READMEnet.txt last update 3/15/01 (Dierk Polzin)

```
                  ###   ######    ###    #####
                   #    #     #    #    #     #
                   #    #     #    #    #
                   #    ######     #     #####
                   #    #     #    #          #
                   #    #     #    #    #     #
                  ###   ######    ###    #####

                  Integrated BIosphere Simulator
```

======================Unix/Linux NetCDF Mode======================

GENERAL INSTRUCTIONS

Compiling and running IBIS requires a FORTRAN compiler, a 'make' utility, and the netcdf library (V3.3 or later, obtained from www.unitadata.ucar.edu) compiled with the FORTRAN option turned on.

1) Set up proper input/output directories in the location from which you will run IBIS:

```
     mkdir output
     mkdir output/yearly output/monthly output/daily
     mkdir input
     mkdir input/anom
     mkdir restart
```

2) Place your input files in input (and input/anom if you have monthly anomalies). Input files are expected to be in netcdf format and be readable by subroutine readvar (ies-io.f). Files should have the dimensions of longitude, latitude, an optional 3rd dimension ('level', usually), and time. See subroutines readit, rdanom, rdday, and inird for details. Subroutine inird is special in that it reads an integer year from the units attribute of the time dimension. Time units for input, therefore are required to be in the form 'days since yyyy-12-31'. Optionally, you may have a clock time after the yyyy-mm-dd portion. Since the yyyy portion of the string is converted to integer with a formatted read statement, it is essential that yyyy sits in the 12th through 15th place in the units string. Also, it is expected that the monthly anomalies begin in January of the year after yyyy (one day since yyyy-12-31) as stored in time's units attribute in all monthly anomaly files, and daily fields begin on January 1st of the year after yyyy as stored in time's units attribute in all daily anomaly files.

subroutine readit:

input file name	when used	variable name	3rd dimension
input/surta.sand.nc	always	surta	level
input/surta.clay.nc	always	surta	level
input/topo.nc	always	topo	level
input/vegtype.nc	isimveg <= 1	vegtype	level
input/deltat.nc	always	deltat	level
input/soita.nc	always	tex	layer
input/wetd.mon.nc	always	wetd	level

input/temp.mon.nc	always	temp	level
input/trange.mon.nc	always	trange	level
input/prec.mon.nc	always	prec	level
input/wspd.mon.nc	always	wspd	level
input/cld.mon.nc	always	cld	level
input/rhum.mon.nc	always	rhum	level

subroutine rdanom:

input file name	when used	variable name	3rd dimension
input/anom/temp.danom.nc	monthly anomalies	temp	level
input/anom/trange.danom.nc	monthly anomalies	trange	level
input/anom/prec.danom.nc	monthly anomalies	prec	level
input/anom/cld.danom.nc	monthly anomalies	cld	level
input/anom/rhum.danom.nc	monthly anomalies	rhum	level
input/anom/wspd.danom.nc	monthly anomalies	wspd	level
input/anom/wetd.danom.nc	monthly anomalies	wetd	level

subroutine rdday:

input file name	when used	variable name	3rd dimension
input/prec.daily.nc	daily means	prec	level
input/temp.daily.nc	daily means	temp	level
input/trange.daily.nc	daily means	trange	level
input/cld.daily.nc	daily means	cld	level
input/wspd.daily.nc	daily means	wspd	level
input/sphum.daily.nc	daily means	sphum	level

subroutine inird:

input file name	when used	variable name	3rd dimension
input/prec.daily.nc	daily means	prec	not used
input/anom/temp.mon.nc	monthly anomalies	temp	not used

3) Edit compar.h so that the following parameters are valid for your particular input data:

nlon - integer number of grid cells in the east-west direction.
nlat - integer number of grid cells in the north-south direction.
npoi - integer number of land grid cells in the lat/lon box over which you will be running IBIS (see ibis.infile explanation of snorth, etc.).
xres - real number of degrees longitude separating the first and second grid points in the east-west direction.
yres - real number of degrees latitude separating the first and second grid points in the north-south direction.

