ELMFIRE 1.1 User Guide

Chris Lautenberger – Cloudfire Inc.
Nikolaos Kalogeropoulos – Cloudfire Inc.
Maria Theodori – Cloudfire Inc.
Kasra Shamsaei – Cloudfire Inc.
Majid Bavandpour – Cloudfire Inc.
Yiren Qin – UC Berkeley
Dwi Marhaendro Jati Purnomo – UC Berkeley
Daniel San Martin – UC Berkeley
Michael J. Gollner – UC Berkeley
Arnaud C. Trouve – University of Maryland

Contact: `nick@cloudfire.com`

June 2026

Contents

Chapter 1

Computational Guide

1.1 Introduction

1.1.1 Installation

This guide will go through the steps of installing ELMFIRE to a linux system. Windows users can use ELMFIRE by using a WSL installation, allowing the use of a linux subsystem within windows. More details on installing WSL can be found in https://www.geeksforgeeks.org/installation-guide/how-to-install-wsl2-windows-subsystem-for-linux-2-on-windows-10/ (make sure to stop at step 7, before the virtual machine section). This installation guide was tested on a clean install of Ubuntu 24.04

Install Prerequisites:

The following commands will install packages needed to build and run ELMFIRE:

`sudoapt-getupdate\&\&sudoapt-getupgrade-y`

`sudoapt-getinstall-ybccsvkitgdal-bingfortrangitjqlibopenmpi-devopenmpi-binpigzpython3python3-pipunzipwgetzip`

Several Python libraries / packages, including three related to Google Remote Proecedure Call (gRPC), are needed to run the CloudFire microservices that provide ELMFIRE with fuel, weather, and ignition geospatial data. These can be installed system-wide as follows:

`sudopip3installgoogle-api-python-clientgrpciogrpcio-toolspython-dateutil--break-system-packages`

Alternatively, virtualenv can be used to prevent forcing a system-wide install as follows:

`sudoapt-getinstallpython3-virtualenv`

`virtualenv\$HOME/virtualenv/elmfire`

`source\$HOME/virtualenv/elmfire/bin/activate/`

`\$HOME/virtualenv/elmfire/bin/pip3installgoogle-api-python-clientgrpciogrpcio-toolspython-dateutil`

Clone ELMFIRE Github repository:

The current ELMFIRE repository can be cloned as follows:

`gitclonehttps://github.com/lautenberger/elmfire.git`

Since this will clone the current repository, including all recent commits, for use in production environ-

ments a user may want to clone the latest stable release / branch instead, i.e.:

`gitclone--branch2025.1002--single-branchhttps://github.com/lautenberger/elmfire.git`

Set environment variables

ELMFIRE uses four environment variables:

1. `ELMFIRE_SCRATCH_BASE`: Full path to a scratch directory where the user has read/write access.
2. `ELMFIRE_BASE_DIR`: Full path to the directory to which the ELMFIRE Git repository was cloned (i.e., the directory containing README.md).
3. `ELMFIRE_INSTALL_DIR`: Full path to the directory where the ELMFIRE executable files will be installed (default: `$ELMFIRE_BASE_DIR/build/linux/bin`).
4. `CLOUDFIRE_SERVER`: IP address of Cloudfire microservices server (this should be set to worldgen.cloudfire.io, more on this later).

The easiest way to specify these variables is by exporting them from your `~/.bashrc` file. For example, one could pico `~/.bashrc` and then add the following lines (replacing `/scratch/clauten`, `/home/clauten/elmfire`, etc.):

```
export ELMFIRE_SCRATCH_BASE=/scratch/clauten
export ELMFIRE_BASE_DIR=/home/clauten/elmfire
export ELMFIRE_INSTALL_DIR=$ELMFIRE_BASE_DIR/build/linux/bin
export CLOUDFIRE_SERVER=worldgen.cloudfire.io
export PATH=$PATH:$ELMFIRE_INSTALL_DIR:$ELMFIRE_BASE_DIR/cloudfire
```

These environment variables will be set on the next login or after sourcing the newly-edited file (`~/.bashrc`).

Build ELMFIRE executables:

ELMFIRE and its postprocessing tool can be built as follows:

```
cd $ELMFIRE_BASE_DIR/build/linux
./make_gnu.sh
```

Unless an error occurs, this will build the executables `elmfire_VERSION` and `elmfire_post_VERSION` (where version is, for example, 1.1) and copy them to `\$ELMFIRE_INSTALL_DIR`. If this directory is not in the user's $PATH it should be added at this time. Note that two debug executables are also built.

At this point ELMFIRE has been installed in your local machine. Run the tutorial cases in `elmfire/tutorials/` to get started and verify everything was installed correctly.

1.1.2 First Run

ELMFIRE models the spread of a wildfire across a landscape and provides insightful outputs regarding the spread of the fire itself, the computation process or other post-processed results. The minimum information required to run an ELMFIRE simulation are:

- Topography Rasters (Elevation, Slope, Aspect)
- Fuel Raster
- Fuel Dictionary
- Canopy Values (Canopy Height, Canopy Base Height, Canopy Bulk Density, Canopy Cover)
- Ignition Location

- Transient Weather Values (Wind Magnitude, Wind Direction, 1-hour Fuel Moisture, 10-hour Fuel Moisture, 100-hour Fuel Moisture, Live Herbaceous Fuel Moisture, Live Woody Fuel moisture).
- Geospatial Domain Information (projection, cell size, lower-left X and Y coordinates).

The details of an ELMFIRE simulation, and the main way of using ELMFIRE, is through a .data text file, editable with a standard text editor program. For this guide, we will be naming this file elmfire.data . Below is the minimum required elmfire.data file to start an ELMFIRE simulation. Inputs are dictated through namelists (such as `&INPUTS`) and inputs (such as `ASP_FILENAME`). All inputs must be within their appropriate namelist. The order of inputs or namelists is not important.

```
! Denotes a comment, everything after ! is ignored

!The inputs namelist contains parameters relating to parsing input data and setting
!relevant parameters. *_DIRECTORY inputs denote parent folders, and *_FILENAME (for
!weather, fuel and topography) refers to filename.tif. So, LANDSCAPE_FILENAME = 'landscape' means
!the aspect raster is /inputs/landscape.tif.
&INPUTS     / Define all the input locations and parameters
FUELS_AND_TOPOGRAPHY_DIRECTORY = './inputs'
LANDSCAPE_FILENAME             = 'landscape'
ADJ_FILENAME                   = 'adj'
PHI_FILENAME                   = 'phi'
WEATHER_DIRECTORY              = './inputs'
WS_FILENAME                    = 'ws'
WD_FILENAME                    = 'wd'
M1_FILENAME                    = 'm1'
M10_FILENAME                   = 'm10'
M100_FILENAME                  = 'm100'
LH_MOISTURE_CONTENT            = 60
LW_MOISTURE_CONTENT            = 90
FOLIAR_MOISTURE_CONTENT        = 80
/

!The outputs namelist specifies details around the data and format that are going to
!be returned. In the example here, we only want the spread rate of the fire and the
!time of arrival. All other outputs (other than general statistics) are assumed false.
&OUTPUTS
OUTPUTS_DIRECTORY    = './outputs'
DUMP_SPREAD_RATE     = .TRUE.
DUMP_TIME_OF_ARRIVAL = .TRUE.
CONVERT_TO_GEOTIFF   = .TRUE.
/

!The time control namelist dictates everything about the duration, step size and
!temporal fidelity of the simulation. For now the only thing that needs to be specified
!is the duration of the simulation.
&TIME_CONTROL
```

```
SIMULATION_TSTOP = 36000
/

! The simulator namelist contains adjustment parameters, constants and other variables
!around the ELMFIRE solver itself. It is also where the ignition spots are specified.
&SIMULATOR
NUM_IGNITIONS = 1
X_IGN(1)      = 0.0
Y_IGN(1)      = 3000.0
T_IGN(1)      = 3000.0
/

! The miscellaneous namelist contains other important parameters, most critically the
!path to GDAL in the user's computer, and a scratch directory for intermediate file
!saves.
&MISCELLANEOUS
PATH_TO_GDAL                  = '/usr/bin'
SCRATCH                       = './scratch'
/
```

Then, this file is saved as elmfire.data in the same directory as the /inputs folder (containing out fuel, topography and weather information), the /scratch temporary file folder, and the /outputs folder. These folders are not automatically created by ELMFIRE and should be created by the user.

Once all the files and folders are in place, the user can start ELMFIRE using the command (in a terminal in the main folder with the inputs/ , scratch/ and outputs/ folders):

```
elmfireinputs/elmfire.data
```

ELMFIRE will then start processing the input rasters, and then will run the main ELMFIRE routine. The expected output for the input file above would look like:

```
 ELMFIRE 1.1
 Reading &MISCELLANEOUS namelist group
 Reading &INPUTS namelist group
 Reading &OUTPUTS namelist group
 Reading &COMPUTATIONAL_DOMAIN namelist group
 Reading &TIME_CONTROL namelist group
 Reading &SIMULATOR namelist group
 Reading &WUI namelist group
 Reading &CALIBRATION namelist group
 Reading &SUPPRESSION namelist group
 Reading &SPOTTING namelist group
 Reading &SMOKE namelist group
 Reading &MONTE_CARLO namelist group
 Reading headers for fuels/topography and weather rasters
 Setting up shared memory, part 1
 Reading weather, fuel, and topography rasters
 Reading fuel and topography rasters
```

```
 Using input Phi grid as ignition source
 Determining number of cases to run
 Allocating additional rasters
 Setting up shared memory, part 2
 Setting up statistics arrays
 ELMFIRE is running each ensemble member
 /usr/bin/gdal_translate -ot Float32 -co "COMPRESS=DEFLATE" -co "ZLEVEL=9" -a_srs "+proj=utm +zone=35 +
Input file size is 126, 126
0...10...20...30...40...50...60...70...80...90...100 - done.
/bin/rm -f ./scratch/vs_0000001_0259832.bil
/bin/rm -f ./scratch/vs_0000001_0259832.hdr
/usr/bin/gdal_translate -ot Float32 -co "COMPRESS=DEFLATE" -co "ZLEVEL=9" -a_srs "+proj=utm +zone=35 +da
Input file size is 126, 126
0...10...20...30...40...50...60...70...80...90...100 - done.
/bin/rm -f ./scratch/time_of_arrival_0000001_0259832.bil
/bin/rm -f ./scratch/time_of_arrival_0000001_0259832.hdr
Meteorology band      1: Case #       1 complete.  Fire area:   3310.3 acres.
 End of simulation reached successfully. Shutting down.
```

The above output shows an ELMFIRE simulation completed with no errors. Tutorial 1 contains a shell file that creates all the necessary files and folders for a simple simulation, and runs elmfire.

The following sections will go into depth of the different variables, options and switches within ELMFIRE.

1.2 Surface Fire Spread

Surface fire spread is the default fire spread mode of ELMFIRE, and will always run when the minimum inputs are provided. The default surface fire rate of spread model is the Rothermel model [23]. Alternative spread models will be described in following sections.

The first step in setting up ELMFIRE is to specify the geospatial input files. A distinction is made between weather files, and fuel and topography files. If the user sources a landscape files, which contains the 8 landscape and fuel rasters as separate bands (the standard that Farsite and Flammap use, and that can be directly obtained from LANDFIRE), it can be used directly in ELMFIRE. Older versions of the landscape files were provided in .lcp format, this format is slowly being deprecated and ELMFIRE does not support it. Flammap does contain tools to convert .lcp landscape files to .tif multiband raster files. To use landscape files, you will just require:

1. `LANDSCAPE_FILENAME`: Multiband geotiff landscape file (16-bit integers)
2. `ADJ_FILENAME`: Surface spread rate adjustment factor (32-bit float)
3. `PHI_FILENAME`: Initial (level set variable) field (32-bit float)

Filenames should be specified without a suffix (e.g., 'lcp', not 'lcp.tif' because ELMFIRE automatically appends the .tif suffix to filenames (if they are virtual raster (.vrt) files, this behavior can be changed by setting `VRT_INSTEAD_OF_TIF=.TRUE.`). Some checks are implemented to ensure that all the rasters are consistent in terms of cellsize, overall extent and projection. The weather inputs and the landscape inputs should be consistent amongst each other, but the weather inputs can be coarser than the landscape inputs.

Users can also specify specific geotiff files for each fuel and landscape raster instead of specifying a

combined landscape file. The required files then become:

1. ASP_FILENAME: Topographic aspect in degrees (16-bit integer)
2. CBD_FILENAME: Canopy bulk density in units of 100 kg per meter cubed (16-bit integer)
3. CBH_FILENAME: Canopy base height in units of 10 meters (16-bit integer)
4. CC_FILENAME: Canopy cover in units of percent (16-bit integer)
5. CH_FILENAME: Canopy height in units of 10 meters (16-bit integer)
6. DEM_FILENAME: Digital elevation model data in units of meters (16-bit integer)
7. FBFM_FILENAME: Fire behavior fuel model file (16-bit integer)
8. SLP_FILENAME: Topographic slope in degrees (16-bit integer)
9. ADJ_FILENAME: Surface spread rate adjustment factor (32-bit float)
10. PHI_FILENAME: Initial (level set variable) field (32-bit float)

All of the above rasters, except for Phi and Adjustment, need to be Int16 variable data type. Phi and Adjustment must be Float32. The canopy parameters CBH and CH are in units of 10 meters, and CBD is in 100 kg per meter cubed. This is consistent with how these values are provided and reported in the US LANDFIRE database. If the input data of these files are in meters and kg per meter cubed respectively, the switches CBD_TIMES_100, CBH_TIMES_10 and CH_TIMES_10 can be set to .FALSE.. Note that the input rasters would then have to change change variable type from Int16 to Float32. Similarly, CC is read in percent as standard, but this can be changed by setting CC_IN_PERCENT=.FALSE.. Similar switches exist for DEAD_MC_IN_PERCENT and LIVE_MC_IN_PERCENT.

All the above is also true for the weather rasters, all of which need to be Float32. The weather values are specifically:

1. WS_FILENAME: 20-ft wind speed in mph
2. WD_FILENAME: 20-ft wind direction in degrees
3. M1_FILENAME: 1-hour dead fuel moisture content in %
4. M10_FILENAME: 10-hour dead fuel moisture content in %
5. M100_FILENAME: 100-hour dead fuel moisture content in %

If the rasters specified above are single-band, then wind/weather conditions are assumed to be unchanging for the duration of the simulation. Transient wind/weather streams can be provided as input by using multi-band, or "stacked", rasters. In that case, the parameter DT_METEOROLOGY should be specified on the &INPUT line. DT_METEOROLOGY is the time interval (in seconds) between bands in the weather rasters. DT_METEOROLOGY is commonly set to 3600 seconds due to the use of hourly reanalysis or forecast products to drive fire spread simulation. If wind speed is provided at 10 meters instead of 20 feet, the parameter WS_AT_10M=.TRUE. should be set to .TRUE.. Similarly, if the wind speed is provided in kph instead of mph, the parameter WS_IN_KPH=.TRUE. should be set to .TRUE.. Note that neither of these options affect the output format of the results.

By default, live fuel moisture content (herbaceous, woody, and foliar) are spatially uniform and temporally unchanging. They are specified in the &INPUTS namelist group via the keywords LH_MOISTURE_CONTENT, LW_MOISTURE_CONTENT, and FOLIAR_MOISTURE_CONTENT, respectively. Spatially varying live fuel moisture content can be specified by setting USE_CONSTANT_LH=.FALSE., USE_CONSTANT_LW=.FALSE., and USE_CONSTANT_FMC=.FALSE.. This directs ELMFIRE to read live fuel moisture from the rasters specified by MLH_FILENAME, MLW_FILENAME, FMC_FILENAME (live herbaceous, live woody, and foliar, respectively). Even though live fuel moistures will typically change very little during a fire forecast (or hindcast), ELMFIRE expects the live fuel moisture rasters to have the same number of bands as the weather rasters, with the time interval between bands specified by DT_METEOROLOGY.

ELMFIRE automatically interpolates between specific weather parameter steps, during the level set mode. The default behavior is a simple linear interpolation. Bilinear interpolation can also be used by turning `WX_BILINEAR_INTERPOLATION=.TRUE.` in the `&SIMULATOR` namelist.

The Rothermel model requires midflame windspeed for its calculation. A wind adjustment factor is automatically calculated based on the algorithms in BEHAVE and Farsite. FARSITE and ELMFIRE assume a crown ratio of 1 for their calculation. This can be manually changed if needed through `CROWN_RATIO`.

1.2.1 Alternative Spread Models

While the standard model to estimate the rate of spread of the simulated wildfire is the Rothermel model [23, 5], alternative models can be integrated into ELMFIRE. A surface rate of spread model that has been integrated to ELMFIRE is the Canadian Fire Behavior Prediction model (FBP), part of the Canadian Forest Fire Danger Rating System (CFFDRS). The FBP is based on a series of developments and submodels developed by the Canadian Forestry Service, including the Forest Fire Weather Index system (FWI) [30], the Fire Behavior Prediction system (FBP) [1], and the 2009 corrections and updates [32]. The model has been implemented operationally to the now deprecated Prometheus software [28] and the new WISE software.

To activate the canadian model in ELMFIRE, specify SURFACE_SPREAD_MODEL = 'CFFDRS' in the `&INPUTS` namelist. By default, SURFACE_SPREAD_MODEL is set to 'ROTHERMEL'. The model then replaces the Rothermel maximum rate of spread calculation with the one specified in the FBP, including prolonged drought effects represented by the Build-Up Index (BUI). Calculating the BUI requires the Drought Code (DC) and Duff Moisture Code (DMC), both of which are sub-indices of the Fire Weather Index (FWI). These two indices are continuously updated (meaning each new measurement requires the previous measurement), and require daily weather data, specifically midday Temperature and Humidity, and 24h precipitation. As such, ELMFIRE requires additional inputs for the CFFDRS model to function.

The additional inputs are all in the `&INPUTS` namelist. The first is START_DC, the Drought Code at the closest midday before the fire ignition. This means that if the fire starts at 16:00, the START_DC should be specified at 12:00 of the same day. If the fire starts at 09:00, the START_DC should be specified at 12:00 of the previous day. The second new input is the starting Duff Moisture Code, START_DMC, that functions similarly to START_DC. The third input is DAILY_WEATHER_FILENAME, which specifies the full name of the file containing the daily midday weather stream, and should be within the WEATHER_DIRECTORY folder. The structure of the weather stream is shown below:

```
DAILY,TEMP,RH,PRECIP
01/01/2025,30.2,25.4,0
02/01/2025,28.2,22.3,0
03/01/2025,28.8,31.5,0
04/01/2025,26.5,39.8,3
05/01/2025,29.4,35.1,0
...
```

The temperature is specified in Celsius, and the precipitation in mm. The date of each weather reading is specified in DD/MM/YYYY and is important, as the exact month of each measurement affects parameters within the DMC and DC calculations. Note that the dates that are specified in the weather file do not, as of yet, overwrite or affect CURRENT_YEAR and HOUR_OF_YEAR parameters in the `&TIME_CONTROL` namelist. HOUR_OF_YEAR must be set as an input, so the rest of the program has a measure of the current simulation hour.

Also note that the standard Canadian FBP fuels are interpreted from the FBFM_FILENAME file, so no change is needed. One change that might be desirable is the omission of Canopy Bulk Density and Canopy Base Height inputs, as the FBP has hardcoded these values as surface fuel parameters. If CH_FILENAME and CBD_FILENAME are not specified, the standard values found in [28] will be used.

All other parts of ELMFIRE, including Crown Fire, Suppression, and Spotting, will work with the CFFDRS with no changes required.

1.2.2 Grid Declination

In rare cases, direction rasters (specifically wind direction (WD) and aspect (ASP)) may have absolute north different from the absolute north of the domain. If this is the case, the difference between the true north of the simulation and the north of the wind direction and aspect measurements is called the grid declination. This can be accounted for in ELMFIRE, by setting the GRID_DECLINATION float variable. Then the user should also set ROTATE_WD to .TRUE. if the WD grid needs adjustment, and ROTATE_ASP to .TRUE. if the Aspect grid needs adjustment.

1.3 Computational domain size, extents, and resolution

Older versions of ELMFIRE required that the computational domain parameters be explicitly set in the input file via the `&COMPUTATIONAL_DOMAIN` namelist group. This group has been removed in newer versions of ELMFIRE, and the parameters below are determined automatically using GDAL. The parameters have been kept here for clarity. The computational domain is specified by the following parameters:

- A_SRS: Projection of output files. Typically, a proj string would be used, i.e. A_SRS = 'EPSG:32610'.
- COMPUTATIONAL_DOMAIN_CELLSIZE: spatial resolution of the computational domain, uniform in the x and y directions. This is commonly set to the spatial resolution of the input fuels layers (often 30 m) but can be set to a smaller value (e.g., 10 m) if a more highly resolved simulation is desired.
- COMPUTATIONAL_DOMAIN_XLLCORNER: x-coordinate of the lower left corner of the fuels inputs.
- COMPUTATIONAL_DOMAIN_YLLCORNER: y-coordinate of the lower left corner of the fuels inputs.

1.4 Time

Parameters related to time (simulation duration, computational timestep, etc.) are specified via the `&TIME_CONTROL` namelist group. A sample `&TIME_CONTROL` namelist group with key inputs is shown below:

```
&TIME_CONTROL
SIMULATION_TSTART   = 0.0
SIMULATION_TSTOP    = 3600.0
SIMULATION_DT       = 5.0
SIMULATION_DTMAX    = 600.0
TARGET_CFL          = 0.4
DT_INTERPOLATE_M1   = 300.0
DT_INTERPOLATE_M10  = 3000.0
DT_INTERPOLATE_M100 = 30000.0
DT_INTERPOLATE_MLH  = 9E8
```

```
DT_INTERPOLATE_MLW  = 9E8
DT_INTERPOLATE_FMC  = 9E8
DT_INTERPOLATE_WIND = 300.0
/
```

Simulation start and stop times are specified via the keywords `SIMULATION_TSTART` and `SIMULATION_TSTOP`, respectively. These parameters have units of seconds, so a 12-hour simulation corresponds to `SIMULATION_TSTOP=43200`. The default value of `SIMULATION_TSTART` is 0 seconds, meaning computations start at $t = 0$ seconds. This is generally appropriate for simulations driven by idealized or synthetic weather data. For transient wind/weather/fuel moisture multi-band rasters, Band 1 always corresponds to $t = 0$ seconds in ELMFIRE. The duration of an ELMFIRE simulation, while explicitly dictated, can be randomized between the start and stop times, through `RANDOMIZE_SIMULATION_TSTOP`.

