

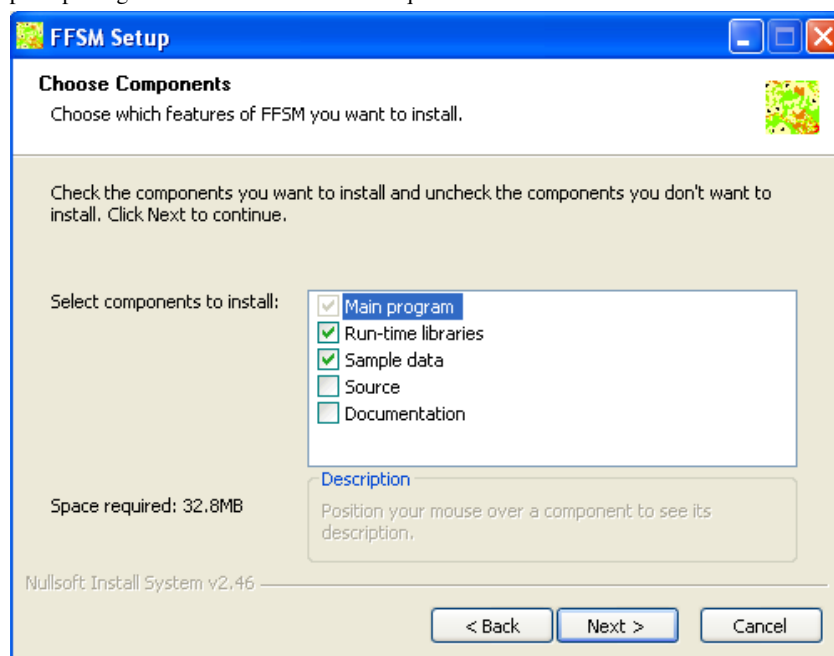
## Installing and running FFSM++

### Installing the compiled software

FFSM, as any model developed in C++, must be firstly compiled for the specific environment in use. For windows users a pre-compiled version of the model is provided in a self-installation package that take care of installing the dependencies.

Just run the installer, choose the components to install (next Figure) and the FFSM program will be available within the installed programs.

For Linux users pre-compiled packages can be released at user's request.



### Running the model

#### Gui mode

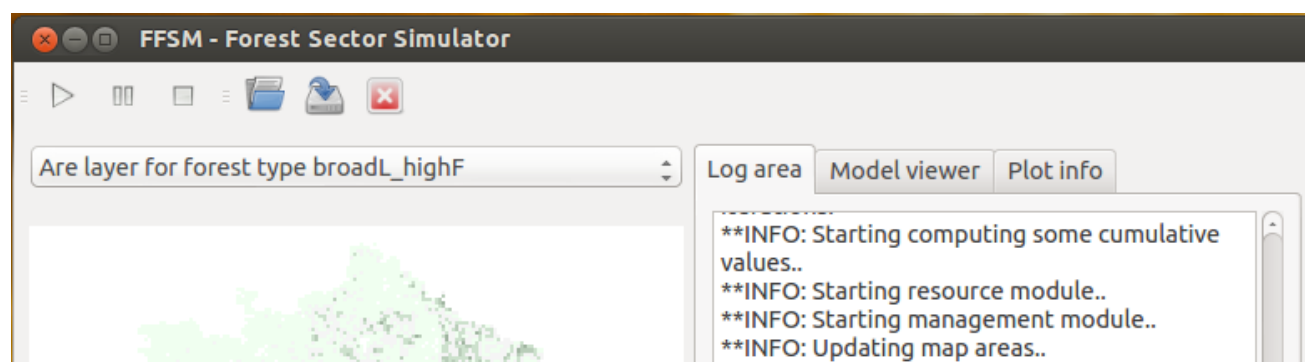
Clicking on the FFSM icon allows the software to open in GUI mode (next Figure). From here we can load a new data input file (or leave the default one at data/ffsmInput.ods) and start the simulation with the “play” button”.