4) Edit ibis.infile so that the following input values are to your liking (note: input values must be in the exact order mentioned below). Unless specifically mentioned, do not change any of the values if you restart the model.

irestart - enter 0 if this is a new run, 1 if it is a continuation of a previous run.
iyear0 - enter the calendar year for the very first run of the simulation. Do not change the values of this year if you are restarting the run (irestart = 1).
nrun - the number of years you wish to run the model. You should

change this value if you are restarting the model. For example, if you initially wanted to run the model for 100 years, but had to restart during year 80, change nrun form 100 (initial value) to 21 (number of years left to complete).

nanom - number of the year for which to start reading monthly anomalies. Set this value to something very large if you don't want to read in anomalies during the simulation (greater than iyear0+nrun-1). If you will be reading in monthly anomalies, set this to the first anomaly year. For example, if iyear0 = 1950 and you wish to start reading monthly anomalies in 1965, then set nanom equal to 1965.

ndprecy - number of the year for which to start reading daily means. Set this value to something very large if you don't want to read in daily means during the simulation (greater than iyear0+nrun-1). If you will be reading in daily means, set this to the first year in which you want to use daily means. For example, if iyear0 = 1950 and you wish to start reading daily fields in 1965, then set ndprecy equal to 1965.

soilcspin - enter 1 if you want to use the accelerated soil spinup, 0 if you don't.

iyearout - enter 1 if you want yearly output, 0 if not.

imonthout - enter 1 if you want monthly output, 0 if not.

idailyout - enter 1 if you want daily output, 0 if not.

isimveg - enter 0 for static vegetation (vegetation will not grow or die), 1 for dynamic vegetation, or 2 for dynamic vegetation starting from a cold start (no vegetation to start out with, just 'seeds'). Note that for isimveg equal to 0 or 1, you must have the file input/vegtype.nc to provide initial conditions for vegetation distribution.

isimfire - enter 0 for fixed fire conditions, 1 for dynamic fire. We are not satisfied with the fire scheme, so we recommend that you don't use dynamic fire.

isimco2 - enter 0 for fixed co2 concentrations (value in co2init), or 1 for ramped co2 concentrations. If you choose 1, the co2 concentration will change each year according to the polynomial equation found in subroutine co2 (physiology.f), which approximates the change in co2 from pre-industrial times to present (this currently commented out, so you'll need to remove the c's).

co2init - initial value for co2 concentration (mol/mol) to be used in the simulation.

o2init - initial value for o2 concentration (mol/mol) to be used in the simulation.

dtime - time step (seconds) to use for the simulation, must be an even divisor of 86400 (24 hours).

idiag - enter 0 for no diagnostic output, 1 through 10 for number of diagnostic files (see diag.infile explanation for details).

snorth - latitude of northern edge of a subset box over which you wish to run IBIS. A subset is considered to be any rectangular (in lat/lon) region that is less than the full lat/lon grid contained in your input files. The value of snorth will be used to calculate the index of the gridcell which contains this value. If the lat/lon boundaries of the subset box that you indicate contains fractions of gridboxes, the entire gridboxes are used. Subsetting ONLY works for rectangular grids - grids in which latitude is constant for all grid cells in each row and longitude is constant for all grid cells in each column.

ssouth - latitude of southern edge of a subset box over which you wish to run IBIS (see snorth).

swest - longitude of western edge of a subset box over which you wish to run IBIS (see snorth).

seast - longitude of eastern edge of a subset box over which you wish to run IBIS (see snorth).

5) If you wish, edit diag.infile. Diag.infile is used to print out selected variables for every nth time step. For example, if you wish to know how the temperature of the top soil layer changes for every other time step in 5 separate grid cells, you would first of all enter 5 for idiag in ibis.infile (each file represents a single grid cell).

Then select five locations and enter the latitude and longitude for each grid cell center in diag.infile's spaces for diaglat0 and diaglon0, diaglat1 and diaglon1, diaglat2 and diaglon2, diaglat3 and diaglon3, and diaglat4 and diaglon4.