When simulating real fires driven by transient, often hourly, weather streams it is usually desirable to start a fire spread simulation at a time ¿ 0 seconds. Assuming that hourly weather fields are provided as input and fire's time of ignition is 14:20, `SIMULATION_TSTART` should be set to 1200.0, i.e., 20 minutes after the hour. In this particular case, Band 1 in all transient raster inputs should correspond to 14:00 and Band 2 should correspond to 15:00. The output files will contain rasters representing the unused weather bands, so some of the beginning or ending bands might be empty (or uniform). To crop the output rasters and only use the required weather bands, set `ONLY_READ_NEEDED_WX_BANDS=.TRUE.` in the `&INPUTS` namelist.

The initial timestep is specified with the `SIMULATION_DT` keyword. The timestep is automatically adjusted at runtime based on the Courant-Friedrichs-Lewy (CFL) conditions. The target CFL number can be specified by `TARGET_CFL`. Since the internal timestep will change during a simulation, an upper limit on the allowable timestep can be specified with the `SIMULATION_DTMAX` keyword.

Since wind/weather fields are often provided at hourly intervals but ELMFIRE's computational timestep is usually on the order of a few to at most tens of seconds, ELMFIRE uses linear interpolation to determine wind/weather/fuel moisture conditions at intermediate times. This interpolation can be computationally expensive, so the user is provided with some control over the interpolation frequency. The keywords `DT_INTERPOLATE_M1`, `DT_INTERPOLATE_M10`, and `DT_INTERPOLATE_M100` control the time between interpolations for 1-hour, 10-hour, and 100-hour fuel moistures. Wind speed and wind direction are controlled by `DT_INTERPOLATE_WS` and `DT_INTERPOLATE_WD`. As with other temporal inputs, units are seconds. The values quoted in the example `&TIME` namelist above correspond to the standard values of ELMFIRE.

The start of each ELMFIRE simulation, or equivalently the specific hour of the first weather band, is assumed to be midnight. This can be changed (primarily for writing outputs) by changing the value of BAND_ONE_HOUR_OF_YEAR.

1.4.1 Overnight Adjustment

The Rothermel model is well known to overpredict the rate of spread at night times, even when accounting for changes in dead fuel moisture content. ELMFIRE can account for this by multiplying its predicted rates of spread by an OVERNIGHT_ADJUSTMENT_FACTOR, set at 0.1 by default. This adjustment factor can be activated by setting USE_DIURNAL_ADJUSTMENT_FACTOR to .TRUE.. ELMFIRE then internally estimates the sunset and sunrise hours, applying the diurnal adjustment factor between those two times. SUNRISE_HOUR and SUNSET_HOUR can be set directly if known.

With the above parameters, the sunrise and sunset times are known either by input or calculation. The start time of the simulation (note the difference between start time of simulation and ignition start time)

can then be set by FORECAST_START_HOUR.

Empirical evidence exists that dictates the main wildfire activity escalates some hours after sunrise and subsides some hours after sunset. To allow the user to include this observation to ELMFIRE, the overnight adjustment factor is not applied strictly between sunset and sunrise. Instead, the user must specify a BURN_PERIOD_LENGTH in hours for the total fire duration, and a BURN_PERIOD_CENTER_FRAC, a fraction denoting whether the fire activity is concentrated in the early or late hours of the day.

All the above is best explained with an example. Assume that for a specific fire the sunrise and sunset hours are 07:00 and 21:00 respectively. With a BURN_PERIOD_LENGTH of 12 hours, and a BURN_PERIOD_CENTER_FRAC of 0.5, the mean hour of fire activity is $7 + 0.5 \times (21 - 7) = 14$. The diurnal adjustment factor then takes effect between $14 + 0.5 \times 12 = 20$ and $14 - 0.5 \times 12 = 8$. Note how this result is antithetical to the initial empirical assumption this function was designed for. With a BURN_PERIOD_CENTER_FRAC of 0.7, the mean hour of fire activity is $7 + 0.7 \times (21 - 7) = 16.8$. The diurnal adjustment factor then takes effect between $16.8 + 0.5 \times 12 = 22.8$ and $16.8 - 0.5 \times 12 = 10.8$, which follows our observation. Expert judgement is required to fine tune these parameters for an accurate result.

1.5 Fire Initialization

The two primary methods to initialize a fire spread simulation include point source ignitions and active fire perimeter initialization. These methods can be used concurrently, e.g. to simulate an active fire perimeter with additional point ignitions or spot fire initiation outside of the fire perimeter.

1.5.1 Point source ignitions

One or more point source ignitions can be specified on the `&SIMULATOR` namelist group via the keywords `X_IGN(:)`, `Y_IGN(:)`, and `T_IGN(:)` which respectively control point ignitions' x- and y-coordinates, and time of ignitions. As an example, the following lines specify two separate point source ignitions:

```
&SIMULATOR
NUM_IGNITIONS = 2
X_IGN(1)      = 1000.0
Y_IGN(1)      = 1000.0
T_IGN(1)      = 0.0
X_IGN(2)      = 2000.0
Y_IGN(2)      = 2000.0
T_IGN(2)      = 7200.0
```

The first ignition occurs at (x,y) = (1000.0, 1000.0) at simulation time 0.0 seconds and the second occurs at (x,y) = (2000.0, 2000.0) at simulation time = 7200.0 seconds. The keyword NUM_IGNITIONS specifies the total number of point source ignitions. Ignitions should be numbered sequentially starting at 1 and ending at NUM_IGNITIONS. The number of point source ignitions is currently limited to 100.

1.5.2 Line source ignitions

Line ignitions can be included in ELMFIRE by specifying the start and end point of each line segment, along with its ignition time. Similarly to the point ignition parameters, multiple line ignitions can be specified, each with its own ignition time. To do this:

```
&SIMULATOR
X_LINE_IGN_START(1)      = 1000.0
Y_LINE_IGN_START(1)      = 1000.0
X_LINE_IGN_END(1)        = 2000.0
Y_LINE_IGN_END(1)        = 2000.0
T_LINE_IGN(1)            = 7200.0
```

Note how the input NUM_IGNITIONS is no longer necessary. The above block creates a diagonal ignition starting from (1000, 1000) and extending to (2000, 2000), igniting all points inbetween starting at t = 7200 seconds.

1.5.3 Active fire perimeters

The fire front position is tracked by solving a conservation equation for the level set variable ϕ where unburned areas correspond to $\phi > 0$, burned areas correspond to $\phi < 0$, and the fire front position is the level set corresponding to $\phi = 0$. At the start of a simulation ELMFIRE reads the initial field from a 32-bit floating point raster with filename `PHI_FILENAME` as specified in the `&INPUTS` namelist group.

If there is no active fire at the start of a simulation, then all pixels in the PHI_FILENAME raster should be initialized with a single start value greater than 0 (usually 1.0). An initial fire front position can be specified by burning a value less than 0 (usually -1.0) into the PHI_FILENAME raster. All pixels with an initial value less than 0 will be marked as burned and fire spread will be initiated from those pixels.

Extinguished or "cold" segments of the fire perimeter can be simulated by modifying the fuel model raster to have a non-burnable fuel model in extinguished segments of the fire perimeter.

1.6 Outputs

A sample `&OUTPUTS` namelist group is shown below:

```
&OUTPUTS
OUTPUTS_DIRECTORY         = './outputs'
DTDUMP                    = 3600.
DUMP_FLIN                 = .TRUE.
DUMP_SPREAD_RATE          = .TRUE.
DUMP_SURFACE_FIRE         = .TRUE.
DUMP_TIME_OF_ARRIVAL      = .TRUE.
/
```

The keyword OUTPUTS_DIRECTORY specifies the directory to which ELMFIRE will write its output files. This output directory must exist at run-time; it will not be automatically created. Transient outputs, such as DUMP_SURFACE_FIRE, are written at intervals specified by DTDUMP. For diagnostic purposes, these outputs can instead be written at every computational time step by setting DUMP_EVERY_STEP = .TRUE. in the `&OUTPUT` namelist. The available rasterized outputs are summarized in Table 1.1; transient outputs are identified by `Transient` in the `Output` column. The corresponding dump times are saved to a CSV file named `dump_times_CASENUMBER.csv` in OUTPUTS_DIRECTORY. Rate of spread outputs are in units of ft/min as standard, but can be converted to m/min by setting SPREAD_RATE_IN_M = .TRUE. in the `&OUTPUTS` namelist.

In general, outputs to be dumped are specified using a logical keyword that begins with DUMP_*. The following is a full list of raster outputs and the logical keywords that control whether they are written to disk.

Most output parameters are written as standard during or after an ELMFIRE simulation. These parameters are given below. Outputs could be rasters of CSV files, showing either the final parameters or transient steps through the simulation, with intervals dictated by DTDUMP.

Table 1.1: The main dump routine outputs of ELMFIRE.

	MAIN DUMP ROUTINE			
Option	Description	Units	Type	Output
DUMP_TRANSIENT_ACREAGE	Total fire scar size over time	acres	CSV	Final
DUMP_SURFACE_FIRE	Status of surface fire (1 = burned, 0 = unburned)	Categorical	Raster	Transient
DUMP_CROWN_FIRE	Status of crown fire (0 = none, 1 = passive, 2 = active)	Categorical	Raster	Transient
DUMP_CRITICAL_FLIN	Surface Fireline intensity required for crown fire ignition	kW/m	Raster	Final
DUMP_SPREAD_RATE	Rate of Spread Magnitude	ft/min	Raster	Final
DUMP_SPREAD_DIRECTION	Rate of Spread Direction	degrees	Raster	Final
DUMP_HPUA	Heat per unit area	kJ/m^2	Raster	Final
DUMP_FLIN	Fireline Intensity	kW/m	Raster	Final
DUMP_FLAME_LENGTH	Flame Length	ft	Raster	Final
DUMP_REACTION_INTENSITY	Reaction intensity	kW/m^2	Raster	Final
DUMP_WS20	Wind speed		Raster	Transient
DUMP_WD20	Wind direction	degrees	Raster	Transient
DUMP_VELOCITY	Rate of Spread	ft/min	Raster	Final
DUMP_TIME_OF_ARRIVAL	Fire arrival time	s	Raster	Final
DUMP_TOTAL_DFC_RECEIVED	Accumulated heat flux due to direct flame contact (WU-E)	kJ/m^2	Raster	Final
DUMP_TOTAL_RAD_RECEIVED	Accumulated heat flux due to radiation (WU-E)	kJ/m^2	Raster	Final
DUMP_TRANSIENT_DFC	Transient DFC heat flux (WU-E)	kW/m^2	Raster	Transient
DUMP_TRANSIENT_RAD	Transient radiation heat flux (WU-E)	kW/m^2	Raster	Transient
DUMP_HRR_TRANSIENT	Transient heat release rate per unit area	kW/m^2	Raster	Transient
DUMP_TAGGED	Show currently 'Tagged' Cells (1 = Tagged)	Categorical	Raster	Transient
DUMP_PHI	Level set phi variable	-	Raster	Transient
DUMP_EMBER_FLUX	Accumulated ember flux per Monte-Carlo ensemble	pieces/m^2	Raster	Final
DUMP_EMBER_FLUX_TRANSIENT	Transient ember flux at the moment	pieces/m^2	Raster	Transient
DUMP_SPOTTING_OUTPUTS	Ember spatial and temporal travel data	coordinates, s	CSV	Final

Output filenames are hardcoded but should be readily discernable, e.g. fireline intensity outputs begin with flin_, time of arrival outputs begin with toa_, etc. Since ELMFIRE is sometimes used to run multiple cases as part of a Monte Carlo analysis or sensitivity analysis, a seven-digit sequential identifier is prepended to the name of each output raster, and the time at which the raster was dumped is appended to the filename.

If a monte carlo simulation set is done, especially with random or multiple ignitions, it is useful to plot

the scalar outputs on a map, depending on their ignition locations. Each run can be given an initial RUN_ID string, to differentiate it from potential other monte-carlo runs. The outputs that are possible in this case are given in the postprocessing table below.

Table 1.2: The postprocessed output routine options of ELMFIRE.

POSTPROCESS		
Switch	Description	Units
CALCULATE_TIMES_BURNED		sims
CALCULATE_FLAME_LENGTH_STATS	Output maximum and average flame length	ft
USE_FLAME_LENGTH_BINS	Split the flame length occurrence in bins	bin number
USE_EMBER_COUNT_BINS	Split the firebrand count occurrence in bins	bin number
DUMP_SURFACE_FIRE_AREA	Surface fire area occurring from each ignition spot	acres
DUMP_CROWN_FIRE_AREA	Crown fire area occurring from each ignition spot	acres
DUMP_FIRE_VOLUME	Fire volume occurring from each ignition spot	ac-ft
DUMP_AFFECTED_POPULATION	Affected population occurring from each ignition spot	input dependent
DUMP_AFFECTED_REAL_ESTATE_VALUE	Affected real estate value occurring from each ignition spot	input dependent
DUMP_AFFECTED_LAND_VALUE	Affected land value occurring from each ignition spot	input dependent
ACCUMULATE_EMBER_FLUX	Accumulated number of firebrand per unit area across all ensembles	pieces/m^2

The flame length and ember count bins are specified manually. That is, following the example below (same for the FLAME parameters).

```
USE_EMBER_COUNT_BINS = .TRUE.
NUM_EMBER_COUNT_BINS  = 3
EMBER_COUNT_BIN_LO(1) = 0
EMBER_COUNT_BIN_HI(1) = 10
EMBER_COUNT_BIN_LO(2) = 11
EMBER_COUNT_BIN_HI(2) = 25
EMBER_COUNT_BIN_LO(3) = 26
EMBER_COUNT_BIN_HI(3) = 50
```

Some additional output switches that might work different from the ones listed above are:

- TIME_AT_BURNED_ACRES(:): Output the time (mins) and runtime (s) taken for the fire to reach some amount of acres. Multiple acre values can be tracked, by including, for example, TIME_AT_BURNED_ACRES(1) = 500 and at another line TIME_AT_BURNED_ACRES(2) = 1000.
- DUMP_HOURLY_RASTERS:
- `DUMP_BINARY_OUTPUTS`: Save some outputs (ignition x, y) and additional parameters if `FULL_BINARY_OUTPUTS=.TRUE.` (time of arrival, flame length, velocity, crown fire) in binary format. Only for fire areas larger than `MINIMUM_AREA_FOR_BINARY_OUTPUTS=0` (standard). To save on space a limited, randomised percentage of timesteps can be written to the binary file by specifying `BINARY_OUTPUTS_DUMP_FRACTION=1.00` (default).

- DUMP_SPOTTING_IGNITION_TIME: (deactivated)
- DUMP_TIMINGS: Show individual processor timings
- If a TIMED_LOCATIONS_CSV file is provided in the &INPUTS namelist, then for all points specified in the csv (rows of point ID, X dimension, Y dimension), the arrival time of the fire will be saved in a new TIMED_LOCATION_TRACKER_1.csv.

ELMFIRE initially outputs all raster data as BIL rasters, a line-separated raster data type easy to use with FORTRAN. Then ELMFIRE converts these BIL rasters (along with their associated HDR header files) to geoTIFF files. In doing so, the BIL and HDR files are permanently deleted. This behavior can be changed by setting CONVERT_TO_GEOTIFF to .FALSE..

1.6.1 Virtual Weather Stations

Virtual weather stations can be specified by setting NUM_VIRTUAL_STATIONS to an integer greater than zero, and then specifying x and y coordinates of each station. As a simple example, data from two virtual stations would be written to disk by adding the following lines to the &OUTPUTS namelist group:

```
NUM_VIRTUAL_STATIONS = 2
VIRTUAL_STATION_X(1) = 12345.0
VIRTUAL_STATION_Y(1) = 67890.0
VIRTUAL_STATION_X(2) = 98765.0
VIRTUAL_STATION_Y(2) = 43210.0
```

The index inside the parentheses denotes the station number. For each virtual station a separate .csv file will be written that includes transient and fixed quantities as a function of time. These quantities are currently:

- Elevation
- Slope
- Aspect
- Canopy Bulk Density
- Canopy Base Height
- Canopy Cover
- Canopy Height
- Fuel model
- 1-hour dead fuel moisture
- 10-hour dead fuel moisture
- 100-hour dead fuel moisture
- Live herbaceous fuel moisture
- Live woody fuel moisture
- Foliar fuel moisture
- 20-ft wind speed
- 20-ft wind direction
- Phi (level set variable)

Note that virtual stations have been temporarily **disabled**.

1.7 Miscellaneous

Custom fuel models, of either vegetated or building fuels, can be specified by including related .csv files, following from fuel_models.csv. The csv files contain a series of fuel parameters. The location of these files is in the folder MISCELLANEOUS_INPUTS_DIRECTORY, and named FUEL_MODEL_FILE.csv and BUILDING_FUEL_MODEL_FILE.csv. The FUEL_MODEL_FILE.csv has the following structure:

```
1,FBFM01,.FALSE.,0.034,0,0,0,0,3500,9999,9999,1,12,8000
```

Corresponding to:

- Fuel Number: 1
- Fuel Name: FBFM01
- Dynamic Fuel: No
- 1h Fuel Load: 0.034 $lb\ ft^{-2}$
- 10h Duel Load: 0 $lb\ ft^{-2}$
- 100h Fuel Load: 0 $lb\ ft^{-2}$
- Live Herbaceous Fuel Load: 0 $lb\ ft^{-2}$
- Live woody Fuel Load: 0 $lb\ ft^{-2}$
- 1h Surface Area to Volume Ratio (SAV): 3500 $ft^2 ft^{-3}$
- Live and Dead Herbaceous SAV: 9999 $ft^2 ft^{-3}$
- Live Woody SAV: 9999 $ft^2 ft^{-3}$
- Fuel Load Depth: 1.0 ft
- Dead Fuel Moisture of Extinction: 12 %
- Heat Content: 8000 $BTU\ lb^{-1}$

Hardcoded values within ELMFIRE are the same for all fuel models:

- 10h SAV: 109 $ft^2 ft^{-3}$
- 100h SAV: 30 $ft^2 ft^{-3}$
- Particle Density: 32 $lb\ ft^{-3}$
- Total Mineral Content: 0.055
- Effective Mineral Content: 0.01

A building fuel model would look like:

```
1,ST01,300,400,10080,14400,360,25,10500,9,0.89,8,0.5,100,0.0,20
```

Corresponding to:

- Fuel Number: 1
- Fuel Name: ST01
- Time to HRR of 1 MW: 300 s
- HRR Rise Time: 400 s
- HRR Peak Time: 10080 s
- FRR Decay Time: 14400 s
- Fuel Load: 360
- Peak HRR Per Unit Area: 25 kW/m^2
- Critical FTP: 10500
- Q Critical: 9
- Absorptivity: 0.89

- Building Height: 8 m
- Nonburnable fraction: 0.5
- Probability of ignition from firebrands: 100 %
- Ember Ignition Hardening Factor: 0
- Time to ignition from Embers: 20 s

For temporary file storage and intermediate saves, ELMFIRE can use a scratch directory instead of the output directory. This folder can be specified using the `SCRATCH` variable. By default, this folder is used to store intermediate geospatial files. ELMFIRE converts the input geoTIFF files to binary, easier-to-process BSQ files. BSQ files come with associated XML or HDR files containing their metadata. If, for any reason, the BSQ files can be provided directly instead of the geoTIFF files (which would slightly speed up the initial data processing step), they can be used in ELMFIRE by setting `USE_EXISTING_BSQS=.TRUE.` in the `&INPUT` namelist. ELMFIRE will then search for associated XML metadata files, but this can be switched to HDR files through `USE_BSQ_XML_HEADER=.FALSE.`. If you want to clear out the scratch directory after the simulation is finished, set `CLEAN_SCRATCH=.TRUE.` in the `&SIMULATOR` namelist.

In some use cases of ELMFIRE, the fire domain may be split into multiple files. Specifically, there are cases where the ignition is within the landscape of one geotiff file, but the overall burn area is extended to eight additional, neighboring rasters. Should this be the case, USE_TILED_IO can be set to .TRUE., and ELMFIRE will combine the fire landscape from the eight input files. These files should be named, using aspect as an example, asp_1_2.tif, where 1 and 2 refer to the rasters position in space. In this case, this raster would be the top-middle part of the landscape. The files should range from asp_1_1.tif to asp_3_3.tif.

The location that contains gdal functions and programs is assumed to be usr/bin/. If that is not the case, the location can be changed through PATH_TO_GDAL.

1.8 Parallelisation and MPI

ELMFIRE has been constructed from the ground up with the primary purpose of being used with multiple cores in local or distributed computing systems. To this end, if ELMFIRE is given multiple cores to use, the reading of input files is parallellized and, should multiple cases be specified, those cases run parallel with a shared internal memory. This is particularly useful with input raster pertrubation, probabilistic averaging (in the case of firebrands and initial attack probability) and multiple weather band combinations. The use of parallelisation substantially decreases computational time with a little extra memory overhead (as each case requires its own output rasters in memory). Theoretically as all the cases are near-independent of each other, there should not be a decrease in marginal gain per core used (although any excess cores in relation to total number of cases will not be contributing to the computational time). Memory is shared between all cores in a host (i.e. physical machine / server rack), but some extra work is required if the cores are distributed between hosts. ELMFIRE offers the `MULTIPLE_HOSTS` input in the `&SIMULATOR` namelist (`.FALSE.` by default) that can be set to `.TRUE.` to deal with memory sharing across hosts.

