Makefile を Perl スクリプトで作成し、インストールする。インストール先は /usr/local/perl-Tk-800.025 とする。作成した Makefile を一部修正してインストール。このときはインストールに gcc を使うので、初期設定のままでもよい。

```
# perl Makefile.PL
```

```
### Makefile
PREFIX = /usr/local/perl-Tk-800.025
PERLPREFIX = /usr/local/perl-Tk-800.025
SITEPREFIX = /usr/local/perl-Tk-800.025
VENDORPREFIX = /usr/local/perl-Tk-800.025
```

```
# mkdir /usr/local/perl-Tk-800.025
# make
# make test
# make install
```

これらが完了したら、パスを通すためのシンボリックリンクを作成する。その後、テストがうまくいったら完了。

```
# cd /usr/local/lib
# rm -rf site_perl
# ln -s /usr/local/perl-Tk-800.025/lib/perl5/site_perl
# perl -e 'use Tk; print " o This is Perl/Tk, version $Tk::VERSION installed in library $Tk::library
```

WRFSI はユーザ権限でインストールする。インストール先はホームディレクトリ直下に WRF ディレクトリを作成し、その中とする。解凍し、ディレクトリへ移動。

```
$ cd ~
$ mkdir WRF
$ cd WRF
$ tar zxvf /tmp/wrfsi_v2.1.2.tar.gz
$ cd wrfsi
```

WRFSI のコンパイルには NetCDF のインストールが必須であり、同じコンパイラでコンパイルする必要がある。もしコンパイルが上手く通らない場合は、ちゃんと NetCDF のインストールが同じコンパイラで成功しているかどうか確認すること。ここでは icc/fort を一貫して使っているので問題はないと思うが。

ソースファイルの記述を修正する。./src/include/makefile_pcintel.inc.in の該当箇所を確認し、修正。

```
### ./src/include/makefile_pcintel.inc.in
CC = icc
NCARGFC = /usr/local/ncarg-4.4.1/bin/ncargf77 # PATH は適宜変更
CPP = ifort -E -free
CPPFLAGS = $(INC) $(DEFS)
```

また、このバージョンでは hinterp.exe のコンパイルエラー問題がある。hinterp.exe をコンパイルする際に、「undefined reference to nc*... (nc*関数が定義 されていない)」という一見して NetCDF のインクルードがうまくいっていないように思えるエラーが発生する。これは NetCDF の問題ではなく、hinterp の Makefile にバグが存在するためである。以下に修正用の Makefile を記述しているので、これを src/hinterp/Makefile と差し替えること。

