SMARTS code, version 2.9.5

For Linux

Quick Start Guide

SMARTS

Christian A. Gueymard
Solar Consulting Services, USA
December 2005

Installation

The distribution package is compressed. Decompression will create the SMARTS_295_Linux folder.

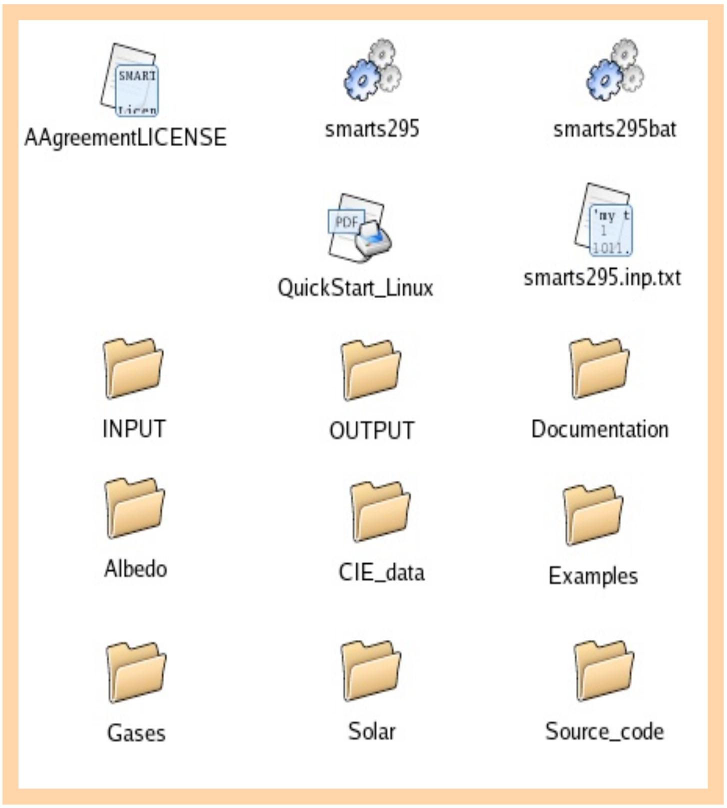
The source code has been successfully compiled and run under Linux RedHat 7, Fedora Core 2, Fedora Core 3, Debian 3.1, SunOS Core 5.9 and Cygwin, using the supplied g77 compiler. Binary executables for either standard execution or batch mode are provided at the root of the main directory. These binaries have been produced under Fedora Core 3.

No special hardware or software is required apart from a text editor. The standard g77 compiler is only required if it is necessary to recompile the code.

Important
Read the **License Agreement** in file **'AAgreement_LICENSE.txt'** and, *if and only if you agree with its terms*, proceed further as described below.

Folder content

A typical screenshot of the SMARTS_295_Linux folder is shown here. The exact appearance depends on the window manager you use and its settings.
The 'smarts295.inp.txt' file is a sample input file that you can use to familiarize yourself.
The User's Manual can be found in the Documentation folder.



Command line execution

To run SMARTS, open a Terminal window or an xterm and move to the directory where you have extracted the package:

```
cd /install_path/SMARTS_295_Linux
./smarts295.exe
```

and hit Enter after each line. This first message appears:

```
***********************************
 Welcome to SMARTS, version 2.9.5!
***********************************
SMARTS_295> Use standard mode with default input file?
 [If YES (or Y), execution will start immediately
 using the default input file smarts295.inp.txt]
 (Y/N) ==>
```

If you type "`y`" or "`yes`" and hit Enter, execution begins immediately. The following message appears when execution ends:

```
Total CPU time:   0.142 sec
```

It is clear that the actual execution time will vary according to your computational power. Execution of the sample input file must have created two separate output files: 'smarts295.out.txt' and 'smarts295.ext.txt', both located at the root of the SMARTS_295_Linux folder.

If you rather type "`n`" or "`no`" and hit Enter, the following message appears:

```
$$$ SMARTS_295> What is the path to the input file?
 * Type only "." if in the same folder
 * Do NOT type the last "/" of the chain
 * 2000 characters max. ==>
```

Type the path pointing to your input file or type "." if the file resides in the current folder. For instance, type "`INPUT`" if it is located in the INPUT folder of SMARTS_295_Linux. Hit Enter and the program will then ask for the proper filename:

```
$$$ SMARTS_295> Generic name for all input/output files
 * without any extension
 * 100 characters max.)? ==>
```

For instance, type "`Test_for_July`" if the input file is 'Test_for_July.inp.txt'. Hitting Enter results in:

```
$$$ SMARTS_295> You chose the following filenames:
 Input: Test_for_July.inp.txt
 Output: Test_for_July.out.txt
 Spreadsheet-ready: Test_for_July.ext.txt
 Smoothed results: Test_for_July.scn.txt
$$$ SMARTS_295> Is this OK? (Y/N) ==>
```

Type "`y`" if it is OK, or "`n`" if not, and hit Enter. If you choose "yes", execution starts normally. If you choose "no", the previous message reappears and the cycle continues until you reply "yes". The output files will be located in the same folder as the input file, e.g., in the INPUT folder for this example.