All examples below apply to single-host instances. To run ELMFIRE in parallel you can and should use MPI through:

```
mpiexec -n 48 elmfire elmfire.data
```

The above script will run ELMFIRE with the input script elmfire.data and 48 allocated cores.

MPI has many options and input arguments you can play around with to change the behavior of the distributed cores. Though no other settings or inputs are needed to run an ELMFIRE simulation, one

useful one is the oversubscribe option that allows physical (hardware) cores to act as multiple computational nodes. This might be a worthwhile addition it for example, you have a simulation with 1000's of simple/fast simulations and only 16 cores in your system. If you try to specify more cores than are available in the system, MPI will throw an error.

```
mpiexec --oversubscribe -n 48 elmfire elmfire.data
```

For a robust, boilerplate MPI command that you can use with ELMFIRE, the following one helps avoid any dynamic reallocation of cores, distribution of cores, and add data transfer protocols. More details about these options can be found online. Note that mpirun and mpiexec are essentially identical and interchangeable in running ELMFIRE. The below command also reroutes the console output of elmfire to a file called elmfire.out.

```
mpirun --mca btl tcp,self,vader --map-by core --bind-to core --oversubscribe -np 48 elmfire elmfire
```

1.9 Physics and Numerics options

Runtime options are specified via the `&SIMULATOR` namelist group. Brief descriptions of how various parameters influence physics options are presented below.

1.9.1 Feedback level

The verbosity level of ELMFIRE can be controlled by the option FEEDBACK_LEVEL (Formerly DEBUG_LEVEL). It can be set to any integer level between 0 and 3. Feedback level 0 only returns the bare necessary outputs to show that ELMFIRE is progressing. Feedback level 1 introduces live updates of the progress of the cases, particularly when each one starts (when multiple cases are ran in parallel) and the current timestep of a case (when only one is run serially). Feedback level 2 adds some statements that describe the progression of fire potential mode (MODE = 2 or 3). Feedback level 3 adds all print statements, including status updates for all concurrent cases running in level set. Feedback level 3 is usually piped to a file instead of the terminal, that helper scripts (not provided as of yet) can use to keep track of the progression of multiple level set cases.

1.9.2 Elliptical dimensions

In ELMFIRE, every point along the fire front behaves as an independent elliptical wavelet (Huygens principle). However, since the underlying surface fire spread model only provides the rate of spread in the head fire direction, the assumed elliptical fire shape is used to calculate spread rates in other directions. A key parameter is the ellipse's length to width ratio, which is estimated as a function of wind speed from an empirical correlation. The maximum allowable value of the length to width ratio is specified with the keyword `MAX_LOW` (default value of 8). Setting this to a lower value can prevent cigar shaped fires under high winds. The midflame windspeed, calculated internally in ELMFIRE, is first calculated in ft/min, but must be converted to m/s for the LOW calculation. The conversion is done through the `WSMFEFF_LOW_MULT` multiplier in the `&SIMULATOR` namelist, equal to 0.00507955 as standard. The user can change this for calibration or optimization.

In calculating the elliptical dimensions of fire spread, the individual wind and slope components of fire spread influence need to be calculated. A factor is included in each one to enhance or reduce the influence

of wind and slope on the final rate of spread magnitude and direction, `PHIW_ADJ` and `PHIS_ADJ` respectively. Both are set to 1 as standard, in the `&SIMULATOR` namelist.

1.9.3 Crown fire

The keyword CROWN_FIRE_MODEL can be used to enable or disable crown fire initiation and spread. By default, CROWN_FIRE_MODEL=1 and crown fire spread rate is calculated from Cruz et al. 2005. Crown fire can be disabled by setting CROWN_FIRE_MODEL=0. In certain cases, crown fire spread rates may be over-predicted, so an upper limit on spread rate can be specified via CROWN_FIRE_SPREAD_RATE_LIMIT, which has a default value of 250 ft/min. Since crown fire may not always propagate in discontinuous canopies, the keyword CRITICAL_CANOPY_COVER is used to specify the minimum canopy cover at which crown fire occurs. The default value is 0.39 (note that this is a fraction, not a percent). The overall calculated crown fire rate of spread can be manually adjusted through the CROWN_FIRE_ADJ variable, as standard set to 1.0.

1.9.4 Wind fluctuations

Wind fluctuations, disabled by default, can be enabled by setting WIND_FLUCTUATIONS = .TRUE.. Doing so directs ELMFIRE to perturb the wind field (speed and direction) in every cell in the computational domain every DT_WIND_FLUCTUATIONS seconds, 15 as standard. The wind speed perturbation is the current wind speed multiplied by WIND_SPEED_FLUCTUATION_INTENSITY multiplied by a randomly generated floating point number between -0.5 and +0.5. As an example, if the current wind speed is 10 mph and WIND_SPEED_FLUCTUATION_INTENSITY is 0.2, then the perturbed (post-fluctuation) wind speed will be between 9.0 mph and 11.0 mph. Wind direction fluctuations are implemented similarly, except that WIND_DIRECTION_FLUCTUATION_INTENSITY specifies the maximum magnitude of the wind direction fluctuation in degrees. For example, if the current wind direction is 90 degrees and WIND_DIRECTION_FLUCTUATION_INTENSITY = 20.0, then the perturbed (post-fluctuation) wind speed would be between 80-100 degrees. Essentially, WIND_SPEED_FLUCTUATION_INTENSITY is a relative value whereas WIND_DIRECTION_FLUCTUATION_INTENSITY is an absolute value in degrees.

1.9.5 Miscellaneous inputs

Fire front expansion calculations are performed only in voxels/grid cells surrounding the fire front (narrow band level set method). The number of voxels on either side of the fire front is controlled by the parameter BANDTHICKNESS (default value of 2). It is normally unnecessary to adjust this parameter.

The parameter `RANDOMIZE_RANDOM_SEED` controls the seed used to initialize the random number generator. By default, `RANDOMIZE_RANDOM_SEED=.FALSE.` and the random number generator is initialized using the `SEED` value as specified in the `&MONTE_CARLO` namelist group. If `RANDOMIZE_RANDOM_SEED=.TRUE.`, then the random number seed is generated from the system clock.

A maximum wall runtime can be setup in ELMFIRE to stop simulations that take excessively long to complete through the MAX_RUNTIME option.

A DEBUG_LEVEL option is available, that, if set to an appropriately high level, provides extra information during the simulation. Right now its only function is to provide some additional information while reading BSQ rasters, if set to a value above 100.

1.9.6 Cell Untagging

ELMFIRE keeps track of and checks all the cells that are currently burning. This can create a large array of cells that need to be checked, especially in larger rasters. To decrease the computational load, a maximum of 100 cells are retained at a time. At simulation intervals of UNTAG_CELLS_TIMESTEP_INTERVAL (10 iterations as standard, not seconds), the total number of burning cells is checked, and if more than 100 are tracked, an untagging method is called. This method untags cells if any of the following are true:

- They have been tagged for more than 1 week
- They are an isolated cell with no tagged neighbors
- They are surrounded by cells within the fire area
- Their arrival time is greater than 20 hours (Activated by UNTAG_TYPE_2 = .TRUE.)
- They are classified as burned and have a ϕ value equal to -1 (Activated by UNTAG_TYPE_3 = .TRUE.)

1.9.7 End Conditions

An ELMFIRE simulation can stop for a variety of reasons. Some of the reasons are straightforward (like reaching a time limit), some are safeguards to prevent cases taking up resources for too long, and some are smart conditions aimed at automatically haulting a stalled fire front. The following list outlines all the ways that an ELMFIRE level set simulation is halted (without an error):

- Elapsed time reaches SIMULATION_TSTOP: The standard way a level set simulation ends is by reaching its allocated elapsed time.
- Simulation starts on a nonburnable cell: The ignition cell must be on a cell that can sustan fire spread, otherwise the simulation quits on startup.
- Initial attack containment successful: If suppression is turned on, and initial attack is turned on, a random number is used along with the calculated chance of successful containment upon initial attack. That randon number can then reflect the containment of the fire, haulting the simulation.
- Less than 2 nodes tagged for fire spread: If there are no firebrands currently in motion, and less than two pixels on the landscape are currently candidates to sustain fire spread, the simulation halts.
- Burned acreage reaches total acreage limit: Users can specify an upper burned area limit that gets checked each timestep. The simulation stops when this area limit is reached.
- Elapsed wall time reaches MAX_RUNTIME: To ensure that a stuck or really slow simulation does not hog computational resources, a MAX_RUNTIME argument is used that compares the elapsed wall (real) time of the simulation to the set maximum. The simulation stops if the elapsed time crosses this maximum limit.
- Stalled fire front: If all ignitions have been applied and there are no firebrands in flight, and the phi-field does not change between timesteps, the fire spread is considered stalled. This can happen if the fire is bound on all sides by the simulation domain or nonburnable cells and barriers. If this occurs, the simulation is stopped early. This feature can be used with single-band weather inputs and a really large simulation time (automatically capped at the largest int32 value internally) to emulate a Minimum Travel Time algorithm.

1.10 Barriers and Breaching

Unburnable fire breaks such as roads, rivers, railroad lines and others can stop the spread of wildfires. Sometimes these landscape features are wide enough to be represented in the fuel raster, but most of the

times they are too small compared to the fuel raster cell size and effectively disappear. This is especially important with flank- and backfire spread, where even narrow fire breaks can be enough to stop the fire spread. ELMFIRE uses the same models as Prometheus [28], using expert opinion and empirical evidence. The main breaching rule is that fire stops spreading when it encounters a barrier with width more than 1.5 times the flame length.

Barriers are read by ELMFIRE as a raster file, with each pixel of the raster having a value equal to the width of the barrier that crosses it. This file is usually generated by rasterizing an equivalent line vector files of roads, rivers, fuel breaks etc. To enable the use of barriers, set `USE_BARRIERS=.TRUE.` in the `&INPUTS` group, along with the `BARRIER_FILENAME` (again omitting the .tif ending).

Note that this setting only applies to surface fire spread. Barriers can still be breached by firebrands if their landing distance is high enough. Right now, the barrier part of ELMFIRE has no effect on the spotting calculations. Also note that enabling spotting forces BANDTHICKNESS = 1, which does not affect the rate of spread of the fire, but does tend to produce more polygonal fire isochrones.

1.11 Spotting

Spotting is a major, often dominant, mode of fire propagation particularly under hot, dry, windy conditions. Parameters that control spotting are specified on the `&SPOTTING` namelist group. Note that the current spotting model is different from the spotting model described in the original ELMFIRE paper. Make sure to include `USE_SUPERCEDED_SPOTTING=.FALSE.` to use the spotting methods in this guide. Spotting is disabled by default and can be enabled by setting `ENABLE_SPOTTING=.TRUE.` in the `&SPOTTING` namelist group. For the purposes or debugging or isolating the spotting model of ELMFIRE, `NO_SURFACE_FIRE=.TRUE.` can be set to disable surface fire spread, and (by setting an initial ϕ input representing a burning area) the spotting model can be tested without fire spread.

ELMFIRE breaks down the spotting procedure into its subsequent steps, namely generation, transport, accumulation and ignition. Each of these four parts has configurable models and options that will be summarized and expanded upon in the next sections.

1.11.1 Generation

By default, when ENABLE_SPOTTING= .TRUE., only pixels that burn as passive or active crown fire trigger the spotting algorithm. The keyword CROWN_FIRE_SPOTTING_PERCENT controls spotting initiation from passive/active crown fire pixels, meaning if it is set to 1.0 then 1 out of every 100 pixels that burn as crown fire will initiate the spotting algorithm. Surface fire spotting may be enabled by setting ENABLE_SURFACE_FIRE_SPOTTING = .TRUE.. Ember generation is treated analogously to crown fire ember generation, except that the surface fire ember generation probability is controlled by the keyword SURFACE_FIRE_SPOTTING_PERCENT(:). The index for the array SURFACE_FIRE_SPOTTING_PERCENT(:) is fuel model number so that surface fire spotting initiation can be controlled by fuel model. As an example, the following lines would disable surface fire spotting for all fuel types except for fuel models 141 - 149 (shrub fuel models in the conventional US system):

```
SURFACE_FIRE_SPOTTING_PERCENT(  1:140) = 0.0
SURFACE_FIRE_SPOTTING_PERCENT(141:149) = 1.0
SURFACE_FIRE_SPOTTING_PERCENT(150:256) = 1.0
```

A global parameter can also be set to set all the fuels to a specific value via `GLOBAL_SURFACE_FIRE_SPOTTING_PERCENT`. The keyword `CRITICAL_SPOTTING_FIRELINE_INTENSITY` (default value of 0.0) is the fireline intensity in units of kW/m below which surface fire spotting does not occur. This parameter has no impact on crown fire spotting.

Note that another set of parameters controls spotting behavior for different fuels. GLOBAL_SURFACE_FIRE_SPOTTING_ and GLOBAL_SURFACE_FIRE_SPOTTING_PERCENT_MIN define the probability of emitting firebrands from all cells as a percentage. The two parameters set the upper and lower bounds of the probability. The simulator selects a random value within these bounds during simulation to introduce perturbations.

Another parameter that may affect the simulation is EMBER_SAMPLING_FACTOR. This is the number of real firebrands represented by each individual numerical particle. This option should be used with the first version of the transport algorithm, which traces the trajectory of each particle generated from a cell. By default, EMBER_SAMPLING_FACTOR = 1, meaning the simulator traces every generated firebrand. However, because each cell may generate a significant number of real firebrands, by setting EMBER_SAMPLING_FACTOR = 10, for example, one numerical particle represents 10 real firebrands. This method is valid when the number of firebrands is large enough that fewer particles can still approximate the ground-level distribution of the firebrands. If the second version of the firebrand transport algorithm is adopted, typically EMBER_SAMPLING_FACTOR should not be specified.

The number of embers generated in every selected cell per timestep can be controlled through three available models. The first model is activated via GENERATION_MODEL = 'RANDOM'. The number of embers that is generated is determined randomly as constrained by the keywords NEMBERS_MIN and NEMBERS_MAX. More specifically, the number of embers generated will be uniformly sampled from a distribution bounded by NEMBERS_MIN and NEMBERS_MAX.

```
&SPOTTING
ENABLE_SPOTTING   = .TRUE.
USE_SUPERSEDED_SPOTTING   = .FALSE.
GENERATION_MODEL                      = 'RANDOM'
CROWN_FIRE_SPOTTING_PERCENT   = 100
ENABLE_SURFACE_FIRE_SPOTTING          = .FALSE.
NEMBERS_MIN                           = 1
NEMBERS_MAX                           = 10
...
/
```

A more physical ember generation option is available in ELMFIRE. By setting GENERATION_MODEL = 'PER-AREA', the number of generated firebrands will no longer be controlled by prescribed maximum and minimum values. Instead, spotting generation is controlled by an ember generation rate, EMBER_GR (= 0.001 embers m^{-2} s^{-1} as default) and the total time that a cell emits firebrands for after ignition TAU_EMBERGEN (= 6 s as default).

```
&SPOTTING
ENABLE_SPOTTING   = .TRUE.
USE_SUPERSEDED_SPOTTING   = .FALSE.
GENERATION_MODEL                      = 'PER-AREA'
CROWN_FIRE_SPOTTING_PERCENT   = 100
ENABLE_SURFACE_FIRE_SPOTTING          = .FALSE.
```

```
EMBER_GR                              = 0.1
TAU_EMBERGEN                          = 5
...
/
```

The following parameters define a firebrand generation scenario with a customized firebrand generation duration (1 hour=3600 seconds) and rate ($10 pcs/s/m^2$). EMBER_SAMPLING_FACTOR is set to 1, which is the default value. To illustrate, we also enable surface fire spotting, with spotting probability between 80% and 100% for all fuels except 91 (1%) and 102 (0%), after surpassing the fireline intensity threshold of 0 kw/m^2

```
&SPOTTING
ENABLE_SPOTTING   = .TRUE.
USE_SUPERSEDED_SPOTTING   = .FALSE.
GENERATION_MODEL                      = 'PER-AREA'
CROWN_FIRE_SPOTTING_PERCENT   = 100

TAU_EMBERGEN = 3600.0
EMBER_GR = 10
EMBER_SAMPLING_FACTOR = 1.0

ENABLE_SURFACE_FIRE_SPOTTING = .TRUE.
GLOBAL_SURFACE_FIRE_SPOTTING_PERCENT_MAX = 80.0
GLOBAL_SURFACE_FIRE_SPOTTING_PERCENT_MIN = 100.0
SURFACE_FIRE_SPOTTING_PERCENT_MULT(91) = 1.0
SURFACE_FIRE_SPOTTING_PERCENT_MULT(102) = 0.0

CRITICAL_SPOTTING_FIRELINE_INTENSITY(:)   = 0.
...
/
```

As an alternative, GENERATION_MODEL can be set to 'PER-MW', and then EMBER_GR will be calculated automatically, based on a customized value for ember generation per MW of the fire. The default value is 33.3 for vegetated fuels, and 10 for building fuels. These two values can be modified by the user through EMBER_GR_PER_MW_VEGE and EMBER_GR_PER_MW_BLDG.

```
&SPOTTING
ENABLE_SPOTTING   = .TRUE.
USE_SUPERSEDED_SPOTTING   = .FALSE.
GENERATION_MODEL                      = 'PER-MW'
CROWN_FIRE_SPOTTING_PERCENT   = 100
...
/
```

The total time that a burning cell will be emitting embers for is set through TAU_EMBERGEN (=6 seconds by default). The switch USE_PHYSICAL_SPOTTING_DURATION can be set to .TRUE. to calculate a more physical total emission value, especially for the building models. For vegetated models, the burn duration is dictated by the rate of spread of the fire and the dimension of computational cell.

1.11.2 Transport

Currently there is a variety of methods that can be used to simulate ember transport. The simplest one can be called with SPOTTING_DISTANCE_MODEL = 'UNIFORM', using the example below. We varied some other parameters as an example as well.

```
&SPOTTING
ENABLE_SPOTTING                      = .TRUE.
USE_SUPERCEDED_SPOTTING              = .FALSE.
CROWN_FIRE_SPOTTING_PERCENT          = 1.0
ENABLE_SURFACE_FIRE_SPOTTING         = .TRUE.
SURFACE_FIRE_SPOTTING_PERCENT(:)     = 1.0
CRITICAL_SPOTTING_FIRELINE_INTENSITY = 2000.0
GENERATION_MODEL                     = 'RANDOM'
SPOTTING_DISTANCE_MODEL              = 'UNIFORM'
MIN_SPOTTING_DISTANCE                = 60
MAX_SPOTTING_DISTANCE                = 90
NEMBERS_MIN                          = 1
NEMBERS_MAX                          = 1
...
/
```

With the Uniform option, the spotting distance will be determined as a random number between, and including MIN_SPOTTING_DISTANCE and MAX_SPOTTING_DISTANCE.

Another option for the spotting distance calculation is with SPOTTING_DISTANCE_MODEL = 'LOG-NORMAL', which treats firebrand landing as a lognormal probabilistic density function. Spotting distance is modeled as a lognormal distribution with the mean value determined semi-empirically as a function of an empirical value MEAN_SPOTTING_DISTANCE, ambient wind speed and fireline intensity. The standard deviation is calculated using the NORMALIZED_SPOTTING_DIST_VARIANCE parameter. Shown below is a sample spotting configuration:

```
&SPOTTING
ENABLE_SPOTTING                      = .TRUE.
USE_SUPERCEDED_SPOTTING              = .FALSE.
CROWN_FIRE_SPOTTING_PERCENT          = 1.0
ENABLE_SURFACE_FIRE_SPOTTING         = .TRUE.
SURFACE_FIRE_SPOTTING_PERCENT(:)     = 1.0
CRITICAL_SPOTTING_FIRELINE_INTENSITY = 2000.0
GENERATION_MODEL                     = 'RANDOM'
SPOTTING_DISTANCE_MODEL              = 'UNIFORM'
NEMBERS_MIN                          = 1
NEMBERS_MAX                          = 1
MEAN_SPOTTING_DIST                   = 5.0
NORMALIZED_SPOTTING_DIST_VARIANCE    = 250.0
...
/
```

In this case, the one firebrand generated in potential cells will travel a random distance dictated by a lognormal PDF with some mean value and a variance of 250. The actual mean value is dependent on the wind speed and fireline intensity. Refer to the mathematical section 2.3.2. To provide an example, with with of 6 m/s and fireline intensity of 300 kw/m, the resulting mean spotting distance is 87m. The correlation factors SPOT_WS_EXP and SPOT_FLIN_EXP can be changed from their standard values of 0.9 and 0.5 respectively if needed. Of course, both this and the UNIFORM option require the user to make assumptions regarding the distribution of firebrand landings.