参考：

- * http://tornado.meso.com/wrf_forum/index.php?showtopic=694
- * <http://cfa-www.harvard.edu/~tkurosu/OnE/WRF/index.html>

```
#dis
#dis  Open Source License/Disclaimer, Forecast Systems Laboratory
#dis  NOAA/OAR/FSL, 325 Broadway Boulder, CO 80305
#dis
#dis  This software is distributed under the Open Source Definition,
#dis  which may be found at http://www.opensource.org/osd.html.
#dis
#dis  In particular, redistribution and use in source and binary forms,
#dis  with or without modification, are permitted provided that the
#dis  following conditions are met:
#dis
#dis  - Redistributions of source code must retain this notice, this
#dis  list of conditions and the following disclaimer.
#dis
#dis  - Redistributions in binary form must provide access to this
#dis  notice, this list of conditions and the following disclaimer, and
#dis  the underlying source code.
#dis
#dis  - All modifications to this software must be clearly documented,
#dis  and are solely the responsibility of the agent making the
#dis  modifications.
#dis
#dis  - If significant modifications or enhancements are made to this
#dis  software, the FSL Software Policy Manager
#dis  (softwaremgr@fsl.noaa.gov) should be notified.
#dis
#dis  THIS SOFTWARE AND ITS DOCUMENTATION ARE IN THE PUBLIC DOMAIN
#dis  AND ARE FURNISHED "AS IS." THE AUTHORS, THE UNITED STATES
#dis  GOVERNMENT, ITS INSTRUMENTALITIES, OFFICERS, EMPLOYEES, AND
#dis  AGENTS MAKE NO WARRANTY, EXPRESS OR IMPLIED, AS TO THE USEFULNESS
#dis  OF THE SOFTWARE AND DOCUMENTATION FOR ANY PURPOSE. THEY ASSUME
#dis  NO RESPONSIBILITY (1) FOR THE USE OF THE SOFTWARE AND
```

```

#dis DOCUMENTATION; OR (2) TO PROVIDE TECHNICAL SUPPORT TO USERS.
#dis

SRCROOT=../..

include $(SRCROOT)/src/include/makefile.inc

RM=rm -f

.SUFFIXES:      .F .o

.F.o: $(FOBJS)
    $(RM) $@
    $(FC) -c $(FFLAGS) $(FREE) $(INC) $(MODULELIB) $(LAPSLIB) -c $< -o $@

EXE= hinterp.exe

FSRC= hinterp.F
    module_domain_info.F
    module_gridded_data.F
    module_hinterp_setup.F
    module_hinterp_gribprep.F
    proc_make_variable_metadata.F
    proc_output_variable.F
    proc_store_global_metadata.F
    wrf_debug.F

FOBJS=$(FSRC:.F=.o)

all:      $(EXE)

$(EXE):      $(FOBJS)
    $(FC) -o $(EXE) $(LDFLAGS) $(FOBJS) $(LAPSLIB) $(MODULELIB) $(IOAPI)
    $(OTHERLIBS)

hinterp.o:      module_hinterp_setup.o module_domain_info.o module_gridded_data.o
    module_hinterp_gribprep.o wrf_debug.o

module_domain_info.o: module_hinterp_setup.o

module_hinterp_gribprep.o:      module_hinterp_setup.o module_domain_info.o
    module_gridded_data.o

proc_output_variable.o: module_hinterp_setup.o module_gridded_data.o

proc_store_global_metadata.o:      module_hinterp_setup.o

debug:
    $(RM) *.o *.exe *.mod *.M ; $(MAKE) $(EXE)
    "MODULELIB      =      $(MODULELIBDEBUG)"
    "FFLAGS      =      $(DBFLAGS)" )

clean:
    $(RM) $(FOBJS) $(EXE) *.o *.mod *.M core

```

```
install: $(EXE)
          $(INSTALL) $(EXE) $(INSTALLROOT)/bin/$(EXE)
```

修正が完了したら、コンパイルを実行する。

```
$ ./install_wrfpsi.pl --machine=pcintel --install_ui=y
```

次に地理データをインストールする。「SI Geographical Data」を取得する。wgetを用いたダウンロードスクリプトがあるのでこれを使用する。ファイアーウォールでwgetが成功しない場合は、--passive-ftpオプションをスクリプト中のwgetに追加する。

```
$ cd ~/WRF
$ sh /tmp/wget_geog.shx
$ mv geog GEOG
$ cd wrfsi/extdata
$ rm -rf GEOG
$ ln -s .././GEOG
```

これで./wrf_toolsでWRFSIを実行できるが、Vine Linux 3.2ではスタックサイズが8192KBに制限されており、この制限にひっかかってLocalizationに失敗してしまう。ログを見ると、Signal 11を吐いて強制終了していることが分かる。

```
### wrf_tools
ERROR: 65280 Command /usr/bin/perl (...)/localize_domain.pl

### localize_domain.log
ERROR: (...)/gridgen_model.exe ran with signal 11
```

このエラーを回避するためには、スタックサイズを無制限にしてツールを実行する必要がある。次のコマンドを用いる。

```
$ ulimit -s unlimited
$ ./wrf_tools
```

WRF Version2.1.2

```
$ cd ~/WRF  
$ tar zxvf /tmp/WRFV2.1.2.TAR.gz  
$ cd WRFV2
```

arch/configure.defaults で、5566 行から始まる「Intel xeon i686 ia32 Xeon Linux, ifort compiler (single-threaded, no nesting)」の設定を修正。その後、./configure スクリプトで環境設定を実行し、上記で修正した番号 (7) を選択。コンパイルを実行する。

```
### arch/configure.defaults line:5566  
CC = icc  
CPP = ifort -E -free
```

```
$ ./configure  
$ ./compile em_real
```

run ディレクトリに ndown.exe、real.exe、wrf.exe が作成されればコンパイル成功。