Then enter the beginning and end years for the desired time span in diag.infile's spaces for diagstart# and diagend# (where # means 0 through 4 in this example).

Also enter the frequency for writing the diagnostic output in the space for nfreq# in ibis.infile. For every time step, nfreq# = 1, every other means nfreq# = 2, every third means nfreq# = 3, etc. Note that if you want to print out every nfreq steps, and that is not an even factor of the number of time steps per day, the diagnostics won't be evenly spaced between the last diagnostic write of the day and the first diagnostic write of the next day.

Lastly, in diag.infile, edit the array at the bottom so that for each diagnostic file (column), there is a 1 in the row for the diagnostic variable(s) that you want. In our example, you would change the 0's to 1's in columns 1 through 5 in the row for tsoi, nsoilay=1. All other places in the array would contain zeroes.

10 is the maximum number of diagnostic files (10 individual grid cells), and a maximum of 12 variables may be chosen for each grid cell. If you want more than 12 variables for a single grid cell, you can request two diagnostic files which merely have the same coordinates, but different variables. Diagnostic files may be written out over the same or different time periods, and the same or different frequency.

Read the comments at the beginning of diag.infile for another explanation of how to use diag.infile. Do not, however, remove thosecomments. The values entered in diag.infile must be in exactly the right rows.

Each time you run ibis with diagnostics output turned on, you may wish to remove any old diagnostic files. Any existing diagnostic files will have output appended to them.

6) Compile IBIS by typing 'make ibisnetcdf' in the directory which contains the ibis FORTRAN code. However, before doing so, you may need to edit makefile so that the F77_OPTIONS, INCLUDE_DIRS, and LD_OPTIONS_NETCDF match those on your computer. The executable will be named 'ibisnetcdf'. You may move the executable to the directory in which your input/output directories exist. If so, be sure to move ibis.infile and diag.infile there too.

7) To run IBIS, type 'ibisnetcdf' while in the directory which contains the ibis executable and the input/output directories. To run ibis in the background and capture all screen output to a file, type 'ibisnetcdf >& ibisnetcdf.out &' instead (without the quotes). All screen output, including system error messages, will be written to ibisnetcdf.out.

8) You do not need to recompile if you do a restart. Just edit ibis.infile. You must recompile if you want to run over a different number of land points (subsetting) or for a different resolution. In either case, you must change compar.h and therefore recompile. You may not restart if you change the number of points, location of points, or resolution. If you run ibis a second time in the same working directory, but do not do a restart, all output files (except diagnostic files) will be overwritten.

9) If you wish to change input or output files, read the comments at the end of io.f before making any changes. Those comments explain the syntax of read/write calls and give examples.

PLATFORM ISSUES

IBIS was written and tested on SGI workstations. We have also successfully run IBIS on a Sun, and on a PC running RedHat Linux. To our knowledge, only two portions of the code may not be consistent with your platform. The first is the function 'date'. The MipsPro compiler on our SGI's returns an 8-character string containing the date. Some Sun compilers return a 9-character string. Space is reserved for up to 10 characters (look for the variable cdate in io.f). If your computer's FORTRAN compiler does not have an intrinsic function for date, has a function of a different name, or returns something other than 10 or fewer characters, you'll have to edit io.f accordingly. Look in wrestart, wdaily, wmonthly, and wyearly for places where you may need to change the code. If nothing is available for you to use, you may eliminate the reference to the function 'date' and the variable cdate. It is only used to create a history attribute in the output files.

The second portion of code which may not consistent with your platform is the command 'flush' in stats.f. This command is used to update the output in the files 'ibis.out.global' and 'ibis.out.vegtype'. This prevents your computer from buffering output for these files. If you do not have a version of flush in your compiler, you may comment out those lines of code. You may not be able to read output for a particular year in ibis.out.global or ibis.out.vegtype until the program has completed a few extra years.