A third option exists for estimating spotting travel distance, activated through SPOTTING_DISTANCE_MODEL = 'EMPIRICAL'. This activates the spotting model developed by UMD, which uses the pre-configured lognormal PDFs developed by Sardoy et al.[24] for vegetative fuels and by Himoto et. al. [12] for structural fuels. Both models use the 10 m wind speed, fireline intensity and Froude number to calculate PDF parameters. The user needs to input minimal new information for this spotting function. Parameters of the PDF in this mode is not editable by the users, but the users have the option to define the maximum travel distance of the firebrands by the parameter P_EPS. This value is the top 0.1% value of the PDF, i.e., the (1-P_EPS)'th percentile of the prescribed PDF. The default value of P_EPS is 0.01, i.e., the 99th percentile of the prescribed PDF.

```
&SPOTTING
ENABLE_SPOTTING   = .TRUE.
USE_SUPERSEDED_SPOTTING   = .FALSE.
GENERATION_MODEL                    = 'PER-MW'
SPOTTING_DISTANCE_MODEL             = 'EMPIRICAL'
...
/
```

To test whether the 'EMPIRICAL' mode is properly coupled with the other spotting functionalities, a fixed set of lognormal distribution parameters can be specified for diagnostic purposes. This avoids the effects of variable heat release rate (HRR) and wind speed on the ember deposition distribution. To do so, set USE_CUSTOMIZED_PDF = .TRUE. and select SPOTTING_DISTANCE_MODEL = 'EMPIRICAL'. The parameters MU_CROSSWIND and SIGMA_CROSSWIND specify the mean and standard deviation of the crosswind ember distribution, while MU_DOWNWIND and SIGMA_DOWNWIND specify the logarithm of the corresponding values for the downwind ember deposition distribution. An example namelist is shown below:

```
&SPOTTING
ENABLE_SPOTTING   = .TRUE.
USE_SUPERSEDED_SPOTTING   = .FALSE.
GENERATION_MODEL                    = 'PER-MW'
SPOTTING_DISTANCE_MODEL             = 'EMPIRICAL'
P_EPS                               = 0.01
USE_CUSTOMIZED_PDF                  = .TRUE.
MU_CROSSWIND                        = 0.0
SIGMA_CROSSWIND                     = 0.0
MU_DOWNWIND                         = 0.0
SIGMA_DOWNWIND                      = 0.0
/
```

There is an additional option to disable firebrand dispersion in the crosswind direction. In some applications, crosswind spreading is negligible, or it may be desirable to simplify diagnostic or verification cases focusing on downwind transport only. This option can be enabled by specifying CROSSWIND_DISPERSION = .FALSE..

1.11.3 Accumulation

The user can also choose the way in which the landing of firebrands is calculated. The two options are Lagrangian (treating each generated firebrand as one explicitly solved particle) and Eulerian (treating the firebrands as a flux, that is deposited in separate cells). The user can specify which one to use through ACCUMULATION_MODEL = 'LAGRANGIAN' or 'EULERIAN'.

The Lagrangian model solves the flight path of each individual generated firebrand. The path follows the streakline of a particle originated from the firebrand-emission source and has a probabilistic length determined by the PDF. It is then allowed to land to a cell and has a subsequent chance to start a new spot fire.

An alternative, and more numerically robust method (that later relies less on user-determined ignition probability) uses the Eulerian model, which does not track individual firebrands for ignition, but rather keeps track of the ember flux into each area.

As the choice of accumulation model is a critical part of the entire spotting submodule, there are parts of the spotting model that only work with the Lagrangian or the Eulerian model. There are ongoing efforts to make all the spotting submodels interoperable. Until then there are some limitations of the combinations of models that can be used. Right now, the most notable limitation is that SPOTTING_DISTANCE_MODEL = 'UNIFORM' only works with ACCUMULATION_MODEL = 'LAGRANGIAN'. Other than this, all other combinations are valid.

```
&SPOTTING
ENABLE_SPOTTING   = .TRUE.
USE_SUPERSEDED_SPOTTING   = .FALSE.
GENERATION_MODEL                    = 'PER-MW'
SPOTTING_DISTANCE_MODEL             = 'EMPIRICAL'
ACCUMULATION_MODEL                  = 'EULERIAN'
...
/
```

1.11.4 Ignition

ELMFIRE provides three methods of calculating if and when a landed firebrand (or total firebrand flux) will result in ignition.

The simplest model can be set through IGNITION_MODEL = 'DIRECT'. Once the particle has landed, the ignition probability (probability of setting ϕ to -1 upon landing) is controlled by the keyword PIGN (%), with no delay. To avoid allocating computational time to tracking embers that don't initiate spot fire, it is most computationally efficient to set PIGN to 100% and then set CROWN_FIRE_SPOTTING_PERCENT to a small number. As an example, CROWN_FIRE_SPOTTING_PERCENT = 50.0 and PIGN = 10.0 will produce comparable results to CROWN_FIRE_SPOTTING_PERCENT = 5.0 and PIGN = 100.0, but the computational expense of the spotting algorithm is reduced by a factor of approximately 10x in the latter case. An

additional multiplier can be added to this calculation, SURFACE_FIRE_SPOTTING_PERCENT_MULT(:), for optimization or calibration.

An entry like the one below is a complete, valid spotting input:

```
&SPOTTING
ENABLE_SPOTTING   = .TRUE.
USE_SUPERSEDED_SPOTTING   = .FALSE.
GENERATION_MODEL                    = 'PER-MW'
SPOTTING_DISTANCE_MODEL             = 'EMPIRICAL'
ACCUMULATION_MODEL                  = 'EULERIAN'
IGNITION_MODEL                      = 'DIRECT'
PIGN                                = 100
/
```

A more advanced model can be activated through IGNITION_MODEL = 'SIMPLE'. The simple model relies on a simple relation that requires the PIGN input, the local ignition time, LOCAL_IGNITION_TIME, and the firebrand ignition delay CELL_IGNITION_DELAY. To physically understand these parameters, once a firebrand flux arrives at a specific cell, PIGN is defined by the probability of a successful transition from smoldering to flaming within the time interval defined by LOCAL_IGNITION_TIME. Once the smoldering-to-flaming igntion is achieved, additional CELL_IGNITION_DELAY seconds will be required to effectively ignite the target cell (set ϕ to -1), which tries to represent the development of fire to a self-sustaining scale. Both LOCAL_IGNITION_TIME and CELL_IGNITION_DELAY have preset values in ELMFIRE of 30 s and 100 s respectively.

```
&SPOTTING
ENABLE_SPOTTING   = .TRUE.
USE_SUPERSEDED_SPOTTING   = .FALSE.
GENERATION_MODEL                    = 'PER-MW'
SPOTTING_DISTANCE_MODEL             = 'EMPIRICAL'
ACCUMULATION_MODEL                  = 'EULERIAN'
IGNITION_MODEL                      = 'SIMPLE'
PIGN                                = 100
/
```

A more complex ignition model, IGNITION_MODEL = 'PHYSICAL', is based on physical observations of the ignition chances of firebrands. Smoldering-to-framing transition probability and time are automatically calculated and parameterized as a function of local wind speed and accumulated firebrand flux. This model reduces the need for PIGN to be specified. Note that this mode has been developed for building fuels; the vegetative fuels can be differentiated by setting DIFF_WILDLAND_IGNITION = .TRUE. in which case the vegetative fuels will definitely ignite after a delay defined by (LOCAL_IGNITION_TIME + CELL_IGNITION_DELAY) when the accumulated firebrand flux is larger than 0.

An example is given below. Note how using the options below does not require any numerical input from the user. This is the great advantage of the physical and empirical firebrand models. The user can edit any of the standard values if they so wish. Note that, although these setting have the great benefit of not requiring any user input, they greatly increase the required computational time.

```
&SPOTTING
```

```
ENABLE_SPOTTING   = .TRUE.
USE_SUPERSEDED_SPOTTING   = .FALSE.
GENERATION_MODEL                    = 'PER-MW'
SPOTTING_DISTANCE_MODEL             = 'EMPIRICAL'
ACCUMULATION_MODEL                  = 'EULERIAN'
IGNITION_MODEL                      = 'PHYSICAL'
/
```

1.11.5 Consumption (under development)

The current version of ELMFIRE implements a simplified ember-consumption model to account for firebrand burnout after deposition. In the default configuration, firebrands are assumed to remain burning for the entire simulation duration, which may lead to unrealistic ignition behavior in some cases.

To address this limitation, a consumption model (currently under development and testing) has been introduced. This feature can be enabled by setting USE_EMBER_CONSUMPTION = .TRUE.. However, it should be noted that the present implementation may overestimate ember consumption rates, which can suppress ignition and lead to reduced or even zero effective ignition in some simulations.

1.11.6 Submodel Cheat Sheet

To keep track of all the available models, and the input parameters that control them, we have produced a cheat sheet that should make selection of parameters easier. Refer to Table 1.3 for this.

GENERATION_MODEL		
RANDOM*	**PER-AREA**	**PER-MW**
NUMBERS_MIN	EMBER_GR	EMBER_GR_PER_MW_BLDG
NUMBERS_MAX		EMBER_GR_PER_MW_VEGE
Generation time control		
USE_PHYSICAL_SPOTTING_DURATION		
TAU_EMBERGEN		
SPOTTING_DISTANCE_MODEL		
L: Available when ACCUMULATION_MODEL=LAGRANGIAN		
E: Available when ACCUMULATION_MODEL=EULERIAN		
UNIFORM (L)	**LOGNORMAL(L,E)**	**EMPIRICAL**(L,E)**
MAX_SPOTTING_DISTANCE	MEAN_SPOTTING_DIST	P_EPS
MIN_SPOTTING_DISTANCE	NORMALIZED_SPOTTING_DIST_VARIANCE	USE_CUSTOMIZED_PDF
	SPOT_FLIN_EXP	MU_CROSSWIND
	SPOT_WS_EXP	SIGMA_CROSSWIND
		MU_DOWNWIND
		SIGMA_DOWNWIND
ACCUMULATION_MODEL		
LAGRANGIAN	**EULERIAN**	
EMBER_SAMPLING_FACTOR		
IGNITION_MODEL		
DIRECT	**SIMPLE**	**PHYSICAL**
PIGN	PIGN	LOCAL_IGNITION_TIME
	LOCAL_IGNITION_TIME	CELL_IGNITION_DELAY
	CELL_IGNITION_DELAY	DIFF_WILDLAND_IGNITION

Table 1.3: Configuration options for ember generation, spotting distance, accumulation, and ignition models.

1.11.7 Stochastic Parameters

Particularly under high winds, the parameters that control spotting exert a significant influence on overall fire progression. For that reason, it is often desirable to treat these parameters stochastically and, for fire hindcasts, as calibration coefficients. Setting the parameter STOCHASTIC_SPOTTING = .TRUE. directs ELMFIRE to treat the parameters that control spotting stochastically. When STOCHASTIC_SPOTTING = .TRUE., the following eight parameters are randomly generated within a user-specified range as will be described later:

- CROWN_FIRE_SPOTTING_PERCENT
- GLOBAL_SURFACE_FIRE_SPOTTING_PERCENT
- MEAN_SPOTTING_DIST
- NEMBERS_MAX
- NEMBERS_MIN
- NORMALIZED_SPOTTING_DIST_VARIANCE
- PIGN
- SPOT_FLIN_EXP
- SPOT_WS_EXP

Note that `GLOBAL_SURFACE_FIRE_SPOTTING_PERCENT` is applied to all fuel models. Shown below is a `&SPOTTING` configuration with stochastic spotting enabled and the 16 parameters that constrain the allowable range of the eight spotting parameters described above.

```
&SPOTTING
ENABLE_SPOTTING                          = .TRUE.
STOCHASTIC_SPOTTING                      = .TRUE.
IGNITION_MODEL                           = 'RANDOM'
SPOTTING_DISTANCE_MODEL                  = 'LOGNORMAL'
ACCUMULATION_MODEL                       = 'LAGRANGIAN'
IGNITION_MODEL                           = 'DIRECT'
CROWN_FIRE_SPOTTING_PERCENT_MIN          = 0.2
CROWN_FIRE_SPOTTING_PERCENT_MAX          = 0.8
ENABLE_SURFACE_FIRE_SPOTTING             = .TRUE.
GLOBAL_SURFACE_FIRE_SPOTTING_PERCENT_MIN = 0.2
GLOBAL_SURFACE_FIRE_SPOTTING_PERCENT_MAX = 0.8
CRITICAL_SPOTTING_FIRELINE_INTENSITY     = 2000.0
SPOTTING_DISTRIBUTION_TYPE               = 'LOGNORMAL'
MEAN_SPOTTING_DIST_MIN                   = 5.0
MEAN_SPOTTING_DIST_MAX                   = 10.0
NORMALIZED_SPOTTING_DIST_VARIANCE_MIN    = 250.0
NORMALIZED_SPOTTING_DIST_VARIANCE_MAX    = 600.0
SPOT_WS_EXP_LO                           = 0.4
SPOT_WS_EXP_HI                           = 0.7
SPOT_FLIN_EXP_LO                         = 0.2
SPOT_FLIN_EXP_HI                         = 0.4
NEMBERS_MIN                              = 1
NEMBERS_MAX_LO                           = 1
NEMBERS_MAX_HI                           = 1
```

```
PIGN_MIN                                = 100.0
PIGN_MAX                                = 100.0
/
```

1.11.8 Spotting Outputs

ELMFIRE provides several output options for diagnosing spotting, ember deposition, and spot ignitions. When DUMP_SPOTTING_OUTPUTS = .TRUE., ELMFIRE writes a CSV file named `spotting_stats_<case>.csv` for each simulation case. Each record includes the source and destination grid cells, launch time, ignition or arrival time, and spotting distance for an explicitly tracked firebrand. This output is intended primarily for debugging the spotting trajectory model, validating spotting-distance distributions, and examining individual long-range spotting events. It is implemented for the newer Lagrangian spotting model, in which individual firebrands are explicitly tracked, and is not a useful diagnostic for the Eulerian model, which transports a spatial distribution rather than individual particles. Because the CSV files can become very large, this option should normally be enabled only for small verification cases or short simulations. When EMBER_SAMPLING_FACTOR is greater than one, the records may represent sampled firebrands rather than the full physical population.

When DUMP_EMBER_FLUX_TRANSIENT = .TRUE., ELMFIRE writes an `ember_flux_transient` raster at each regular output time. This raster contains the number, or expected number, of firebrands deposited in each cell since the preceding output. After the raster is written, the transient array is reset to zero; therefore, the output interval is controlled by DTDUMP. This output is useful for examining when and where firebrands arrive during a simulation, rather than only their total deposition over the complete case. Eulerian values may be fractional because they represent probability-weighted expected numbers of deposited firebrands, and small values should not be interpreted as individually resolved particles. Frequent output intervals can generate many large raster files and increase both storage and I/O costs.

When DUMP_EMBER_FLUX = .TRUE., ELMFIRE writes the cumulative number, or expected number, of firebrands deposited in each grid cell during a simulation case. This output is cumulative over simulation time, unlike DUMP_EMBER_FLUX_TRANSIENT. Lagrangian calculations generally add discrete or sampling-weighted firebrand counts, whereas Eulerian calculations add probability-weighted expected counts. This output is appropriate for identifying locations exposed to concentrated firebrand deposition and for comparing spotting exposure among cases. If ACCUMULATE_EMBER_FLUX = .TRUE., a separate accumulated raster can be used to sum deposition across Monte Carlo cases and MPI ranks. The per-case working raster must still be reset at the start of each case so that ember deposition from one ensemble member does not affect the ignition calculation of another ensemble member.

When DUMP_EMBER_IGNITION = .TRUE., ELMFIRE writes an `ember_ignition_<case>_<time>` raster at the final output time. Cells assigned a value of one indicate locations ignited by the spotting algorithm. This output distinguishes successful spot ignitions from cells that only received deposited firebrands. A cell with nonzero ember flux does not necessarily appear in the ember-ignition raster, because ignition also depends on the selected IGNITION_MODEL, fuel burnability, ignition probability or physical criterion, and any ignition-development delay. The ember-ignition raster should be initialized or reset for every simulation case to prevent ignition markers from an earlier Monte Carlo case from appearing in a later per-case output.

ELMFIRE can also produce Monte Carlo summaries of deposited ember counts using ember-count bins. The valid namelist parameter is NUM_EMBER_COUNT_BINS. Setting USE_EMBER_COUNT_BINS = .TRUE. enables the summary, and NUM_EMBER_COUNT_BINS specifies the number of bands in the

resulting `ember_bin_counts` raster. The lower and upper limits of each band are specified using EMBER_COUNT_BIN_LO(i) and EMBER_COUNT_BIN_HI(i). For bin i, a case is counted when its deposited firebrand count satisfies

$$\text{EMBER\_COUNT\_BIN\_LO}(i) < N_{\text{embers}} \leq \text{EMBER\_COUNT\_BIN\_HI}(i).$$

Each output band therefore reports how many Monte Carlo cases placed that cell's ember count within the specified interval. For example:

```
USE_EMBER_COUNT_BINS   = .TRUE.
NUM_EMBER_COUNT_BINS   = 3
EMBER_COUNT_BIN_LO(1)  = 0
EMBER_COUNT_BIN_HI(1)  = 10
EMBER_COUNT_BIN_LO(2)  = 10
EMBER_COUNT_BIN_HI(2)  = 100
EMBER_COUNT_BIN_LO(3)  = 100
EMBER_COUNT_BIN_HI(3)  = 32767
```

The bin count must be positive and cannot exceed the source-code capacity of 100. Bin intervals should be ordered, non-overlapping, and wide enough to cover the expected counts. The current counting implementation is based on explicitly landed particles and is therefore most appropriate for Lagrangian or legacy spotting. It should not be assumed to provide equivalent statistics for fractional Eulerian deposition. Enabling this option also increases memory use and MPI communication, particularly when many firebrands are tracked.

1.12 Monte Carlo Analysis

Keywords controlling Monte Carlo parameters are specified in the `&MONTE_CARLO` namelist group. Several types of Monte Carlo simulations can be performed with ELMFIRE. Point source ignitions may be placed randomly across the landscape, wind and weather inputs may be varied stochastically, and input rasters may be spatially and temporally perturbed. The randomisation seed can be set manually through the `SEED` input, if the user wants to control for randomness.

1.12.1 Randomized ignition locations

Setting RANDOM_IGNITIONS = .TRUE. directs ELMFIRE to conduct a Monte Carlo analysis with ignition locations distributed randomly within the computational domain. Allowable locations of those ignitions are constrained by the parameter EDGEBUFFER which is the distance from the edge of the GIS input data tile where ignitions will not be placed as shown graphically below:

Random ignition location may also be constrained by an ignition mask raster. To do this, set `USE_IGNITION_MASK=.TRUE.` and specify `IGNITION_MASK_FILENAME` on the `&INPUTS` line. This directs ELMFIRE to read a Float32 raster named `IGNITION_MASK_FILENAME.tif` from the `FUELS_AND_TOPOGRAPHY_DIRECTORY`. You can also build an ignition mask within ELMFIRE by automatically excluding all nonburnable cells, by setting `ADD_TO_IGNITION_MASK=.TRUE.`. If the user wants to include nonburnable pixels to be potential ignition spots, then set `ALLOW_NONBURNABLE_PIXEL_IGNITION=.TRUE.` in the `&SIMULATOR` namelist.

Distribution of ignitions across the landscape is controlled by the parameter RANDOM_IGNITIONS_TYPE:

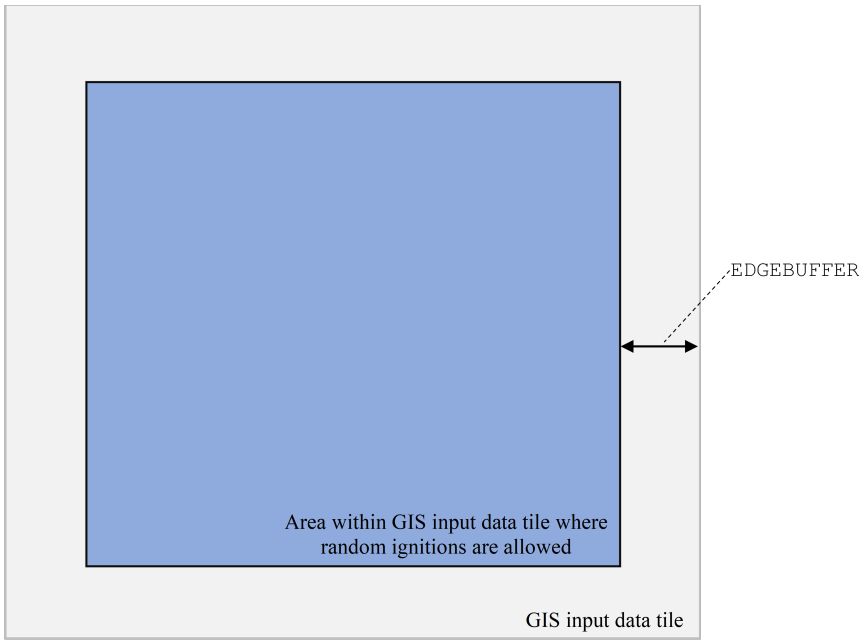


Figure 1.1: A graphical display of EDGEBUFFER.

- RANDOM_IGNITIONS_TYPE=1 (default): Ignitions are spatially distributed uniformly across the landscape in any pixels where the ignition mask raster is greater than zero. This is useful for modeling ignitions originating from powerlines or roads, for example.
- RANDOM_IGNITIONS_TYPE=2: Ignitions are spatially distributed across the landscape at a density that is proportional to the ignition mask raster. This is useful for burn probability modeling where some locations across the landscape have higher ignition probabilities than other locations. A global IGNITION_MASK_SCALE_FACTOR can be set to upscale or downscale the mask for calibration or optimization.

The number of randomly-generated ignition points can be specified in two ways:

By directly specifying NUM_ENSEMBLE_MEMBERS, or

By specifying PERCENT_OF_PIXELS_TO_IGNITE and setting NUM_ENSEMBLE_MEMBERS to a value less than 0.

In the first case, ELMFIRE will randomly generate NUM_ENSEMBLE_MEMBERS ignition locations in the allowable ignition area as defined by EDGEBUFFER (and, if USE_IGNITION_MASK = .TRUE., the user-specified ignition mask raster as well). In the second case, ELMFIRE will count all pixels within the allowable ignition area and randomly ignite the user-specified percent of those pixels. In some cases, a pixel may be selected more than once to be ignited. This can be disabled by setting ALLOW_MULTIPLE_IGNITIONS_AT_A_PIXEL = .FALSE..

Ignitions can also be set manually though a csv file with coordinates of ignitions. Setting `CSV_FIXED_IGNITION_LOCATIONS` = .TRUE. will instruct ELMFIRE to look for a file called `IGNITIONS_CSV_FILENAME.csv` within the `FUELS_AND_TOPOGRAPHY_DIRECTORY` folder.

The wind direction can either be changed randomly using the standard monte carlo parameters. In some cases it may be desirable for the wind to always drive the fire from its ignition point towards the center of the domain. For this case, `POINT_WIND_TO_CENTER` can be set to .TRUE. and the wind direction of every

case will point from the ignition point to the center of the domain.