To use any sample input file contained in one of the example subfolders within the Examples folder, move it to the root of the SMARTS_295_Linux folder, thus replacing any pre-existing file of the same name. However, before undertaking the execution of any new run using an input file with the same name, it is imperative to either rename the existing output files, or to move them to the OUTPUT folder or any other folder. *Failing to do so would prevent proper execution.*

If this is a hindrance, you can change the appropriate options in the "OPEN" statement of the output files, so that at anytime they are overwritten. Recompile the Fortran code and run again. The source code resides in the Source_Code folder and can be compiled with any Fortran77 compiler. For example, the command

```
g77 –O –o smarts295.exe smarts295.f
```

should complete without any errors and create an optimized executable called 'smarts295.exe'. Finally, move or copy this new executable to the SMARTS_295_Linux folder.

Compiler issues
The standard Fortran compiler "g77" provided with the Linux distributions can be used as discussed above. However, users of Pentium processors can download the Intel Fortran compiler. The specific compiler is free for the Linux OS, while it is a commercial product for the Windows. The installation of the compiler is easily performed. The compilation time is increased but the execution time is substantially reduced.
The code would be compiled with the command "`ifort –o smarts295.exe smarts295.f`".

Batch mode
In batch mode the program runs without user interaction. This means that once the command "`./smarts295bat.exe`" is issued, the program will read the default input file ('smarts295.inp.txt'), calculate the solar spectrum, and pass the results to the default output files (e.g. 'smarts295.out.txt', 'smarts295.ext.txt', and 'smarts295.scn.txt'). The file 'smarts295bat.exe' has been produced by activating the command "batch=.TRUE." (line 188 of the code) and recompiling the code. Move or copy this new executable to the SMARTS_295_Linux folder. If you wish to avoid typing the "`./`" before the executables and you are using csh or tcsh, you can add the following line in your "`.login`":

```
set PATH = ( $PATH  /install_path/SMARTS_295_Linux).
```

Scripting capabilities
Unix offers a series of shell commands that can enhance the batch capabilities of the SMARTS code. A sample shell file, called 'smarts295script.csh', is provided in the Source_code folder. It uses the 'smarts295bat.exe' executable compiled for pure batch mode (see previous paragraphs). The content of this file is listed below. A user familiar with shell scripting can take further advantage of the batch mode and create even more complicated scripts.

```
#!/bin/csh
# Use "source smarts295script.csh" to execute this file
# or
# "chmod u+x smarts295script.csh"
# and then
# "./smarts295script.csh"
#
foreach i (01 02 03 04 june july 20050914)
cp -f smarts_$i.inp.txt smarts295.inp.txt
./smarts295bat.exe
mv smarts295.out.txt smarts_$i.out.txt
mv smarts295.scn.txt smarts_$i.scn.txt
mv smarts295.ext.txt smarts_$i.ext.txt
end
```

This example script assumes that seven input files are present in the root folder, with a variety of naming conventions: 'smarts_01.inp.txt', 'smarts_02.inp.txt', 'smarts_03.inp.txt', 'smarts_04.inp.txt', 'smarts_june.inp.txt', 'smarts_july.inp.txt', and 'smarts_20050914.inp.txt'.
When the script is activated (after it is moved to the root of the main folder, and the instructions from the comments in the first few lines of the script are followed), the content of each of these files is copied to the default input file with the help of the "`cp`" command (copy). Then the 'smarts295bat.exe' executable reads this input file, make the appropriate calculations and pass the output to the default files. In order not to overwrite the default output files, they are renamed with the help of the "`mv`" command (move). Once the "end" statement is reached, the "`foreach`" command will move to the next argument. If all arguments are used, the user will be returned to the standard prompt.
It is clear that the user can change the arguments of the "`foreach`" command to suit his/her needs. All text files and the executable are assumed to reside in the root directory of the code.

What's next?

At this point, you are certainly eager to customize the input file to obtain the results you really want. Use an appropriate text editor, such as vi, nedit or xemacs, to modify the input file ('smarts295.inp.txt') according to your needs, based on the detailed explanations provided in Section 6 of the User's Manual. Save it as TEXT, with the same filename, or any other name of your choice, in the format 'any_file_name.inp.txt' (e.g., 'Test_for_July.inp.txt'). Note that the default input filename is 'smarts295.inp.txt', and remember that filenames are case sensitive.

Acknowledgement

Dr. Fotis Mavromatakis has been instrumental in preparing the Linux version and this QuickStart Guide. His volunteer help is deeply appreciated.