Bern Simple Climate Model (BernSCM) manual

Kuno Strassmann

September 22, 2017

The model as described in this manual is designed for a Unix/Linux shell environment. Model code is written in Fortran90 and set up for compiling with gfortran or pgf90 (tested with gfortran 5.4.1 and pgf90 7.2-3). Required additional tools are GNU make and sed for compiling, and (optionally) xmgrace for checking results.

Getting started

To generate executables and run example simulations, execute the script

```
run_examples.sh
```

To plot the example simulations, execute the scripts

```
xmgrace_pulse.sh
xmgrace_c4mip.sh
```

The usage is explained in detail below.

1. Compiling

Within the directory where this readme file resides, do

```
make
```

to compile with the default compiler gfortran, or alternatively

```
make profile=pgf90
```

to compile with the pgf90 compiler (requires gnu make, and gfortran or pgf90). This will produce the executable bernSCM.

2. Running a simulation

When bernSCM is executed, it will interactively query some parameters for the simulation (climate sensitivity, C cycle sensitivities to temperature/CO2, forcing file identifier (scenario), and an additional identifier for the simulation). This information can be entered by hand, but will typically be fed to the model by a "runfile", a text file containing one input parameter per line.

```
bernSCM < runfiles/run_mysimulation
```

The contents of the runfile are, for example

```
2.5      #climate sensitivity
.true.   #temperature dependent
.true.   #CO2 dependent
c4mip_a2 #simulation ID
test     #Optional additional identifier
```

3. Input data: the forcing file

The forcing file contains the boundary conditions or forcing of the simulation.

Forcing files are located in the directory "forcing", and follow the naming convention

```
forcing_<scenario>.dat
```

where <scenario> is an identifier entered as interactive input (see 2.).

Forcing files are text files containing a table with the following data columns:

- simulation time (Time/yr)
- global mean surface temperature deviation from preindustrial (glob_temp_dev/K)
- non-CO2 radiative forcing (RF_nonC/Wm-2)
- radiative forcing for budget closure (budget_RF/Wm-2)
- atmospheric CO2 (co2_atm/ppm)
- fossil/anthropogenic CO2 emissions (fossil_em/GtC/yr)
- C sink flux for budget closure (budget_sink/GtC/yr).

Important specifications:

- All forcing variables must be present IN THE GIVEN ORDER (header is not read)
- The model is set up to run from preindustrial equilibrium, thus initial time must be preindustrial.
- Data may be arbitrary spaced in time and will be interpolated.
- Each record refers to a POINT in time.
- Some columns will contain the "missing value" flag -9999.9999 (NA). NAs indicate variables to be determined by budget closure.
- At the initial record, ALL variables must be defined (not NA).
- All physical quantities may be determined by budget closure, but for RF and C emissions/uptake, dedicated budget variables must be used (budget_RF, budget_sink). Budget variables must be set to zero when not used for closure.
- Variables used for budget closure may change along the simulation (e.g., different over historic and future period). When the budget variable changes, e.g. from to budget_sink to co2_atm, there must be a record with both these variables defined (not NA). This is because each record represents a point in time.
- The following budget cases as indicated by NAs are possible:

Budget	glob_temp_dev	budget_RF	co2_atm	budget_sink
CT	NA	0	NA	0
ET	NA	0	prescribed	NA
ER	prescribed	NA	prescribed	NA
CE	prescribed	prescribed	NA	NA

Legend:

CT given CO2 emissions and non-CO2 RF, solve for atmospheric CO2 and temperature

ET given CO2 concentrations, emissions and non-CO2 RF, solve for residual CO2 uptake and temperature

ER given CO2 concentrations, emissions, non-CO2 RF, and temperature, solve for residual CO2 uptake and RF

CE given CO2 emissions, non-CO2 RF, and temperature, solve for atmospheric CO2 and CO2 uptake

For determining non-CO2 RF and allowable emissions, the entries budget_RF and budget_sink can be used, while setting the columns RF_nonC and fossil_em to 0.

4. Numerical solution

The code supports timesteps ranging from subyear to decadal. The larger timesteps require more complicated numerical schemes for a stable solution. Therefore the timestep is set at the preprocessing stage, by setting the preprocessor flag deltat in the file src/control.inc, for example:

```
#define deltat 1d0
```

To set the appropriate numerical scheme, the file may contain the preprocessor flags shown in the following table:

numerical scheme	deltat (yr)	implicitO	implicitL	linear
Euler forward, simplest	~0.1	0	0	0
Implicit step(a)	~1	1	(1)	0
Implicit step(b)	~10	1	(1)	1

(a) For stability, implicit treatment is essential for ocean C, optional for land C. (b) Linear discretization is needed for accuracy with large timesteps.

5. Model versions

The land/ocean pulse-response functions are specified in src/parLand*.inc, src/parOcean*.inc. Similarly, NPP parametrizations are defined in src/parNPP*.inc, src/npp*.inc.

Different model components are available, including for the ocean HILDA (parOceanHILDA.inc), Bern2D (parOceanB2D.inc), and for the land the Bern 4box model (parNPP_4box.inc, npp_4box.finc), and the HRBM substitute model (parLandHRBM.inc, parNPP_HRBM.inc, npp_HRBM.finc).

When compiling, generic parameter files will be read in, thus to include any component, link or copy the corresponding file, e.g.:

```
parLandHRBM.inc -> parLand.inc
```

Additional model components may be added in the same way, using the existing files as a template.

Note on land models:

- For temperature-dependent land models (like HRBM), the preprocessor flag

```
#define LandTdep 1
```

in src/control.inc MUST be set to 1! Otherwise T-dependence will not be compiled in, leading to erroneus results without warning! For non-T-dependent models (4box), the flag must be removed or set to 0.

- NPP and pulse response from different parent models may be combined if so desired (e.g., 4box NPP with HRBM biosphere).

Note on preprocessed code:

- The source code (src/*.F files) is not very readable because it accomodates a range of numerical schemes for different purposes. Practical applications will often use a specific version exclusively (e.g. with a given timestep), which correspond to preprocessed code produced when compiling (src/*.for files), using flags set in the file src/control.inc. Thus it is recommended that the more readable preprocessed code be used for the implementation of a specific model version.

6. Output control

Internally, the model defines state variables (CO2, Temp, etc.) at time points, while C fluxes are defined in between at mid-timestep. Fluxes may be interpolated to time points with the preprocessor option dointerpol, which is set at the top of file src/bernSCM-output.F. Interpolation causes a negligible error in integrals. It is enabled per default (dointerpol=1), and can be switched off (dointerpol=0) if desired.

7. Varia

- The preprocessor flag #tequil (in src/control.inc) may be used to treat boxes with small turnover times as equilibrated. All timescales below tequil (yr) will be equilibrated, thus a value of 0, or missing flag, implies no equilibration. As equilibration does not strongly affect performance and may reduce accuracy, its use is not recommended.