1.12.2 Randomized weather streams

In the example discussed in Randomized ignition locations, ignition locations are selected randomly and a single weather stream (hindcast or forecast) is provided as input to those simulations. Often, as part of a climatological fire risk analysis, it is desirable to simulate wind and weather conditions under a range of historical (or forecasted) wind and weather conditions. This can be accomplished in ELMFIRE using stacked/multiband weather/meteorology rasters.

The manner in which these rasters are used is controlled by the parameters NUM_METEOROLOGY_TIMES, METEOROLOGY_BAND_START, METEOROLOGY_BAND_STOP, and METEOROLOGY_BAND_SKIP_INTERVAL. This is perhaps best illustrated by an example. Assume that numerical weather prediction has been used to generate 100 24-hour "blocks" of historical wind/weather data at hourly intervals that will be provided as input to a Monte Carlo analysis. The goal is to simulate 24-hours of fire spread, for each of these 24-hour blocks. However, these 24-hour blocks are not temporally contiguous, meaning the first 24-hour block could be from August 1983 and the next 24-hour block could be from October 2012, and so on.

Each 24-hour block consists of 25 bands since Band 1 is $t = 0$ and Band 25 is $t = 24$ hours. Therefore, each meteorology raster would contain 2,500 separate bands (100 blocks $\times$ 25 bands per block). Since ELMFIRE needs 25 bands of wind/weather data to drive each 24-hour simulation, we would set METEOROLOGY_BAND_START=1, METEOROLOGY_BAND_STOP=2476, and METEOROLOGY_BAND_SKIP_INTERVAL=25. This tells ELMFIRE to use every 25th band as the starting band for a 24-hour simulation and ensures that each 24-hour simulation starts with Band 1, 26, 51, and so on until the final 24-hour block is reached at meteorology band 2476.

If, instead, the 100 24-hour blocks were temporally contiguous (for example, a continuous hindcast from mid-July through the end of October) and the purpose of the Monte Carlo analysis is to ignite fires every 3 hours and model their spread, we would set METEOROLOGY_BAND_SKIP_INTERVAL=4.

The output of each weather band range will be saved as a separate raster file, with the band number appended at the end. This number is restricted to 3 digits, but 4 can be used by adding USE_FOUR_DIGITS_IN_IWX_BAND = .TRUE..

1.12.3 Spatial and temporal perturbations of input rasters

All inputs are subject to inherent uncertainty. To address this uncertainty, input rasters may be perturbed stochastically from their baseline values in a Monte Carlo analysis. Currently, the following rasters may be perturbed:

- ADJ: Spread rate adjustment factor (-)
- CBD: Canopy bulk density (kg/m^3)
- CBH: Canopy base height (m)
- CC: Canopy cover (-)
- CH: Canopy height (m)
- FMC: Foliar moisture content (-)
- M1: 1-hour fuel moisture (-)
- M10: 10-hour fuel moisture (-)
- M100: 100-hour fuel moisture (-)
- MLH: Live herbaceous fuel moisture (-)

- MLW: Live woody fuel moisture (-)
- WAF: Wind adjustment factor (-)
- WD: Wind direction (deg)
- WS: Wind speed (mph)

Parameters controlling stochastic perturbations of input rasters are specified in the `&MONTE_CARLO` namelist group. A random sampling procedure is implemented such that the spatial perturbation to be applied to a given input raster sampled from a probability density function (pdf). The keyword `NUM_RASTERS_TO_PERTURB` prescribes the number of input rasters to be perturbed and the keyword `RASTER_TO_PERTURB(:)` is a string corresponding to one of the raster names in the bulleted list above (ADJ, CBD, etc.). As an example, consider the following lines:

```
NUM_RASTERS_TO_PERTURB   = 1
RASTER_TO_PERTUB(1)      = 'ADJ'
SPATIAL_PERTURBATION(1)  = 'GLOBAL'
TEMPORAL_PERTURBATION(1) = 'STATIC'
PDF_TYPE(1)              = 'UNIFORM'
PDF_LOWER_LIMIT(1)       = -0.10
PDF_UPPER_LIMIT(1)       = 0.10
```

These lines specify that a randomly selected value between -0.1 and 0.1 should be added to the spread rate adjustment factor(ADJ). This perturbation will be applied globally to all pixels (SPATIAL_PERTURBATION = 'GLOBAL') and is temporally invariant for the duration of the simulation (TEMPORAL_PERTURBATION = 'STATIC'). Rather than applying such perturbations globally to all pixels, different perturbations can be applied to different pixels by setting SPATIAL_PERTURBATION = 'PIXEL'. Some raster inputs, such as wind speed and direction, are multi-band rasters that vary temporally. Different perturbations may be applied at each time by setting TEMPORAL_PERTURBATION = 'DYNAMIC'. Normally, this would only be done for wind speed and direction.

ELMFIRE now supports three kinds of statistical raster pertrubations, 'UNIFORM', 'GAUSSIAN' and 'LOGNORMAL' specified by the PDF_TYPE input. The 'UNIFORM' option selects a random pertrubation amount ranging between PDF_LOWER_LIMIT and PDF_UPPER_LIMIT. The 'GAUSSIAN' option randomly selects a pertrubation value based on a normal (or gaussian) distribution with mean PDF_MEAN and standard deviation PDF_SIGMA. The 'LOGNORMAL' option is a bit more complicated, as to allow for positive and negative pertrubations, a mean-shifted lognormal implementation is applied instead. The standard deviation of the lognormal curve is controlled by PDF_SIGMA and the mean shift amount is controlled by PDF_MEAN.

To visualise each of these options, the example case monte_carlo has been set up with three variables being perturbed with the three different methods:

```
&MONTE_CARLO
NUM_METEOROLOGY_TIMES  = 72
RANDOM_IGNITIONS  = .TRUE.
USE_IGNITION_MASK  = .TRUE.
EDGEBUFFER                  = 5.
NUM_ENSEMBLE_MEMBERS  = 5000
!PERCENT_OF_PIXELS_TO_IGNITE = 10
NUM_RASTERS_TO_PERTURB      = 3
```

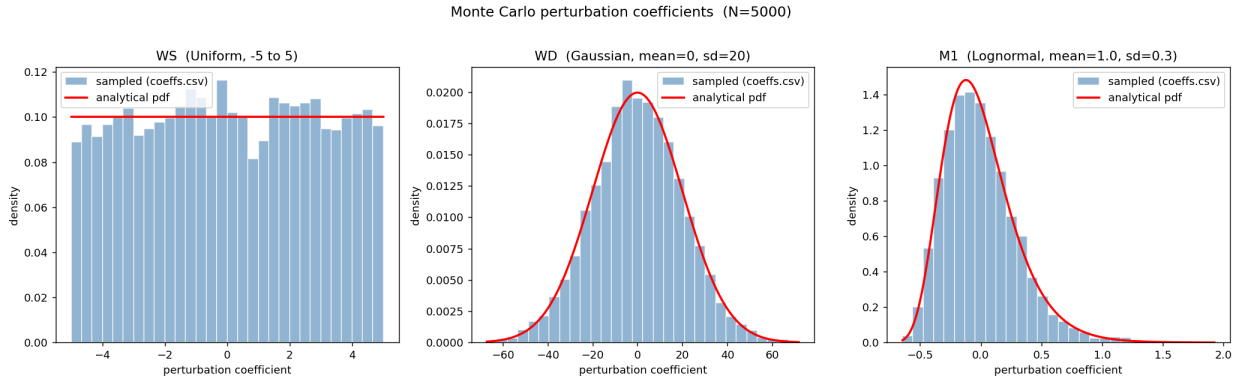


Figure 1.2: The perturbation coefficient histograms, each using a different pertrubation method.

```
RASTER_TO_PERTURB(1)        = 'WS'
SPATIAL_PERTURBATION(1)     = 'GLOBAL'
TEMPORAL_PERTURBATION(1)    = 'STATIC'
PDF_TYPE(1)                 = 'UNIFORM'
PDF_LOWER_LIMIT(1)          = -5.0
PDF_UPPER_LIMIT(1)          = 5.0
RASTER_TO_PERTURB(2)        = 'WD'
SPATIAL_PERTURBATION(2)     = 'GLOBAL'
TEMPORAL_PERTURBATION(2)    = 'STATIC'
PDF_TYPE(2)                 = 'GAUSSIAN'
PDF_MEAN(2)          = 0.0
PDF_SIGMA(2)          = 20.0
RASTER_TO_PERTURB(3)        = 'M1'
SPATIAL_PERTURBATION(3)     = 'GLOBAL'
TEMPORAL_PERTURBATION(3)    = 'STATIC'
PDF_TYPE(3)                 = 'LOGNORMAL'
PDF_MEAN(3)          = 1.0
PDF_SIGMA(3)          = 0.3
/
```

This case is run 5000 times and the pertrubation amounts are saved in the coeffs.csv file in outputs/. We have included a python script to help visualise the results and compare them with the requested distributions. Below are the results.

The number of ensemble members in the Monte Carlo analysis should be specified using the keyword `NUM_ENSEMBLE_MEMBERS` (`&MONTE_CARLO` namelist group). However, when using randomly-placed ignitions it may instead be preferable to specify the total number of ensemble members by specifying the percentage of pixels within the computational domain and possibly within a user-specified ignition mask to ignite. This is described in Randomized ignition locations.

1.12.4 Randomized wind fluctuation intensities

Wind fluctuations are implemented by changing `PERTURB_WIND_DIRECTION_FLUCTUATION_INTENSITY=.TRUE.` and `PERTURB_WIND_SPEED_FLUCTUATION_INTENSITY=.TRUE.` respectively. The keywords that control these fluctuations are `WIND_SPEED_FLUCTUATION_INTENSITY` and `WIND_DIRECTION_FLUCTUATION_INTENSITY`. These values can be randomized by setting the following four parameters (in the `&MONTE_CARLO` namelist group):

- WIND_DIRECTION_FLUCTUATION_INTENSITY_MIN: Minimum wind direction fluctuation intensity value
- WIND_DIRECTION_FLUCTUATION_INTENSITY_MAX: Maximum wind direction fluctuation intensity value
- WIND_SPEED_FLUCTUATION_INTENSITY_MIN: Minimum wind speed fluctuation intensity value
- WIND_SPEED_FLUCTUATION_INTENSITY_MAX: Maximum wind speed fluctuation intensity value

The wind fluctuation intensities are then randomly generated according to a uniform probability density function, with the upper and lower values specified by the above four parameters.

1.12.5 Outputs

Several outputs are specific to Monte Carlo analyses. Often, when conducting a Monte Carlo analysis with randomly distributed ignitions, it is informative to view spatial burn probabilities, defined as the fraction of simulations in which a given pixel burned. To enable this calculation (which adds some computational and network overhead), set CALCULATE_TIMES_BURNED = .TRUE. in the `&OUTPUTS` namelist group.

This instructs ELMFIRE to aggregate all simulated fire perimeters and calculate the burn probability across all runs and write the results to disk (in a file called burn_probability.tif). The output raster has three bands. The first is the conventional burn probability. The second is passive crown fire burn probability, meaning the fraction of times in which a pixel burned as passive crown fire. The third band is active crown fire burn probability.

During a Monte Carlo simulation, it is often useful to compute the total deposited firebrand count across all ensemble cases and MPI ranks and write the result to a single final ember-flux raster. This can be enabled by setting ACCUMULATE_EMBER_FLUX to .TRUE.

1.13 Urban Fire Spread

ELMFIRE includes two experimental building fire-spread models for WUI applications, based on the work of Hamada [11] and Purnomo et al. [18]. Most Rothermel-based models treat built-up areas as nonburnable, since urban conflagrations were not a primary concern when the model was developed. Consequently, the associated spread mechanisms and building fuel models were not included in the original framework.

The experimental models described here extend ELMFIRE to estimate surface fire spread through built-up areas in a manner analogous to the Rothermel model. The Hamada model assumes elliptical fire spread within an uniform urban community. The WU-E model provides semi-physical predictions based on the heat flux from discretized sources representing burning structures. Additional details are provided in the mathematical formulation section.

`USE_BLDG_SPREAD_MODEL=.TRUE.` must be set to enable either building fire-spread model. The input

parameters for these models include the characteristic building footprint dimension, building separation distance, nonburnable fraction of the structure, fraction of the computational cell occupied by the building footprint, and building type. These parameters can be specified as constants over the entire simulation domain by setting `USE_CONSTANT_BLDG_SPREAD_MODEL_PARAMS=.TRUE.`, which is the default option. The `BUILDING_FUEL_MODEL_FILE` must also be specified. The `BLDG_SPREAD_MODEL_TYPE` option is used to select the building spread model: Hamada model (1) or UCB–UMD WU-E model (2).

The Hamada and WU-E models account only for fire spread within built-up areas. Fire spread from wildland fuels into built-up areas is treated separately using one of two interface-spread options: the original formulation used by Purnomo et al. [18] (1), or a simplified formulation based on distance and fireline-intensity thresholds (2). This option is selected in the `&WUI` namelist using `INTERFACE_MODEL_TYPE`.

The corresponding universal input parameters can be set in the `&WUI` namelist as follows:

- `BLDG_AREA_CONSTANT`: characteristic building footprint dimension (m)
- `BLDG_SEPARATION_DIST_CONSTANT`: building separation distance (m)
- `BLDG_NONBURNABLE_FRAC_CONSTANT`: nonburnable fraction of the structure
- `BLDG_FOOTPRINT_FRAC_CONSTANT`: fraction of the computational cell occupied by the building footprint
- `BLDG_FUEL_MODEL_CONSTANT`: building fuel model number

The equivalent raster inputs, specified in `&INPUTS` are:

- `BLDG_AREA_FILENAME`
- `BLDG_SEPARATION_DIST_FILENAME`
- `BLDG_NONBURNABLE_FRAC_FILENAME`
- `BLDG_FOOTPRINT_FRAC_FILENAME`
- `BLDG_FUEL_MODEL_FILENAME`

As such, a valid part of the input file would be:

```
&WUI
BLDG_AREA_CONSTANT                    = 20.0
BLDG_SEPARATION_DIST_CONSTANT         = 10.0
BLDG_NONBURNABLE_FRAC_CONSTANT        = 0.9
BLDG_FOOTPRINT_FRAC_CONSTANT          = 0.2
BLDG_FUEL_MODEL_CONSTANT              = 2
BLDG_SPREAD_MODEL_TYPE                = 2
INTERFACE_MODEL_TYPE                  = 1
USE_BLDG_SPREAD_MODEL                 = .TRUE.
USE_CONSTANT_BLDG_SPREAD_MODEL_PARAMS = .TRUE.
/
```

The keyword USE_BLDG_SPREAD_MODEL determines whether or not the urban fire spread is activated. The keyword BLDG_SPREAD_MODEL_TYPE corresponds to the urban fire spread model selected, 1 for Hamada model and 2 for WU-E model. The keyword USE_CONSTANT_BLDG_SPREAD_MODEL_PARAMS determines if the structure attributes are assumed to be uniform. If this parameter is set .TRUE., then the structures in the domain to be simulated have uniform attributes determined by the 5 first parameters in the &WUI namelist. All the structures in the domain will have characteristic dimension specified by BLDG_AREA_CONSTANT, separation distance by BLDG_SEPARATION_DIST_CONSTANT, burnable fraction by BLDG_NONBURNABLE_FRAC_CONSTANT, and building footprint by BLDG_FOOTPRINT_FRAC_CONSTAN When BLDG_SPREAD_MODEL_TYPE is set to 2 (i.e., WU-E model), the structures also have attributes fol-

lowing the structure fuel model in building_fuel_models.csv. The value specified in BLDG_FUEL_MODEL_CONSTANT corresponds to the row number in the building_fuel_models.csv. The structure fuel model categorizes structures based on their burning behavior consisting of heat release rate (HRR) curve and heat transfer coefficients. Columns 3 to 6 correspond to the time constant in the HRR curve, column 8 corresponds to the peak HRR, column 9 corresponds to critical flux time product (FTP), column 10 corresponds to radiation absorptivity, column 11 corresponds to structure height, and column 12 corresponds to noncombustible fraction by surface area of structures. A complete example, with units and indicative values, can be found in section 1.7.

If USE_CONSTANT_BLDG_SPREAD_MODEL_PARAMS is set .FALSE., the structures attributes in the domain will become nonuniform and are specified based on input rasters, which are declared in the &INPUTS namelist as the following:

- BLDG_AREA_FILENAME: Structure area, m^2 (32-bit floating)
- BLDG_FOOTPRINT_FRAC_FILENAME: Structure footprint fraction (32-bit floating)
- BLDG_FUEL_MODEL_FILENAME: Structure fuel model in a cell (16-bit integer)
- BLDG_NONBURNABLE_FRAC_FILENAME: Nonburnable fraction of structure in a cell (32-bit floating)
- BLDG_SEPARATION_DIST_FILENAM: Separation distance, m (32-bit floating)

One important thing to highlight is the effect of roadways to WU-E. In the original LANDFIRE data, roadways are classified as fuel type 91, grouping them with urban cells. To differentiate between roadways and structures within these urban cells, roadway data such as from the Natural Earth public repository can be overlaid on the fuel type data. Initially, all urban cells, including structures and paved surfaces, were classified as fuel type 91, which is "burnable" when the urban fire spread model is enabled. However, by using the roadway data, cells containing paved areas can be identified and separated from those containing buildings. Urban cells without major roads were reclassified as "burnable", indicating the presence of structures that are susceptible to fire.

Cloudfire provides a tool for creating the required raster inputs for a urban fire spread simulation that takes into account the separation of roads and structures. It uses global data sources for most of the required parameters, and US-based data sources for a complete dataset. The tool makes conducting urban fire spread simulation much easier and feasible without extensive knowledge of the target area's layout. The tool can be found at `https://github.com/ma-th/firedx`.

In the most recent version, a new parameter is introduced in &WUI namelist: HRR_ELLIPSE_ADJ. This parameter control the effective area of the flame ellipse affecting heat transfer to subsequent structures. A default value of 0.5 is used here; in future, this can be subject to calibration.

Note that the WU-E model employs an algorithm in which only fuels within a finite distance are considered WUI fuels and are included in the heat-flux contribution to structural fuels. This approach facilitates fire spread between regular wildland fuels, typically represented by wildland fire models (e.g., Rothermel-based models), and structural fuels.

This distance is defined by a user-specified parameter in the `&WUI` namelist, `BANDTHICKNESS_WUI`. By default, `BANDTHICKNESS_WUI=5`, corresponding to a neighborhood of computational cells within an 11×11 cell square centered on each structural cell. Increasing this parameter allows heat-flux contributions from more distant vegetative fuels to be included, but it also significantly increases computational cost.

For typical applications with a grid resolution of approximately 30 m, `BANDTHICKNESS_WUI` values between 3 and 5 are generally sufficient.

Apart from the outputs that can be generated by the original ELMFIRE, when WU-E model is activated

(BLDG_SPREAD_MODEL_TYPE is set to 2), additional outputs on HRR and incident heat flux can be generated as the following:

- DUMP_TOTAL_DFC_RECEIVED: total accumulated incident energy from direct flame contact (kJ/m^2)
- DUMP_TOTAL_RAD_RECEIVED: total accumulated incident energy from radiation (kJ/m^2)
- DUMP_TRANSIENT_DFC: incident heat flux from direct flame contact (kW/m^2)
- DUMP_TRANSIENT_RAD: incident radiative heat flux (kW/m^2)
- DUMP_FUEL_CONSUMPTION: total consumed structural fuel load at a given time (kJ/m^2). Fuel consumption is allowed only when the total accumulated incident energy from direct flame contact and radiation exceeds the critical heat-flux threshold specified by CRITICL_HF_WUI in the &WUI namelist. The current consumption model is a placeholder for future development.
- DUMP_HRR_TRANSIENT: heat release rate per unit area at a given time (kW/m^2)

Extra outputs can be specified with the building spread model. Turning ESTIMATE_URBAN_LOSSES=.TRUE. in the &OUTPUTS namespace will include the results of the building fire spread to the affected population and surface fire area statistics.

1.14 Suppression

ELMFIRE can represent the effects of wildfire suppression on the advancing fire front. The suppression module does not simulate individual firefighting resources or tactical actions explicitly. Instead, it provides simplified representations of initial attack and extended attack that modify fire growth and containment during a simulation.

1.14.1 Initial Attack

Initial attack can be activated through the switch ENABLE_INITIAL_ATTACK in the &SUPPRESSION namelist. The attack time is specified using INITIAL_ATTACK_TIME in seconds. At the specified time, ELMFIRE estimates the probability of successful initial attack based on the current fire area and average fireline intensity. If the initial attack successfully contains the fire, the simulation terminates.

1.14.2 Extended Attack

Extended attack is activated by setting

```
ENABLE_EXTENDED_ATTACK = .TRUE.
```

Two extended-attack formulations are available through EXTENDED_ATTACK_MODEL:

- EXTENDED_ATTACK_MODEL = 0: area-growth-based containment model;
- EXTENDED_ATTACK_MODEL = 1: spatially explicit suppression model.

Area-Growth-Based Extended-Attack Model

An example configuration for the Area-Growth-Based containment model is

```
&INPUTS
...
SDI_FILENAME               = 'sdi'
```

```
...

&SUPPRESSION
ENABLE_EXTENDED_ATTACK     = .TRUE.
EXTENDED_ATTACK_MODEL      = 0
DT_EXTENDED_ATTACK         = 3600
AREA_NO_CONTAINMENT_CHANGE = 10000
MAX_CONTAINMENT_PER_DAY    = 100
SDI_FACTOR                 = 1
USE_SDI                    = .TRUE.
USE_SDI_LOG_FUNCTION       = .TRUE.
B_SDI                      = 1
```

The Area-Growth-Based containment model estimates changes in containment at intervals defined by `DT_EXTENDED_ATTACK`. At each interval, the model compares the change in fire area with `AREA_NO_CONTAINMENT_CHANGE` and uses this information, together with the Suppression Difficulty Index (SDI), `MAX_CONTAINMENT_PER_DAY`, and `B_SDI`, to determine the change in containment.