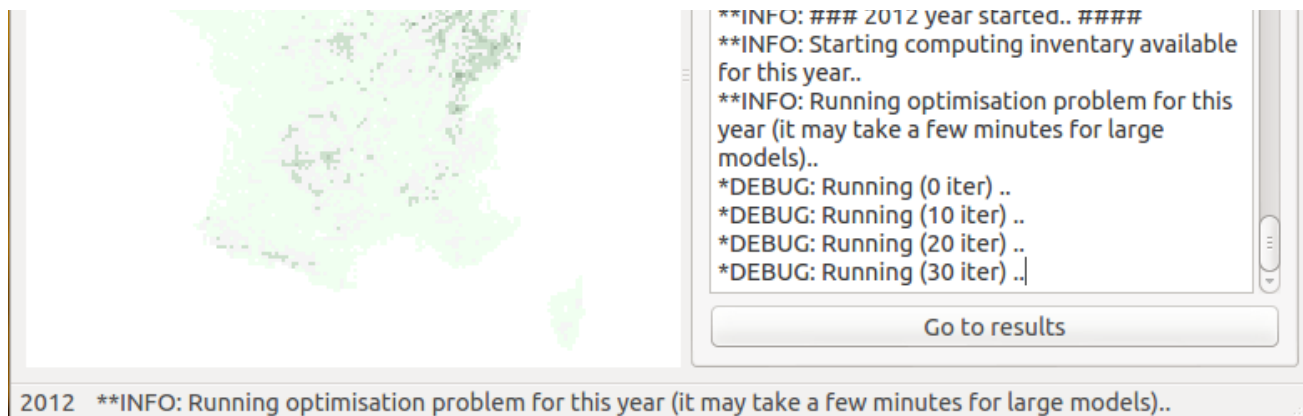
We will then be presented with a pop-up to choose between the available scenarios in the loaded input file and continue.

During the proceeding of the simulation we can observe its status loading any of the layers using the drop-down menu, analyse the live logs or retrieve any pixel detail right clicking on it.

At any time we can either pause the simulation or abort it.

At the end of the simulation, provided that we have a compatible spreadsheet application with macro enabled, we can click on the results button to analyse the results directly on it.





## Command line options

FFSM has also a command line mode that is activable using the DOS or unix prompt.

It accepts the following options:

Short option version	Long option version	Description
-h	-help	Print the help
-c	-console	Run in console mode (no gui, default: false)
-i [input_file_name]	-input_file [input_file_name]	Input file (relative path, default: 'data/ffsmInput.ods')
-s [scenario_name]	-scenario [scenario_name]	Scenario name (default: the first defined in the input file)

Command line mode is very useful for unattended simulations of the model, e.g. when working on sensitivity analysis that requires the run of many similar scenarios.

In such case it is advisable to prepare the desired scenarios in the input file and then run the model sequentially in a batch mode changing only the -s switch.

While the program use a single process, in Linux you can use the following scripts to run several scenarios in parallel, hence benefitting of modern multi-core processes:

### run\_single\_scenario.sh

```

#!/bin/bash

#-----
# Shell script to run a single ffm scenario, where the scenario name is the first argument.
#-----

./ffsm -c -s $1 > logs/${1}.txt
echo "Ended running scenario " $1

```

### runscenarios.sh

```

#!/bin/bash

#-----
# Shell script to run ffm scenarios
#-----

# Safe parallel..
./ffsm -c -s scenarioName1 > logs/scenarioName1.txt &
./ffsm -c -s scenarioName2 > logs/scenarioName2.txt &

# Running the same scenario (e.g. for repetitions) in parallel is safe as long as newRandomSeed
# is set to true and outputSingleFile is set to false..
for i in {1..30}
do
    ./ffsm -c -s randomSpace 1> /dev/null 2> /dev/null &
done

# A better approach to run scenarios in parallel is using GNU parallel: you can set the maximum
# number of processes and then the jobs are put in a queue.
# In that case run this script as:
# parallel --jobs <n of jobs> -a runscenarios.sh
# and put something like this in the script

./run_single_scenario.sh 'scenarioName1'
./run_single_scenario.sh 'scenarioName2'

```

