Topographic Shading Subroutine for ARPS5.0.0Beta8

Augustin Colette – Robert L. Street

Environmental Fluid Mechanics Laboratory
Civil and Environmental Engineering
Stanford University
Stanford, CA 94305
augustin@stanford.edu , street@stanford.edu
June 2002

I - Introduction:

We decided to add a new subroutine to the Advanced Regional Prediction System. As of now ARPS takes into account the angle between the sunrays and the slope in the computation of the radiation balance. Consequently, a north-facing slope is not illuminated if the sun is south. However, over complex topographies, the sunrise can be delayed of several hours by the topographic shade. A new subroutine has been added to the code to account for that effect.

II - Method:

The topographic shade subroutine draws a line between each node of the horizontal grid and the sun and checks whether or not this line hits the topography.

1- Start at the point (i,j) of altitude hterain(i,j)
2- Follow a line going toward the sun (the sun position is given by its altitude and azimuth)
3- Each time that the projection of this line on the horizontal plane meets a grid cell boundary, a test is performed to compare the altitude along the line and the altitude of the terrain linearly interpolated between the two neighboring grid nodes.
4- A shade matrix is so computed: sh(i;j) = 0 if (i,j) is shaded, =1 otherwise.
5- The shade matrix is added in the radiation balance:

The incoming shortwave radiation (radsw) is multiplied by sh, so that a shaded point receives no sunlight

The direct components (as opposed to the diffusive terms) of the net radiation flux (rnflx) are also multiplied by sh.

III- Modified files of the code
(compared to arps5.0.0Beta8.tar.gz):

Each time a line has been modified, a comment is written in the code beginning with !augustin.

Initpara3d.f90:

A new variable is added in the list: radshade
The modification in initpara3d.f90 include:
Namelist, default values, input control parameters, write the log file.

Radshade	=0 :	no shade computed
	=1 :	topographic shade computed
	=2:	topographic shade computed on the whole topography but only the shade as computed at j=ny/2 is kept, this has been successfully used in the simulations of 2D idealized valleys, uniform in the north south direction. It was necessary to avoid any discrepancy in the radiation balance in the north-south direction due to the fact that the sun trajectory is not exactly East-West.

Radfrc3d.f90

Subroutine RADIATION:

- Add the variable sh(nx,ny)
- Add the variables sazimuth saltitude (solar azimuth and altitude), which will be outputs of ZENANGL and inputs of SHADE
- Change the CALL zenangl so that saltitude and sazimuth are outputted
- CALL subroutine SHADE
- Take the shade into account in the radiation balance if radopt = 1 (simplified radiation)
- Change the CALL radtrns so that the shade is an input of this subroutine

Subroutine ZENANGL:

-Add variables saltitude and sazimuth
- Compute these variables as functions of cosz, sdeclin, latscl, shranlg, sinz.
  - The solar altitude and azimuth are only computed at the center of the domain, considering that if the shade is taken into account, the domain is rather small (saltitude and sazimuth almost constant)
  - References:
  http://www.usc.edu/dept/architecture/mbs/tools/vrsolar/Help/solar_concepts.html
  http://www.uwinnipeg.ca/~blair/physclim/lab2.htm
  http://ra.stsci.edu/cgi-bin/gethelp.cgi?altaz.src

Subroutine SHADE:

Added, for more details, see the code, algorithm_sun_north.jpg and algorithm_sun_northwest.jpg .

Radtrns3d.f90

Subroutine RADTRNS

- Add sh in the inputs of this subroutine
- Add sh as a variable of this subourtine
- Take sh into account in the radiation balance (both radsw and rnflx, with and without staggering)

Globcst.inc

The variable radshade is added in the list of common variables

IV - Testing:

An Input file is provided to test the topographic shading subroutine, it comes with a topography consisting of two peaks located a latitude 45N and longitude 0E. This topography was generated with the matlab file testshade.m. If you would like to produce the same topography at a different location, you should change and run this script.

Shade.input
Testshade.trndata
Testshade.m

Running
../bin/arps <shade.input
gives sone .bin outputs

The modified input file for arpspltpost called shadeplt.input, when processed with the executable arpspltsh will print out a postscript file containing plots of the topography, radsw, rnflx.
This executable was obtained by doing some small changes of arpspltpost so that it would print out the incoming solar radiation and the net radiation flux.

V- Parallelization:

Method:

Some attempts were made to make this subroutine available in the mpi version of ARPS. *Unfortunately, the debugging was not fully successful*. We give here some hints on the issues to make subroutine shade available for arps_mpi.

The main problem here is that the shading subroutine needs to know the whole topography even when running in mpi (because a grid point can be shaded by the topography being stored on another processor).

Consequently, when the splitted topography is read at the beginning of the run on each processor, processor zero should read the whole topography and store it in an array of dimensions of the whole domain.

When calling the shading subroutine, all the processors should be put on hold exept processor zero. The mpi option mp_opt should be set to zero temporarily so that the shading will be computed as if the run was serial. Some changes must thus be made in the shading subroutine (nx,ny,x,y,terrain), so that it will have the correct grid parameters.

After computing the shade on the whole topography, processor zero should send it to the other processors, and a manual splitting must be done on each processor.
Afterwards, the runs can continue with no other modifications.

Issues:

These modifications were successfully made. The grid parameters must be very carefully changed and the splitting is also a big issue.

Unfortunately, I could not complete the parallelizing since I end up with an overflow problem. At some point of the shading subroutine, it seems that I was writing in a memory area reserved for variable f34. The run stops when computing the evaporation, variable f34 having some random values as 3.213421241E22 instead of 0.0000000E00.

List of the subroutine modified in the mpi version of shade:

1/ list of rhe subroutine to modify to read the whole terrain (e.g. hterglob) during the initalization
arps.f90 :
add hterglob in the CALL of subroutine INITIAL
init3d.f90:
subroutine INITIAL
add hterglob in the attributes of subroutine INITIAL
add hterglob in the CALL of subroutine INIGRDVAR
subroutine INIGRDVAR
add hterglob in the attributes of subroutine INIGRDVAR
add hterglob in the CALL of subroutine INIGRD
subroutine INIGRD
add hterglob in the attributes of subroutine INIGRD
read the terrain once for the whole topography on processor zero.

2/ when running the code, the array hterglob (the whole topography) must be an attribute of SHADE, to achieve that a couple of subroutines must be modified:

arps.f90
add hterglob in the attributes of CORDINTG
tinteg3d.f90
subroutine CORDINTG
add hterglob in the attributes of subroutine CORDINTG
add hterglob in the CALL of subroutine RADIATION
radfrc3d.f90
add hterglob in the attributes of subroutine RADIATION
+ A whole lot of changes in subroutine SHADE !!!

GOOD LUCK !