When SDI is enabled, spatial differences in suppression difficulty modify the effective change in fire area. `SDI_FACTOR` provides an additional scaling factor that can be used for calibration or sensitivity analysis. The relationship between fire-area growth and containment change is linear by default and can be changed to a logarithmic formulation using `USE_SDI_LOG_FUNCTION`. If SDI data are unavailable, `USE_SDI` can be set to `.FALSE.`, in which case a uniform SDI factor of 1 is applied.

Spatially Explicit Extended-Attack Model

The new extended-attack model is activated by setting

```
&SUPPRESSION
ENABLE_EXTENDED_ATTACK           = .TRUE.
EXTENDED_ATTACK_MODEL            = 1

ENABLE_INDIRECT_ATTACK           = .TRUE.
EXTENDED_ATTACK_TIME             = -1.0
INITIAL_CONTAINMENT_SHAPE_FACTOR = 10
AVAILABLE_SUPPRESSION_CAPACITY   = 2000
FIRELINE_LENGTH_REF              = 15000

PCL_THRESHOLD                    = 30.0
FL_MAX_DIRECT_ATTACK             = 8.0
SDI_MAX_DIRECT_ATTACK            = 100.0

FIRE_LINE_THICKNESS              = 1
DELTA_ROS                        = 10
DELTA_FL                         = 2
DELTA_SDI                        = 10
DELTA_PCL                        = 10
```

The spatially explicit model represents suppression through two linked components: direct attack on the active fireline and indirect attack at favorable control locations. Suppression is evaluated during the fire-spread simulation so that successfully suppressed portions of the fireline influence subsequent fire growth.

Suppression Capacity and Deployment Delay: `AVAILABLE_SUPPRESSION_CAPACITY` specifies the maximum suppression capacity available to the incident in units of meters of fireline per hour. The full capacity does not need to be available immediately after ignition. Instead, the model allows suppression capacity to increase from zero to its specified maximum as resources are deployed.

The deployment period can be defined in two ways using `EXTENDED_ATTACK_TIME`. If a positive value is specified, it represents a fixed time, in seconds, at which the full suppression capacity becomes available. This option can be used when an approximate mobilization or deployment time is known.

If `EXTENDED_ATTACK_TIME` is negative, the model estimates the deployment time from the early growth of the active fireline. The average fireline-growth rate during the first day of simulation is calculated and used to estimate the time required for the fireline to reach `FIRELINE_LENGTH_REF`. `FIRELINE_LENGTH_REF` therefore represents the expected active fireline length, in meters, at which full suppression capacity is deployed. Because this option relies on fireline growth during the first day, the simulated fire duration must exceed one day.

The parameter `INITIAL_CONTAINMENT_SHAPE_FACTOR` controls the shape of the capacity buildup from zero to the full available capacity. Smaller values produce a more gradual increase, whereas larger values delay a larger fraction of the capacity until closer to the estimated or specified full-deployment time and therefore produce a sharper increase in suppression capability.

Direct Attack: Direct attack operates on the active fire edge. The model first identifies active fireline cells and groups neighboring cells with similar fire behavior and suppression conditions into fireline segments. Segment formation considers fireline type, rate of spread (ROS), flame length, SDI, and Potential Control Location (PCL).

The parameters

`DELTA_ROS`
`DELTA_FL`
`DELTA_SDI`
`DELTA_PCL`

define the allowable differences between neighboring cells when constructing fireline segments. Their units are ft min^{-1} for `DELTA_ROS`, ft for `DELTA_FL`, and the corresponding scaled SDI and PCL units for `DELTA_SDI` and `DELTA_PCL`. `FIRE_LINE_THICKNESS` specifies the number of raster cells used to represent the active fireline.

For each fireline segment, the model evaluates the feasibility and relative difficulty of direct attack using flame length and SDI. `FL_MAX_DIRECT_ATTACK` specifies the reference flame length for direct attack, with a suggested default value of 8 ft. `SDI_MAX_DIRECT_ATTACK` specifies the corresponding reference SDI value, with a suggested default value of 100. For SDI products stored as SDI multiplied by 100, values may extend to approximately 318 for the USDA SDI product.

Segments with more favorable fire behavior and suppression conditions receive higher priority for direct attack. Available suppression capacity is progressively allocated among eligible segments, while increasingly difficult segments consume a larger fraction of the available capacity.

Indirect Attack: Indirect attack can be enabled or disabled using

```
ENABLE_INDIRECT_ATTACK = .TRUE.
```

When enabled, the model uses PCL information to identify favorable locations where the advancing fire may be stopped. `PCL_THRESHOLD` defines the minimum PCL value used to identify candidate control locations. A suggested default value of 30 is used for PCL data represented on a 0–100 scale.

Spatially connected candidate cells are grouped into control-location segments, and the representative PCL value of each segment is calculated from the mean PCL of its cells. When the advancing fire reaches a candidate control location, the model estimates its probability of holding based on the segment PCL and the local fire behavior. Increasing flame length reduces the probability that the control location will hold. A random value drawn from a uniform distribution is then compared with the calculated holding probability. If the control location holds, fire spread across the affected portion of the segment is prevented; otherwise, the location is breached and the fire continues to spread.

The parameters shown above provide suggested default values and can be modified to represent different suppression environments, input datasets, or calibration requirements.

1.15 Fire Potential Mode

So far, ELMFIRE has been set to estimate the fire propagation based on some starting conditions and (potentially) transient weather. This is similar to the function of FARSITE. ELMFIRE is also capable of estimating the fire risk of an area at a given weather, much like the FLAMMAP program. This behavior is controlled through the MODE variable.

The modes that ELMFIRE has are:

- MODE = 1: The standard ELMFIRE spread model, with (potentially) transient weather, ignitions, firebrands, level set propagation.
- MODE = 2: Fire potential mode. No ignitions are provided, instead ELMFIRE calculated the maximum rate of spread and potential crown fire activity for all cells of the domain, for each weather band.
- MODE = 3: Both output types from above.

1.16 Assets at Risk

An important part of quantifying fire risk is assessing potential impacts to assets at risk. ELMFIRE is currently capable of quantifying impacts to three assets at risk. Since some of the most relevant assets at risk are population, real estate/structures, and land/timber, the keywords that are used to specify assets at risk in the &INPUTS namelist group are hardcoded as REAL_ESTATE, POPULATION_DENSITY, and LAND_VALUE although each of these is treated identically. Since other assets at risk may be relevant (cultural resources, sensitive habitat, watershed, etc.), the total number of assets at risk eventually will be expanded. However, for the time being, assets at risk are specified as follows:

- Land value: Set USE_LAND_VALUE = .TRUE. and specify the land value filename via LAND_VALUE_FILENAME
- Real estate value: Set USE_REAL_ESTATE_VALUE = .TRUE. and specify the real estate value filename via REAL_ESTATE_VALUE_FILENAME

- Population density: Set USE_POPULATION_DENSITY = .TRUE. and specify the population density filename via POPULATION_DENSITY_FILENAME

Note that assets at risk are read in from the FUELS_AND_TOPOGRAPHY directory. Additionally, all assets at risk should be Float32 GeoTiff rasters. Units should be quantity per acre, e.g. structures per acre, population per acre, $ per acre, etc.

If assets at risk rasters are provided as input, ELMFIRE will sum total impacts by integrating fire area over asset at risk density and report this in the fire_size_stats.csv output file. ELMFIRE will also create impact rasters affected_land_value.tif, affected_real_estate_value.tif, and affected_population.tif.

1.17 Pyrome Calibration

Large areas of the landscape can be categorised or delineated based on their historic fire conditions, or the way that wildfires historically spread and behave. Such areas are called Pyromes, and outline areas of similar fire behavior beyond having similar fuels, weather patterns or topography. Such delineations require expert judgement or knowledge of the local fire landscape, though an initial delineation of the continental United States, along with guides on how they can be generated, can be found in Short et al.

If the simulation area contains multiple pyromes, the fire behaviour of each area can be adjusted. Pyrome adjustment can be activated through USE_PYROMES = .TRUE. in the &SIMULATOR namelist. A raster named PYROMES_FILENAME.tif can be specified in the &INPUTS namelist. Each point in that raster is assigned an integer, specifying which pyrome it belongs to. Subsequent Pyrome files belong to the &CALIBRATION namelist can be specified to make adjustments to various parameters depending on which pyrome they belong to. As an example, below is a sample file which contains different adjustment factors, ADJUSTMENT_FACTORS_FILENAME (which requires ADJUSTMENT_FACTORS_BY_PYROME = .TRUE. to be used):

```
   108 121 162 163 181 202
1  1.0 1.1 1.1 0.9 1.0 0.8
2  1.1 1.0 0.9 1.0 1.2 1.0
3  1.2 1.0 1.0 1.1 1.1 0.9
4  1.0 0.8 1.1 0.9 1.0 0.8
```

In the case of ADJUSTMENT_FACTORS_FILENAME, each of these values adjusts the predicted surface and crown fire rates of spread. For example, within pyrome 3, surface and crown fire rate of spread will be multiplied by 1.2 for cells with fuel model 108, 1.0 for fuel model 121 and so on.

If CALIBRATION_CONSTANTS_BY_PYROME is set to .TRUE., and accompanying CALIBRATION_CONSTANTS_FILENAME file should be provided, with the structure:

```
   InitialAttackTime BSDI MAXCPERDAY AREANCCHANGE SIMDURATION IGNDENSITYADJ
1  3600              0.9  100        10000        16000       0.8
2  3600              1.0  90         8500         12000       1.0
3  3600              1.0  100        10000        16000       0.9
4  1800              0.9  100        10000        14000       0.8
```

The spaces between the values are just added here to illustrate which column they correspond to, they are not necessary. The columns are, in order, suppression initial attack time, adjustment factor $B_S DI$, max

containment per day, area of no containment change, ELMFIRE simulation duration and ignition density (probability) adjustment. See the Suppression section for more details on the first four parameters, and the monte carlo (ignition) section for the latter two.

For long-term monte carlo simulations, another way to set the total simulation time is by providing probability density functions to individual day durations. By setting DURATION_PDF_BY_PYROME = .TRUE., an input file DURATION_PDF_FILENAME can be specified:

```
   1     2     3   ... 364   355
1  0.000 0.001 0.050   0.000 0.000
2  0.000 0.001 0.001   0.010 0.000
3  0.000 0.001 0.100   0.000 0.000
...
```

Each value denotes the probability that the duration of the simulation will be equal to that day duration. A hard limit of DURATION_MAX_DAYS is set, so no simulation can be longer.

1.18 Smoke

ELMFIRE can calculate the smoke emissions of any given simulation for use with external dispersion models. As of this version, ELMFIRE treats each simulation as one source of dispersants (meaning any output files will treat any part of the burning landscape as one source instead of splitting them apart). It is assumed that the main dispersant that users may be interested in is PM2.5, so most program constants have been selected based on this assumptions. They have been made user-modifiable if a different pollutant is required.

Smoke calculations can be enabled by setting `ENABLE_SMOKE_OUTPUTS=.TRUE.` in the `&SMOKE` namelist. Enabling this feature will result in the output of an additional `smoke_000001.csv` file being generated, containing the simulation time in hours, the timestamp of the simulation, the x and y center of the burning area of that time interval (geometric center, does not get weighted by any parameter as was the case in previous versions), the total burned area of that time interval, the average HRR in MW and the average pollutant injection per hour. The center of the burning area is given in units related to the projection of the input geotiff rasters.

The dispersion is based on a simple calculation relating total energy released by the wildfire at time interval `DT_SMOKE_OUTPUTS`, energy released per kg of wood (used to estimate total burned mass, with a constant `DRY_WODD_CALORIFIC_VALUE=19MJ/kg`, later adjusted by the 10-h fuel moisture content), and the pollutant released per kg of burned wood (`PM_EMISSION_FACTOR_FLAMING` and `PM_EMISSION_FACTOR_SMOLDERING` for the two burning phases). The total flaming and smoldering time (also referred to as residence time) can also be adjusted from their default values, using inputs `FLAMING_TIME` and `SMOLDERING_TIME` in seconds.

This module was made for use with external dispersion program, with HYSPLIT as a main example. For a better integration with HYSPLIT, the `EMITIMES.txt` file can be written directly by ELMFIRE. `EMITIMES.txt` is used to specify complex, transient or moving sources, all three of which are the case with an evolving fire front. To generate this file, set `DUMP_EMITIMES=.TRUE.` in the `&OUTPUTS` namelist.

1.18.1 Running HYSPLIT

In the examples/ folder we have provided a set of helper scripts to run elmfire simulations, whose outputs can be used to run HYSPLIT. HYSPLIT requires the following files to run a simulation:

- SETUP: global simulation settings
- EMITIMES.txt: emission and fire size entries per hour
- CONTROL: specific simulation parameters
- ASCDATA.CFG: simulation domain config file

The examples/smokeHelpers/elmfire2hysplit.py file helps prepare all the needed files for a HYSPLIT run, based on an ELMFIRE simulation. The helper script creates the CONTROL, SETUP, ASCDATA.CFG files with some baseline options setup. Advanced users are free to change these values. The helper file also automatically downloads the weather files required by HYSPLIT that describe the global composition of the atmosphere (weekly GDAS1). The standard source of these files is US NOAA.

The simulation is then run automatically via examples/smoke/elmfire_hysplit.sh, which handles everything from running the ELMFIRE simulation, setting up the necessary folders, and running HYSPLIT. The final files are in a binary format that is difficult to use, and to help convert them to a format that is more friendly to GIS systems, bin2nc.py is provided. It converts the smoke.bin output to smoke.nc, which can then be opened with programs like QGIS. bin2nc.py is automatically run within elmfire_hysplit.py.

smoke.nc can then be imported directly to GIS applications. With QGIS, specifically, two features will be imported, one with individual smoke raster bands, and one with no bands and just a premade colormap of PM2.5 concentration in $\mu g/m^3$. This file is already temporally enables, and Temporal Controller can be used to scroll through the simulation daterange.

1.19 Memory Optimization

ELMFIRE has been developed from the start to be a fast wildfire spread model, with great parallel processing and monte-carlo processing capabilities. These capabilities require that all data is loaded in memory at the start of model calculations. For the majority of cases there is no issue with this setup, but for cases with larger rasters and longer run times (in the order of 2000 x 2000 points and 800 hours or above), RAM usage can reach 60 GB and be unsustainable for many systems. The problem stems from the need to load all temporal data in memory at once, with larger rasters requiring 150 MB of RAM per weather band.

To reduce the RAM usage of ELMFIRE, a MEMOPT version of the code has been developed that requires far less RAM resources to run, for the price of longer run times. ELMFIRE-MEMOPT only loads part of the temporal weather rasters in memory at a time. Once all processes/simulations are finished with the current weather band range, a new one is loaded. This can bring the RAM requirement of a large simulation down from 60 GB to 5 GB, and increase the simulation run time by 10 % (depending on the case). The switch that controls how many weather bands are loaded at a time is WX_BANDS_KEPT_IN_MEM, in the &SIMULATOR namelist. A standard value of 30 has been set as standard based on a balance of runtime and RAM usage.

1.20 Docker and Containerisation

The ELMFIRE distribution also provides a docker image to run ELMFIRE without the need to install any packages or specific python versions. Using it should be straightforward if you are familiar with docker; this guide has been written for people with no docker familiarity.

Running ELMFIRE through docker ascertains that ELMFIRE will run as expected every time, avoiding the 'it works on my machine' debugging problem. Users can just mount a pre-made image, run ELMFIRE,

and save the results without any further installations needed.

You can install docker either through its desktop version or though a more minimal command line version. If using ELMFIRE on windows via WSL2, you can download docker for desktop on windows and then enable docker for WSL (some setting changes are needed).

Once you have installed docker in your system, `cd` to the directory where elmfire is installed. That directory should have a Dockerfile and a docker-compose.yml file. Within that directory, first build the image. This creates the environment where ELMFIRE can run. It usually takes a relatively long time (about 10 mins) to download all dependencies, python packages, setup environment variables, and compile ELMFIRE.

```
docker compose up -d --build
```

This command will build and activate the ELMFIRE image. To then use this image,

```
docker compose exec elmfire bash
```

This will activate a bash terminal within the docker image where you can use ELMFIRE just like in any other linux installation. The examples, tutorials and verification cases all work within the image. Once you are done, to stop using up resources, you can stop the image through:

```
docker compose down
```

You do not need to build the image again to activate it, you only need to build it if there are changes made to the ELMFIRE distribution.

Please note that once you bring down the image, all changes and files generated will be deleted. The one exception to this, and the way that a docker version of ELMFIRE becomes useful, is through the `docker_shared_folder` that appears once you build the image. The contents of that folder are visible to both the docker image and the host machine and persist even when the docker image is brought down. A viable workflow is to then copy specific cases into that folder, run ELMFIRE through the image, and retrieve the results. Note that sometimes there are issues with folder ownership and you may be prohibited from copying files into the generated folder. In this case, you might need to change ownership of the folder to your username through `chown`.

1.21 Data Sources and Helper Scripts

This section is devoted into compiling data sources and helper scripts that can be used alongside ELMFIRE to either compile all the data needed for the simulation or pre- / postprocess the results. Note that this list is a compilation of frequently used resources and does not necessarily reflect implicit approval of the accuracy of the data or continuous support alongside ELMFIRE.

- FireDX [US / GLOBAL]: Automatic generation of urban fire spread input rasters. `https://github.com/ma-th/firedx`
- utils/makeBarrierFile.py [GLOBAL]: Create a valid barrier file based on OSM roads and rivers.
- LANDFIRE [US]: Download complete landscape files for any US area. `https://landfire.gov`
- CWFIS [CANADA]: Download topography, fuel, weather, and danger index information for all of Canada. `https://cwfis.cfs.nrcan.gc.ca/downloads/docs/en/how-tos/how-to-access-cwfis-data-services.pdf`

- MesoWest [US]: (Scheduled for sunsetting by end of 2026) Automated Weather Station data repository. `https://mesowest.utah.edu`

1.22 Process Flowchart

ELMFIRE Workflow:

1. Setup variables
2. Setup MPI
3. Read and check input files
4. Set randomizer seed
5. Calculate sunrise and sunset times
6. Calculate and save lookup tables (for easy/fast access)
7. Read headers of weather, fuels & topography
8. Read weather, fuels & topography raster data
9. Determine number of cases to run
10. Rotate wind if GRID_DECLINATION
11. If pyromes are used, parse adjustment factors
12. Remap the weather input grids (coarse) to the simulation cellsize (fine)
13. Allocate memory for additional rasters (outputs, intermediate rasters etc.)
14. (if Random Ignitions) Determine ignition locations
15. Allocate memory for output rasters
16. If in fire potential mode (MODE = 2 or 3):
    (a) Initialize spread output rasters
    (b) Keep track of all burnable cells (burnable fuel types and outside of Edgebuffer range)
    (c) For each weather band:
        i. Calculate surface and crown spread rates
        ii. Check surface flame length, if high enough then apply crown fire calculations.
        iii. Output spread raster values
17. If in fire spread mode (MODE = 1 or 3)
    (a) (ensemble monte carlo) initialize random parameters and pretrubations.
    (b) For each scenario call level set propagation algorithm.
    (c) Accumulate ensemble statistics
    (d) Run postprocessing for advanced outputs
18. Shutdown

Main level set propagation algorithm:

Since the first Memory Optimisation (MEMOPT) version of ELMFIRE, the structure of the level set method has been changed to:

1. Check if new weather band range is needed and update weather band range loaded in memory
2. If ignition time is reached:
    (a) Determine simulation stop time
    (b) Allocate intermediate rasters
    (c) Determine output time interval
    (d) Allocate output rasters
    (e) Set pointer addresses

(f) If random ignitions are on:
    i. Set ignition location
    ii. Adjust simulation time and suppression parameters through Pyrome
    iii. Exclude and set outputs if nonburnable cell is selected for ignition
(g) Check that weather band exists for this time step
(h) Initialize weather, suppression, point list variables
(i) If random ignitions are not enabled:
    i. Check each landscape point for $\phi < 0$, set as burning, and set initial weather parameters
    ii. Calculate surface and crown rates of spread for all points currently burning
    iii. Calculate front normal vectors, calculate X and Y ROS components
    iv. Check for eligible fire spread cells within fire front BANDTHICKNESS

3. Start the main loop until stop condition is reached:
    (a) If applicable, calculate overnight adjustment factor time span
    (b) If random ignitions are on, set ϕ of ignition point to zero and save potential spread points.
    (c) If multiple ignitions are specified in the input, check if any new ignitions apply.
    (d) Interpolate current weather values
    (e) Apply wind fluctuations
    (f) Calculate surface spread rate
    (g) Calculate crown spread rate
    (h) Calculate components of normal vectors of fire front
    (i) Calculate Ux and Uy velocity components
    (j) If FLIN is high enough for crown fire, apply results of crown spread rate to total.
    (k) Calculate new time step with CFL and Flux limiter
    (l) Integrate level set equations with Runge Kutta 2nd order
    (m) Update burning point values and statistics
    (n) If enabled, save binary outputs
    (o) If enabled, calculate new spot fire locations
    (p) If UMD spotting is enabled, calculate physical parameters and run spotting model.
    (q) Calculate and clear extinguished embers (separate processes for eulerian and lagrangian)
    (r) Calculate interpolated weather values for newly-included cells
    (s) Calculate surface and crown spread rates for new cells
    (t) If applicable, calculate initial attach probability
    (u) Stop simulation if less than two points are burning, or if a max acreage criterion is reached.
    (v) If applicable, append acreage timings statistics
    (w) Calculate extended attack containment
    (x) Dump main outputs
4. Advance simulation time
5. If simulation stop time has been reached:
    (a) Process and output timed location data (does not apply to newer versions).
    (b) Calculate and output fire size statistics
    (c) If applicable, dump binary outputs
    (d) If applicable, dump ember count bin measurements.

