Introduction

This document provides a description of the code which can be used to include the random effects of turbulence and fire-spotting by post-processing the output from an existing wild-fire simulator. The present code in the folder `LSfire` provides the proof of concept with an existing wildfire simulator : ForeFire [2] and also with a basic wild-fire simulator based on the level set method and Rothermel model for the Rate of Spread (RoS) [1].

Compiling the code

1. To compile ForeFire, see `firefront/readme.md`.
2. To compile the code in `LSfire` see the following steps:
   (a) For Linux machines, compile the code by running the makefile. Also see the file `LSfire/README.md`.
   (b) A successful run should create an executable `LSFire+`.
   (c) The compilation has been tested only with a `gcc 4.8.2` compiler. No shared libraries are required. The only libraries which are required (in addition to OpenMP, which is totally optional, and to standard libraries) are `firelib` and `lsmlib`, which are both provided in the folders `firelib-1.0.4` and `lsmlib-1.0.1`, respectively. `firelib-1.0.4` is compiled by the makefile (it is a very small library, just a couple of files); `lsmlib` is not compiled (the `*.a` files are directly linked by the makefile without compiling them). You need to recompile `lsmlib` only if you are using an operating system and/or architecture different than Ubuntu 14.04 on a `x86_64` architecture.

Test cases with level set method

1. All the input data to run the model is given through the `get_input_fire` function, located in the file `input_fire.c`. There is also the possibility to provide some of the most useful input data through the command line. Note that only few of all the input data can (at present) be provided by command line arguments. If you need to change some of the other values, you will have to edit the `input_fire.c` file and re-compile. Luckily, it takes only few seconds to recompile the code.
2. In order to run the code and see if it produces the right results (and also to provide some basic examples of usage), three test cases have been provided. The user has freedom to choose the type of simulation by defining one simple flag:

flag	Operation
0	No turbulence and No Fire-spotting
1	Only turbulence
2	Redundant, not used
3	Turbulence + Fire-spotting

3. the script `script_1.sh` can be used to run the simulations with the basic wildfire simulator based on level set method and Rothermel model.

4. The user can also define the fire-obstacles (fuel break zones) in the input file, `input_fire.c`.

5. Python script `makefig_1.py` utilises `matplotlib` to reproduce the results as figures.

6. The user can also edit the subroutine `get_RoS` in `LSFire+.c` to include different choices of RoS.

Note: The flag 4 and the input file `input_adiff.c` can be safely ignored in the context of wild-fire propagation. They are used to define the front propagation in anomalous diffusion as described in the article: *A. Mentrelli and G. Pagnini, Front propagation in anomalous diffusive media governed by time-fractional diffusion, Journal of Computational Physics 293 (2015), 427-441.*

Test cases with ForeFire

The python script `swig/FFpost_LS.py` combined with the script `LSfire/FF_LS.sh` provides an interface to post-process the moving interface from ForeFire to include the random fluctuations. This can be termed as a post processing of the output from ForeFire [3]. As a future task, the random effect formulation will be included inside the framework of ForeFire itself.

1. All the input data should be defined in both `FFpost_LS.py` and `input_fire.c`. After editing `input_fire.c`, the code should be compiled again (See step 1 in previous section).

2. The user can choose the simulation according to the flag in `FF_LS.sh`:

flag	Operation
5	Only turbulence
6	Turbulence + Fire-spotting

3. Presently the idealised cases have been defined by using a constant RoS and non-variable wind velocity. The RoS is assumed as 3% of the average wind speed. The subroutines `get_RoS` in `LSFire+.c` and `UnsteadyBalbi.cpp` can be edited to accommodate these changes. Make sure to compile the code again to realise the changes.

4. Running the turbulent case: set flag in `FF_LS.sh` as 5 and run the python script `swig/FFpost_LS.py`. This script also uses `matplotlib` to produce a plot showing the fire contours at every 20 min (approx).

5. Running fire-spotting : set flag in `FF_LS.sh` as 6 and run the python script `swig/FFpost_LS.py`.

6. Changing the mean wind speed (U) or the grid size (dx) affects the CFL condition and the time step of ForeFire. The time step in `swig/FFpost_LS.py` should be defined according to $dx/(2 * U)$.

Caution

1. Make sure the definition of the domain size and wind speed and other input parameters are in agreement between the two files: `input_fire.c` and `swig/FFpost_LS.py`.

2. For test cases with ForeFire, the RoS should be defined identically both in `LSFire+.c` and `UnsteadyBalbi.cpp`.

3. The inputs in the `input_fire.c` should be provided in feets instead of metres. Necessary precautions should be taken to avoid a mismatch.

4. Make sure that a folder named `data/FF_LS` exists in the path `firefront/swig` before running the code.

Contact

The users can address their queries, or report bugs to the following:

G. Pagnini,
Ikerbasque Research Fellow,
BCAM-Basque Centre for Applied Mathematics,
Bilbao, Basque Country, Spain,
gpagnini@bcamath.org

I. Kaur,
BCAM-Basque Centre for Applied Mathematics,
Bilbao, Basque Country, Spain,
ikaur@bcamath.org

References

[1] G. Pagnini, A. Mentrelli, Modelling wildland fire propagation by tracking random fronts, Nat. Hazards Earth Syst. Sci. 14 (2014) 2249–2263.

[2] J. B. Filippi, F. Morandini, J. H. Balbi, D. Hill, Discrete event front tracking simulator of a physical fire spread model, Simulation 86 (2009) 629–646.

[3] I. Kaur, A. Mentrelli, F. Bosseur, J. B. Filippi, G. Pagnini, Wildland fire propagation modelling: A novel approach reconciling models based on moving interface methods and on reaction-diffusion equations, in: J. Brandts, S. Korotov, M. Křížek, K. Segeth, J. Šístek, T. Vejchodský (Eds.), Proceedings of the International Conference on Applications of Mathematics 2015; Prague, Czech Republic, 18–21 November (2015), Institute of Mathematics – Academy of Sciences, Czech Academy of Sciences, Prague, Czech Republic, 2015, pp. 85–99.

January 21, 2016