Chapter 2

Mathematical Background

2.1 Introduction

2.2 Wildfire Spread

The foundation of ELMFIRE is the Rothermel model [23, 2], which estimates the head fire rate of spread in the direction of the wind and slope combined. An extensive and thorough guide has been compiled by Andrews [5] covering all the aspects of the model, the input parameters, experimental background and corellations between values. For completeness, and consistency of units used in ELMFIRE, critical parts of the model will be repeated in this section.

The Rothermel model, at a high level, outlines the rate of spread of a wildfire as an energy argument, specifically as the ratio of the energy produced by the fire front and the energy required for ignition. Rigorously, it is:

$$R = \frac{I_R \xi (1 + \phi_w + \phi_s)}{\rho_b \epsilon Q_{ig}} \tag{2.1}$$

In general terms, the numerator is the propagating flux, and the denominator is the heat required to ignite the next parcel of fuel. Analytically:

- R: Rate of Spread (ft/min): 1D velocity of the flaming front aligned with prevailing wind and slope.
- I_R: Reaction Intensity ($BTU/ft^2/min$), energy released per unit area of fire front.
- ξ: Propagating Flux Ratio, fraction of released energy that reaches unburned fuel under no wind/slope.
- ϕ_w: Wind Factor, dimensionless factor accounting for the effect of wind on the rate of spread.
- ϕ_s: Slope Factor, dimensionless factor accounting for the effect of slope on the rate of spread.
- ρ_b Fuel Bulk Density (lb/ft^3), oven-dry density of surface fuel.
- ϵ: Effective Heating Number: Ratio of unburned fuel reaching ignition temperature before ignition.
- Q_{ig}: Heat of Preignition (BTU/lb): Energy required to ignite 1 lb of fuel.

Of the above values, almost none are material values and they all need to be calculated through further equations and correlations. ELMFIRE makes no modifications to the Rothermel model as outlined in Andrews [5]. Note that english units are used in the original model formulation, but metric equivalent equations have been developed.

From the above formulation of the Rothermel equation it may seem that the wind and slope must be

aligned, and are then linearly added to the no-wind, no-slope rate of spread. Most of the times, the direction of maximum slope and the wind direction will not be matching, and need to go through vector addition instead. To do this, we can rewrite the Rothermel model as:

$$R = V_{s,0}\alpha(1 + \|\vec{\phi_w} + \vec{\phi_s}\|) = V_{s,0}\alpha(1 + \|\vec{\phi}\|) \tag{2.2}$$

where $V_{s,0}$ is the no-wind, no-slope rate of spread estimated by the Rothermel model, and α is a constant added by ELMFIRE to account for point source acceleration (only relevant in the early stages of the fire) such that $0 \leq \alpha \leq 1$. By assuming that the directional effects of wind and slope are independent, ϕ_x and ϕ_y (the x and y components of an "overall" $\vec{\phi}$) can be calculated with:

$$\phi_x = \alpha(\phi_{w,x} + \phi_{s,x}) = \alpha[\phi_w \sin(\theta_w - \pi) + \phi_s \sin(\theta_a - \pi)]$$

$$\phi_y = \alpha(\phi_{w,y} + \phi_{s,y}) = \alpha[\phi_w \cos(\theta_w - \pi) + \phi_s \cos(\theta_a - \pi)]$$

Then, by adding the constituents of the vector,

$$|\vec{\phi}| = \sqrt{\phi_x^2 + \phi_y^2}$$

Finally, the direction of maximum spread (the direction toward which the fire spreads most rapidly, i.e. the head fire direction) is calculated as:

$$\theta_{DMS} = \begin{cases} \frac{\pi}{2} - \tan^{-1}\left(\frac{\phi_y}{\phi_x}\right), & \phi_x > 0 \text{ and } \phi_y \geq 0 \\ \frac{3\pi}{2} - \tan^{-1}\left(\frac{\phi_y}{\phi_x}\right), & \phi_x < 0 \text{ and } \phi_y \geq 0 \\ \frac{3\pi}{2} - \tan^{-1}\left(\frac{\phi_y}{\phi_x}\right), & \phi_x < 0 \text{ and } \phi_y < 0 \\ \frac{\pi}{2} - \tan^{-1}\left(\frac{\phi_y}{\phi_x}\right), & \phi_x > 0 \text{ and } \phi_y < 0 \end{cases}$$

An effective midflame windspeed can then be calculated to represent the combined effect of wind and slope as one effective wind value. Effective mid-flame wind speed is calculated by solving Equation 47 from [23] based on the 20ft wind speed and slope inputs, vector added to the parameter ϕ:

$$U_{m,f,e} = \left(\frac{|\phi|}{C\left(\frac{\beta}{\beta_{op}}\right)^{-E}}\right)^{\frac{1}{B}} \tag{2.3}$$

The parameters C, β, β_{op}, E, and B are defined in [23] and depend on the fuel model.

2.2.1 The Canadian Fire Spread Model

ELMFIRE can also use the Canadian FBP model to estimate maximum rate of spread. all other parts of ELMFIRE, such as the elliptical rate of spread breakdown and canopy fire effects, stay the same. The FBP system is a purely empirical set of models, derived from historic observations of natural and prescribed

wildfires throughout Canada. Its development is well documented in a series of reports by the Canadian Forestry Service [30, 1, 32, 28]. The main equations and workflow of the models will be repeated in this section, but no modifications are made in ELMFIRE.

The main equation for calculating the maximum rate of spread is shown below. The equations in the section below broadly apply to all fuel models, although some special cases apply for fuel models O1a, O1b (grass fuels), M1, M2, M3 and M4 (mixedwood stands).

$$RSI = a(1 - \exp(-b \times ISI))^c \times CF \times BE \tag{2.4}$$

Where a, b and c are fuel model parameters, ISI is the initial spread index, part of the FWI, CF is the curing factor, only relevant for grass-type fuels, and BE is the build-up effect, a function of the Build-Up Index (BUI).

$$BUI = \frac{0.8 \times DMC \times DC}{DMC + 1.4DC}$$

DMC and DC are calculated using a starting value and an incoming weather stream, with no changes from the standard fire weather index calculations.

ISI is a function of wind and fuel moisture. Note that "wind" in this function represents effective wind, a vector combination of actual measured wind (at 10 m height) and wind-equivalent slope.

$$ISI = 0.208 f(F) f(W)$$

$$f(F) = 91.9 \, \exp(-0.1386 \, M) \left(1 + \frac{M^{5.31}}{4.93 \times 10^7}\right)$$

$$f(W) = \begin{cases} \exp\left(0.05039 \, \bar{U}\right), & \text{if } 1.61 \, U \leq 40, \\ 12 \left(1 - \exp\left(0.0818 \left(\bar{U} - 28\right)\right)\right), & \text{if } 1.61 \, U > 40 \end{cases}$$

Where M is the fine fuel moisture content (set as the 1h fuel moisture content in ELFMIRE, ranging from 0 to 1), $\bar{U}$ is the effective wind speed, and U is the actual 10m wind speed in kph. The process of finding the effective wind speed requires finding the slope-only rate of spread and converting that value back into an equivalent wind speed. This is done by:

$$RSF = RSZ \times SF \tag{2.5}$$

$$SF = \begin{cases} \exp\left(3.533 \left[\tan(a)\right]^{1.2}\right), & \text{if } a \leq 63, \\ 10, & \text{if } a > 63 \end{cases}$$

$$RSZ = a(1 - \exp(-b \times ISI_0))^c \, , \quad ISI_0 = 0.208 f(F)$$

Where a is the slope in degrees. RSF is then used in reverse to find the wind-based ISI for the given slope, yielding a slope-equicalent wind. Vector addition of the slope-equivalent wind and the actual wind then produce the effective wind $\bar{U}$.

2.2.2 Wind Scaling

As might have been hinted at in the previous section, the wind factor in the Rothermel equation depends on the midflame windspeed, that is the wind at the height of the flames. Most weather stations and weather measurements in the world require wind be measured at some height above ground, to capture "free wind velocity" and avoid ground effects. In the US, that height is 20ft, while in other parts of the world it is 10m.

ELMFIRE as standard expects wind inputs in mph and at 20ft height. Readings at 10m height are also acceptable, and are internally converted to 20ft readings by multiplying them by a constant 0.87. Then the wind needs to be scaled again to be brought down to midflame height. This requires the calculation of the wind adjustment factor, which is explained in depth in [4]. This final scaling depends on the flame length, which is assumed to be equal to the fuel bed depth. It also depends on the existence of canopy, as it would further reduce the midflame wind speed. For unsheltered surface fires (in the case of ELMFIRE, cases with canopy cover percentage of 0), the following conversion is used:

$$U_{mf} = U_{20} \times WAF$$

$$WAF = \frac{1.83}{\ln\left(\frac{20+0.36H}{0.13H}\right)} \tag{2.6}$$

For sheltered surface fires:

$$WAF = \frac{0.555}{\sqrt{fH}\ln\left(\frac{20+0.36H}{0.13H}\right)} \tag{2.7}$$

Where:

- H: Fuel bed depth (ft)
- $f = \frac{CC}{100}\frac{\pi}{12}$
- CC: Canopy Cover (%)

When the Canadian FBP system is used, and Fuel bed depth is not a parameter that the fuel model system stores, an alternative WAF calculation method is used based on [26], using canopy cover (CC):

$$WAF = \begin{cases} 0.10 & \text{if } CC > 50 \\ 0.15 & \text{if } 30 < CC \leq 50 \\ 0.20 & \text{if } 15 < CC \leq 30 \\ 0.25 & \text{if } 10 < CC \leq 15 \\ 0.30 & \text{if } 5 < CC \leq 10 \\ 0.50 & \text{if } CC \leq 5 \end{cases}$$

This midflame windspeed can then be used to calculate the wind adjustment factor.

2.2.3 Directional Rate of Spread

The Rothermel model provides the head fire rate of spread; the spread rate at any angle from the head fire is needed for landscape spread calculations. ELMFIRE implements the elliptical wildfire propagation proposed by Anderson [3] and later modified by Finney [9]. This theory suggests that wildfire spread from a point source can be described by an ellipse, following from experimental observation and a balance for ease

of mathematical description. The equations describing the dimensions of the ellipse ultimately can be solved to find the rate of spread about any angle ω from the maximum spread direction. The dimensions of the ellipse are dictated by the following equation, describing the Length/Width ratio of the ellipse based on the effective midflame windspeed in mph $U_{mf,e}$:

$$\frac{L}{W} = \min\left[0.936\, e^{0.1147\, U_{mf,e}} + 0.461\, e^{-0.0692\, U_{mf,e}} - 0.397,\ 8\right] \tag{2.8}$$

As in FARSITE, the maximum value of L/W is limited to 8 by default but in ELMFIRE this is a user-specifiable parameter. ELMFIRE computes this equation directly, compared to older versions of the code that used polynomial approximations.

The equations below show how the rate of spread is calculated, with $V_{DMS,\|}$ being the rate of spread at direction of maximum spread parallel to the ground, V_{s0} being the backing rate of spread assumed equal to the no-wind/no-slope spread rate, and ω being the angle from the maximum spread direction.

$$U^*_{y,\|} = \frac{a^2 \cos(\omega)}{\sqrt{a^2 \cos^2(\omega) + b^2 \sin^2(\omega)}} + c \tag{2.9}$$

$$U^*_{x,\|} = \frac{b^2 \sin(\omega)}{\sqrt{a^2 \cos^2(\omega) + b^2 \sin^2(\omega)}} \tag{2.10}$$

$$a = \frac{1}{2}\left(\left|V_{DMS,\|}\right| + V_{s0}\right) \tag{2.11}$$

$$b = \frac{1}{2}\left|V_{DMS,\|}\right| + V_{s0}\frac{L}{W} \tag{2.12}$$

$$c = \frac{1}{2}\left(\left|V_{DMS,\|}\right| - V_{s0}\right) \tag{2.13}$$

2.2.4 Canopy Fires

A review of canopy fire modelling systems can be found in [25]. The one implemented in ELMFIRE is similar to that in Farsite [9, 10]. The criterion for a canopy fire starting is the critical fireline intensity of the surface fire, I_0,depends on the height of the crown base (CBH, m) and the canopy foliar moisture content (M, %), following Van Wagner's model [31] and the adjustments from Cruz et al [6]. It is calculated by:

$$I_0 = [0.010 CBH(460 + 25.9M)]^{3/2}$$

Then the "intensity" of the crown fire is dictated by another value, the critical minimum rate of spread for active canopy fire, R_0, calculated through the Crown Bulk Density, kgm^{-3}:

$$R_0 = 3.0/CBD$$

And the predicted rate of spread of the canopy fire, $R_{C,P}$ can be calculated using the CBD, 10m wind in kph (note the unit and reference height change), and M_1 being the 1-hour fuel moisture content of the surface fuel. Due to its direct dependence on wind, this model is not applicable for zero-wind conditions, which would only be encountered in idealized conditions. hence this case will be run similarly to the "Windy"

case.

$$R_{C,P} = 11.02 \times U_{10}^{0.9} \times CBD^{0.19} \times \exp(-0.17 \times M_1)$$

The combination of the two values can be used to determine if and what kind of crown fire occurs, depending on the surface fire fireline intensity I_b:

1. Passive Crown Fire ($Ib \geq Io$ but $R_{C,P}/R_0 <= 1$)
2. Active Crown Fire ($Ib \geq Io$, $R_{C,P}/R_0 > 1$)
3. Independent Crown Fire ($Ib > Io$, $R_{C,P}/R_0 > 1$)

Then, if the fire is determined to be a passive crown fire, its rate of spread will be adjusted to:

$$R_C = R_{C,P} \times \exp\left(-CAC\right)$$

For an active crown fire, the following equation is used, as long as the canopy cover of that point in the landscape exceeds the critical canopy cover (set by default as 0.39):

$$R_C = R_{C,P}$$

The calculated rate of spread of the canopy fire, whether passive or active, is then converted to an effective wind factor, to directly affect the calculated surface rate of spread. This way ELMFIRE can directly use the surface spread model to also include crown fire acceleration effects. The new effective wind factor $\phi_{w,e}$ is the maximum value between the Rothermel wind factor, $\phi_{w,R}$, and the Cruz canopy wind factor $\phi_{w,C}$, calculated by:

$$\phi_{w,C} = \frac{R_C}{R_{R,0}} - 1$$

With $R_{R,0}$ being the no wind, no slope Rothermel predicted surface rate of spread. The calculation is then repeated to find the modified, surface-canopy coupled rate of spread.

2.2.5 Level Set Propagation

At this stage the head rate of spread, the direction of primary spread, the rate of spread about any direction and canopy effects have been quantified. ELMFIRE can now use the level set method to spread the wildfire about a 2d landscape. The level set method is a common mathematical concept used to solve differential equations, find local maxima or minima, or solve for a specific variable, formulated in [16], building upon the work of [20]. In this case, ELMFIRE restructures the problem of fire spread across a landscape as a differential equation, solving for time. The exact formulation is:

$$\frac{\partial \phi}{\partial t} + U_x \frac{\partial \phi}{\partial x} + U_y \frac{\partial \phi}{\partial y} = 0$$

ϕ has no physical meaning except that the $\phi = 0$ isopleth (or level set) corresponds to the fire front. This provides a convenient way to track a curved surface, such as a fire front, on a regular grid. ELMFIRE integrates the governing PDE using a narrow-band formulation [27] with a second-order Runge–Kutta method and superbee flux limiters to prevent numerical oscillations.

In a previous step we had derived the values $U^*_{y,\|}$ and $U^*_{x,\|}$. Those are the y- and x-directions rate of

spread about the rothermel-calculated main spread direction, parallel to the ground. To use them in the level set method, they first have to be realigned about the main coordinate grid, and then projected flat. As such, the orthogonal rate of spread components can be calculated by:

$$U_{x,\|} = U^*_{y,\|} \sin(\theta_{DMS}) + U^*_{x,\|} \cos(\theta_{DMS})$$

$$U_{y,\|} = U^*_{y,\|} \cos(\theta_{DMS}) - U^*_{x,\|} \sin(\theta_{DMS})$$

And then to correct for slope:

$$\frac{U_x}{U_{x,\|}} = 1 - |\sin(\theta_a)| \ (1 - \cos(\gamma))$$

$$\frac{U_y}{U_{y,\|}} = 1 - |\cos(\theta_a)| \ (1 - \cos(\gamma))$$

With θ_α being the topographical aspect, and γ being the topographical slope. This correction has the correct limiting behavior, firstly resulting in 1 in flat terrain. Now take the case of an east-facing slope ($\theta_a = \pi/2$). Since $\sin(\pi/2) = 1$, $U_x/U_{x,\|} = \cos(\gamma)$. However, the y-direction spread rate is unaffected since $\cos(\pi/2) = 0$ and $U_y/U_{y,\|} = 1$.

The fire front can propagate only normal to itself. A key part of calculating the spread rate along the fire front is the normal vector to the ϕ field, representing the direction of spred of the fire front. This is shown graphically in Fig. 2.1.

$$\hat{n} = \frac{1}{|\nabla\phi|}\left(\frac{\partial\phi}{\partial x}\,\hat{i} + \frac{\partial\phi}{\partial y}\,\hat{j}\right) = n_x\hat{i} + n_y\hat{j}$$

$$|\nabla\phi| = \sqrt{\left(\frac{\partial\phi}{\partial x}\right)^2 + \left(\frac{\partial\phi}{\partial y}\right)^2}$$

Now we can define θ_n as the angle to which the ϕ field normal vector points about our reference grid system:

$$\theta_n = \begin{cases} \frac{\pi}{2} - \tan^{-1}\left(\frac{n_y}{n_x}\right), & n_x \geq 0,\ n_y \geq 0 \\ \frac{3\pi}{2} - \tan^{-1}\left(\frac{n_y}{n_x}\right), & n_x \leq 0,\ n_y \geq 0 \\ \frac{3\pi}{2} - \tan^{-1}\left(\frac{n_y}{n_x}\right), & n_x \leq 0,\ n_y \leq 0 \\ \frac{\pi}{2} - \tan^{-1}\left(\frac{|n_y|}{n_x}\right), & n_x \geq 0,\ n_y \leq 0 \end{cases}$$

The steps in solving the ODE are given in good detail in Rehm and McDermott [20], and they will be repeated here based on their explanation. At any time step t_n the gradient of ϕ can be calculated by central differentiation at each direction:

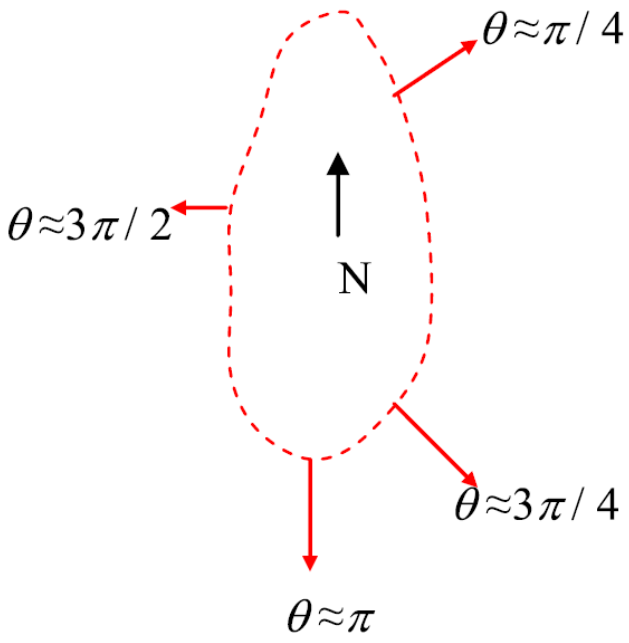


Figure 2.1: Graphical illustration of θ, the angle normal to fire front measured clockwise from North,at several locations along a sample fire perimeter.

$$\left(\frac{\delta\phi}{\delta x}\right)^n_{i,j} = \frac{\phi^n_{i+1,j} - \phi^n_{i-1,j}}{2\,\Delta x},$$

$$\left(\frac{\delta\phi}{\delta y}\right)^n_{i,j} = \frac{\phi^n_{i,j+1} - \phi^n_{i,j-1}}{2\,\Delta y}.$$

From these we can find the normal vector $\hat{n}$, and the angle θ_n of fire front spread. U_x and U_y can then be found by setting $\omega = \theta_{DMS} - \theta_n$. Before doing so, we set a flux limit to stabilise the ODE solver.

The central differentiation scheme used above can be reframed, using the ϕ values at the faces of the cell, in the example below the east and west face, instead of the centers of the adjacent cells:

$$\frac{\delta\phi}{\delta x} = \frac{\phi_{east} - \phi_{west}}{\Delta x}$$

We then define the local data ratio:

$$r = \frac{\phi_{i+1,j} - \phi_{i,j}}{\phi_{i,j} - \phi_{i-1,j}}$$

And the flux limiter function (Superbee):

$$B(r) = max[0, min(2r, 1), min(r, 2)]$$

The flux limiter can then be applied directly to the face values, such as:

$$\phi_{east} = \phi_{i,j} + \frac{1}{2} B(\phi_{i+1,j} - \phi_{i,j})$$

The rest of the face ϕ values can be computed similarly, based on the cell center ϕ values, and the original gradients of the ODE can be solved once again.

To integrate the ODE through time, the second order RK scheme is used. The time step Δt is controlled by the CFL criterion, set as standard as 0.2 but modifiable by the user. The mathematical formulation used in ELMFIRE is given below:

$$\phi^* = \phi^t - \Delta t \left(U_x^t \frac{\phi^t_{\text{east}} - \phi^t_{\text{west}}}{\Delta x} + U_y^t \frac{\phi^t_{\text{north}} - \phi^t_{\text{south}}}{\Delta y} \right) \tag{10a}$$

$$\phi^{t+\Delta t} = \tfrac{1}{2}\phi^t + \tfrac{1}{2} \left(\phi^* - \Delta t \left(U_x^* \frac{\phi^*_{\text{east}} - \phi^*_{\text{west}}}{\Delta x} + U_y^* \frac{\phi^*_{\text{north}} - \phi^*_{\text{south}}}{\Delta y} \right) \right) \tag{10b}$$

And thus, each time step a new ϕ is calculated and the fire front is allowed to progress. For the first time step, ignition is dictated by the points where $\phi = 1$.

2.2.6 Acceleration

The Rothermel model described above provides the equilibrium rate of surface fire spread, but in reality this equilibrium rate of spread is not achieved instantaneously [10, 16]. Consequently, for initial ignition locations and spot fires that form during a simulation, the rate of spread increases from zero to the equilibrium spread rate as follows:

$$\frac{R_{ac}}{R} = 1 - \exp\left(- \frac{t - t_{\text{ign}}}{\tau_{\text{accel}}} \right)$$

where T_{ign} is the time (s) that a point is ignited (by the initial conditions or by a spot fire, and τ_{accel} is a user specified acceleration time constant (s), usually on the order of 600.

2.3 Spotting

The process of a spot fire being created (generation, transport, ignition) can be broken up to its constituent processes, and each one can be studied separately. Much of the firebrand model in ELMFIRE is taken from the thesis of Yiren Qin from the University of Maryland [19], and will in large parts be reproduced below for completeness.

2.3.1 Generation

In ELMFIRE, the generation of firebrands is primarily controlled by two parameters: the **generation duration** (t_g) and the **generation rate** (N) of each computational cell. The firebrand generation duration is defined as the total time a cell is permitted to produce firebrands following its ignition. Ignition occurs when the local value of the level-set variable ϕ becomes less than or equal to 0. The generation rate N represents the total number of firebrands produced by a computational cell during a single level-set equation solver time step (Δt). Users may specify an arbitrary rate, use physics-based estimations derived from

literature correlations, or apply customized empirical parameters.

Three distinct approaches are available in ELMFIRE to define the generation rate:

- **Mode A**: N is randomly sampled from a bounded uniform distribution.
- **Mode B**: A constant generation rate is prescribed based on the number of firebrands per unit area per second.
- **Mode C**: A constant generation rate is prescribed based on the number of firebrands per megawatt of Heat Release Rate (HRR) per second (pcs/MW/s). In this mode, the generation rate N correlates linearly with the local HRR.

Modes B and C are components of the UMD spotting model.

Mode A: Stochastic Sampling

For Mode A, the upper (N_M) and lower (N_m) bounds of the generation rate are prescribed. Upon cell ignition, N is sampled from the uniform distribution Unif$[N_m, N_M]$, and the corresponding number of firebrands is generated during that time step. In this mode, firebrands are generated only once; thus, the effective generation duration is $t_g = \Delta t$.

Mode B: Area-Based Generation

Mode B defines the ember generation rate using a factor G_t, representing the number of firebrands per square meter per second (pcs/m^2/s). The generation rate is calculated as:

$$N = G_t A_c \min(\Delta t, t_g - t_e)$$

where A_c is the cell area ($A_c = c^2$, where c is the cell size), Δt is the level-set solver time step, and t_e is the elapsed time since the cell began emitting firebrands. The term $(t_g - t_e)$ represents the remaining generation window. This formulation ensures that the total number of generated firebrands remains consistent at $(t_g \times A_c \times G_t)$, independent of fluctuations in Δt during the simulation.

Mode C: HRR-Based Generation

Mode C normalizes the generation rate by fire power using a factor G (pcs/MW/s). The generation rate is defined as:

$$N = \text{HRR} \times G \times \min(\Delta t, t_g - t_e)$$

where HRR is the heat release rate of the source cell in MW. Empirical values for G are derived from literature, typically 33.3 pcs/MW/s for vegetative fuels and 10 pcs/MW/s for structural fuels. These values should be used with caution but can be modified by the user if site-specific data is available.

For vegetative fuels, the HRR is estimated via fireline intensity and cell size (HRR $= I_f \times c$). For structural fuels, WUI surface fire models utilize a prescribed design fire curve based on HRR per unit area (HRRPUA); in this case, the local HRR is calculated as HRRPUA $\times A_c$.

Computational Implementation

When using Mode B or C, the resulting firebrand count N is divided by an **ember sampling factor** (defaulting to 1) to manage computational load. Because the Lagrangian transport approach requires N to be an integer, non-integer values are processed using **stochastic rounding**. The value is rounded to the nearest higher or lower integer with a rounding probability equal to its decimal component. For example, if $N = 1.6$ pcs/Δt and the cell is expected to emit firebrands for 100 steps ($t_g = 100\Delta t$), we expect 60 steps to emit 2 firebrands and 40 steps to emit 1 firebrand. This approach ensures that the total number of generated firebrands converges to the expected value ($N \times t_g/\Delta t$) when $\Delta t \ll t_g$. Consequently, a sufficiently small time step is recommended to achieve convergence when running the Lagrangian transport scheme.

2.3.2 Transport

ELMFIRE provides two primary approaches for modeling firebrand transport: a Lagrangian scheme and an Eulerian scheme. In both approaches, firebrands are assigned travel distances based on prescribed probability distributions.

In the Lagrangian scheme, the spotting distance is explicitly sampled for each firebrand from a specified probabilistic distribution. In contrast, in the Eulerian scheme, the probabilistic distribution is interpreted as a deposition distribution over the landscape. In this context, the distribution represents the spatial number density of firebrands after all firebrands emitted from a source under given conditions have landed.

Three types of probabilistic (or density) distributions are available in ELMFIRE: (i) a user-defined uniform distribution, (ii) a user-defined lognormal distribution, and (iii) physics-based lognormal distributions parameterized following Sardoy et al. [24] and Himoto et al. [12].

All three distributions are available in the Lagrangian scheme, whereas only the physics-based lognormal distributions are used in the Eulerian scheme. The following sections describe the formulation and sampling procedures used in the Lagrangian scheme.

Uniform Distribution

For the uniform distribution, the spotting distance is sampled within a prescribed range defined by minimum and maximum values. A random number is drawn from a uniform distribution over this interval, and the spotting distance is assigned accordingly.

User-Defined Lognormal Distribution

The spotting distance X is assumed to follow a lognormal distribution:

$$pdf(X) = \frac{1}{X\,\sigma_x\sqrt{2\pi}} \exp\left[-\frac{(\ln X - \mu_x)^2}{2\sigma_x^2}\right].$$

The parameters μ_x and σ_x are related to the physical mean spotting distance (MSD) and its standard deviation (v) as:

$$\mu_x = \ln\left(\frac{\mathrm{MSD}^2}{\sqrt{\mathrm{MSD}\cdot v + \mathrm{MSD}^2}}\right),$$

$$\sigma_x = \sqrt{\ln\left(1 + \frac{\mathrm{MSD}\cdot v}{\mathrm{MSD}^2}\right)}.$$

The mean spotting distance is further parameterized as:

$$\mathrm{MSD} = \overline{x}\, I_b^a U_{20}^b.$$

Here, $\overline{x}$, v, a, and b are empirical parameters that control the shape of the distribution. By default, $a = 0.5$ and $b = 0.9$, while $\overline{x}$ and v are case-dependent.

In ELMFIRE, these parameters are specified in the `\&SPOTTING` namelist as follows: $\overline{x}$ corresponds to `MEAN_SPOTTING_DIST`, v to `NORMALIZED_SPOTTING_DIST_VARIANCE`, a to `SPOT_FLIN_EXP`, and b to `SPOT_WS_EXP`.

Sampling is performed using inverse transform sampling. A random number $R_0 \sim \mathrm{Unif}[0, 1]$ is generated, and the spotting distance is computed as:

$$X = \exp\left(\mu_x + \sqrt{2}\sigma_x \ \mathrm{erf}^{-1}(2R_0 - 1)\right).$$

The right-hand side of the equation corresponds to the inverse of the cumulative distribution function (CDF) associated with the given probability density function (PDF). In ELMFIRE, erf^{-1} is evaluated using a sixth-order polynomial approximation for small arguments and a Winitzki analytical approximation for large arguments, providing a balance between accuracy and computational efficiency.

Physics-Based Lognormal Distribution

For physics-based models, the downwind spotting distance X follows:

$$pdf(X) = \frac{1}{X\,\sigma_x\sqrt{2\pi}} \exp\left[-\frac{(\ln X - \mu_x)^2}{2\sigma_x^2}\right],$$

where μ_x and σ_x are determined from physical considerations.

Wildland Fuels

For wildland fuels, the parameters follow Sardoy et al. [24]. The Froude number is defined as:

$$Fr = \frac{U_{10}}{\sqrt{gL_c}}, \qquad L_c = \left(\frac{1000\,I_b}{\rho_\infty c_{p,g} T_\infty \sqrt{g}}\right)^{2/3}.$$

The lognormal parameters are:

For $Fr \leq 1$:

$$\mu_x = 1.47\,I_b^{0.54} U_{10}^{-0.55} + 1.14, \quad \sigma_x = 0.86\,I_b^{-0.21} U_{10}^{0.44} + 0.19.$$

For $Fr > 1$:

$$\mu_x = 1.32\,I_b^{0.26} U_{10}^{0.11} - 0.02, \quad \sigma_x = 4.95\,I_b^{-0.01} U_{10}^{-0.02} - 3.48.$$

Urban/Structural Fuels

For urban and structural fuels, the formulation of Himoto et al. [12] is used.

A characteristic length scale L_c represents the building size (default $L_c = 10$ m). The heat release rate is estimated as:

$$Q = I_b\,L_c \times 1000.$$

A dimensionless parameter B^* is defined as:

$$B^* = \frac{U_{10}}{\sqrt{gL_c}} \left(\frac{\rho_p}{\rho_\infty}\right)^{-3/4} \left(\frac{D_p}{L_c}\right)^{-3/4} \left(\frac{Q}{\rho_\infty c_{p,g} T_\infty \sqrt{g}\,L_c^{5/2}}\right)^{1/2}.$$

The mean and standard deviation in physical space are:

$$\mu_x^* = 0.47\,B^{*\,2/3}\,L_c, \quad \sigma_x^* = 0.88\,B^{*\,1/3}\,L_c.$$

These are converted to lognormal parameters:

$$\mu_x = \ln\left(\frac{\mu_x^*}{\sqrt{\left(\frac{\sigma_x^*}{\mu_x^*}\right)^2 + 1}}\right), \quad \sigma_x = \sqrt{\ln\left(1 + \left(\frac{\sigma_x^*}{\mu_x^*}\right)^2\right)}.$$

To ensure numerical robustness when using physics-based lognormal distributions, the probability density function (PDF) is truncated over the interval $[0,\, X_{1-P_{\mathrm{EPS}}}]$, corresponding to the $(1 - P_{\mathrm{EPS}})$ percentile:

$$X_M = \exp\left(\mu_x + \sqrt{2}\sigma_x \ \mathrm{erf}^{-1}(2P_{\mathrm{EPS}} - 1)\right).$$

The truncation limit is aligned with the grid spacing Δx:

$$X_M^* = \Delta x \left\lceil \frac{X_M}{\Delta x} \right\rceil.$$

The truncated PDF is subsequently renormalized to ensure that its integral over the truncated domain equals unity:

$$pdf^*(X) = \frac{pdf(X)}{\int_0^{X_M^*} pdf(X)\, dX}.$$

Sampling is performed by mapping a uniform random variable to the truncated CDF:

$$R_0^* = R_0(S_H - S_L) + S_L,$$

with

$$S_L = \tfrac{1}{2}\left[1 + \mathrm{erf}\left(\frac{\ln(10^{-6}) - \mu_x}{\sqrt{2}\sigma_x}\right)\right],$$

$$S_H = \tfrac{1}{2}\left[1 + \mathrm{erf}\left(\frac{\ln(X_M + 0.5\Delta x) - \mu_x}{\sqrt{2}\sigma_x}\right)\right].$$

The final sampled spotting distance is:

$$X = \exp\left(\mu_x + \sqrt{2}\sigma_x \ \mathrm{erf}^{-1}(2R_0^* - 1)\right).$$

The procedures described above provide the scalar distance that firebrands travel. The actual trajectory of the firebrands is then determined based on the input wind field. The treatment of particle accumulation and subsequent ignition during transport depends on whether a Lagrangian or Eulerian approach is used.

In the Lagrangian approach, firebrands are tracked individually and accumulate discretely at their landing locations. In the Eulerian approach, the probabilistic distribution is interpreted as a spatial deposition field, and firebrand number fluxes are distributed continuously along the transport trajectory. The details of each approach are described below.

Lagrangian Model

The Lagrangian model computes a unique, probabilistic landing location for each firebrand, or for each particle representing a group of firebrands defined by `EMBER_SAMPLING_FACTOR`, generated from a burning

cell. The downwind displacement is determined by the wind velocity field and the sampled spotting distance from the distributions described previously.

After the spotting distance is sampled, the particle trajectory is integrated using a first-order Forward Euler scheme. Firebrands are assumed to be massless and therefore travel at the local wind speed. The particle position is updated as:

$$\vec{X}_t = \vec{X}_{t-1} + \left(\vec{U}_{20} + \vec{\varepsilon}\right) \Delta t,$$

where $\vec{X}_t = (x, y)$ denotes the particle position at time t, $\vec{U}_{20} = (U_{x,20},\, U_{y,20})$ is the 20-ft wind velocity vector, and $U_{x,20}$ and $U_{y,20}$ are its components in the x and y directions, respectively. The term $\vec{\varepsilon}$ represents a stochastic perturbation to the wind direction.

The integration time step Δt is independent of the time step used in the level-set solver and is defined as:

$$\Delta t = \min\left(\frac{0.5\,\Delta x}{\max(\|\vec{U}_{20}\|,\, 0.01)},\, 5.0\right),$$

which corresponds to a Courant–Friedrichs–Lewy (CFL) condition of 0.5 based on the 20-ft wind speed. The magnitude of the two-dimensional wind velocity vector is given by:

$$\|\vec{U}_{20}\| = \sqrt{U_{x,20}^2 + U_{y,20}^2}.$$

The trajectory is integrated until the accumulated travel distance reaches or exceeds the sampled spotting distance.

The perturbation term $\vec{\varepsilon}$ introduces a randomized angular deviation to account for turbulent fluctuations in the ambient wind. A fluctuation is sampled uniformly between $-4°$ and $4°$ for each particle and is applied consistently throughout the trajectory integration.

For example, if the sampled fluctuation is $1°$, then at every time step (with interval Δt), the particle trajectory is deviated by $1°$ from the local wind direction. This constant angular offset is maintained over the entire trajectory, representing a persistent turbulent effect.

For the UMD spotting model, an additional crosswind dispersion is incorporated. The crosswind displacement is modeled independently using a normal distribution centered on the downwind trajectory:

$$pdf_y(Y) = \frac{1}{\sigma_y\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{Y-\mu_y}{\sigma_y}\right)^2\right],$$

$$\mu_y = 0, \qquad \sigma_y = 0.92\, L_c,$$

where L_c is the characteristic length scale defined previously.

This dispersion is applied after the particle has reached its downwind travel distance. For example, if a particle terminates at $(x_{\text{end}}, y_{\text{end}})$ based on the trajectory integration, it is subsequently displaced in the direction normal to the trajectory. An offset distance is sampled from the normal distribution, and the particle is reassigned to the corresponding grid cell at that offset location.

The sampling is again performed using the inverse transform sampling method. A random number R_0 is first drawn from the unit uniform distribution Unif$[0, 1]$. The crosswind displacement is then computed as:

$$Y = \sqrt{2}\,\sigma_y\,\mathrm{erf}^{-1}(2R_0 - 1) + \mu_y.$$

At the end of the transport process, the number of firebrands deposited in each grid cell is accumulated and subsequently used in the ignition step.

Eulerian Model

The Eulerian model treats firebrand transport as a continuous, spatially distributed process, rather than tracking individual particles. Instead of assigning a discrete landing location to each firebrand, the model computes the evolution of a firebrand number density field, representing the expected deposition of firebrands over the computational domain. This formulation is consistent with the physics-based Eulerian framework developed by Qin et al. [19].

In this approach, the downwind firebrand number distribution is described using the physics-based lognormal distribution introduced in the previous section, while the crosswind distribution is modeled independently using a normal distribution centered on the mean trajectory, consistent with the crosswind dispersion applied in the UMD spotting model in the Lagrangian scheme. The corresponding parameters μ_x, σ_x, and σ_y are determined from the fireline intensity and wind conditions, as described above.

Note that the crosswind distribution is truncated symmetrically such that the retained domain accounts for approximately 99% (by default) of the original probability density function (PDF). The truncation bounds are further adjusted to align with the nearest grid boundaries to ensure numerical robustness of the scheme. This procedure defines a one-sided maximum crosswind distance, Y_M^*, measured normal to the firebrand trajectory.

To update the firebrand number distribution, the maximum transport distances in the downwind and crosswind directions, denoted as X_M^* and Y_M^*, are first evaluated based on the truncated distributions. This treatment ensures that the required integrals are performed over a finite domain, thereby limiting the computational cost associated with firebrand transport while maintaining a negligible loss of the long-range tail of the distribution.

A tracer representing the transport path is then integrated over time using the same forward Euler scheme adopted in the Lagrangian model, with the total trajectory length equal to X_M^*. The time step is defined by a CFL condition:

$$\mathrm{CFL} = \frac{\|\vec{U}_{20}\|\,\Delta t}{\Delta x} = 1.$$

With this fixed time step, the tracer advances exactly one computational grid cell at each time step.

Based on this formulation, at the k-th time step following emission, the tracer originating from a source travels from X_{k-1} to X_k, corresponding to cell k. Firebrands are then distributed within this cell and across neighboring cells in the crosswind direction according to the prescribed probability distributions. The number of firebrands allocated is proportional to the joint probability of deposition over the corresponding spatial intervals.

Consider a cell l located in the crosswind direction, bounded by the interval $[Y_{k-1}, Y_k]$ relative to the trajectory. The probability of a firebrand landing in cell l is given by:

$$Pr_l = \left(\int_{X_{k-1}}^{X_k} pdf^*(x)\,\mathrm{d}x\right)\left(\int_{Y_{k-1}}^{Y_k} pdf_y^*(y)\,\mathrm{d}y\right).$$

Similarly, for the central cell k along the trajectory, the landing probability is:

$$Pr_k = \left(\int_{X_{k-1}}^{X_k} pdf^*(x)\,\mathrm{d}x \right) \left(\int_{-0.5\Delta x}^{0.5\Delta x} pdf_y^*(y)\,\mathrm{d}y \right).$$

Assuming that the emission source releases N firebrands during the level-set solver time interval Δt_l at time t_0, the number of firebrands allocated to cells k and l at time $(t_0 + k \times \Delta t)$ is given by $(N \times Pr_k)$ and $(N \times Pr_l)$, respectively. Note that Δt_l is distinct from the time step Δt used for tracer trajectory integration.

This deposition process is applied to all cells along the trajectory within the downwind range $[0,\, X_M^*]$, and across all crosswind cells within the range $[-Y_M^*,\, Y_M^*]$ normal to the trajectory, resulting in a spatial distribution of accumulated firebrands. The temporal evolution of this deposition field implicitly captures the arrival time distribution of firebrands, which is subsequently utilized in the ignition model.

2.3.3 Ignition

In ELMFIRE, ignition of target fuels by firebrands is modeled using two approaches: (1) a probabilistic method and (2) a physics-based method. The probabilistic approach is only compatible with the Lagrangian transport scheme, whereas the physics-based approach can be applied to both Lagrangian and Eulerian schemes, provided that the local evolution of the total firebrand number is properly accounted for.

Probabilitic ignition model

In the Lagrangian transport scheme, each deposited particle—either an individual firebrand or a particle representing a group of firebrands—is assigned an ignition probability P_i upon landing.

The ignition process is modeled using stochastic sampling. Specifically, when a particle reaches its destination, a random number R_0 is drawn from a unit uniform distribution, $R_0 \sim \mathrm{Unif}[0, 1]$, and compared with P_i. If $R_0 \leq P_i$, the target cell is considered ignited and the level-set variable is set to $\phi = -1$. Otherwise, the particle is discarded for ignition purposes, and subsequent ignition of the same cell depends on the arrival of other particles.

Under the default configuration, all deposited firebrands lead to ignition, corresponding to $P_i = 1$.

In addition, a final spatial consistency check is applied in the default ELMFIRE ignition model within the Lagrangian scheme. A 7×7 kernel centered at the landing location is evaluated. If any cell within this kernel is already part of the fire front (i.e., $\phi < 0$), the firebrand is considered sufficiently close to the existing fire, and no additional spot fire is initiated.

Physics-based ignition model

In ELMFIRE, a physics-based ignition model is available in addition to the probabilistic approach described above. Ignition of a target cell receiving firebrands is implemented by setting the level-set variable ϕ to -1 at time $t = \mathrm{TOA} + t_{\mathrm{ignition}}$, where TOA denotes the time of arrival of the first firebrand, and t_{ignition} is the time required for the fire to reach a sufficiently developed state.

The ignition process is modeled as two consecutive stages: (i) the formation of a local small flame (associated with smoldering-to-flaming transition), and (ii) the transition from a small flame to a fully developed fire. Accordingly, the total ignition time is defined as:

$$t_{\mathrm{ignition}} = t_{\mathrm{ign,small}} + t_{\mathrm{ign,large}}, \tag{2.14}$$

where $t_{\mathrm{ign,small}}$ represents the time required to establish a small flame, and $t_{\mathrm{ign,large}}$ represents the subsequent transition time to a fully developed fire.

Modeling of $t_{\text{ign,small}}$ In the present formulation, $t_{\text{ign,small}}$ is treated as a random variable determined from a probabilistic ignition model. Let $P_{\text{ign,small}}$ denote the probability that firebrands ignite a small flame on a flammable substrate within a specified duration $\tau_{\text{ign,small}}$:

$$P_{\text{ign,small}} = \text{Probability}(t_{\text{ign,small}} \leq \tau_{\text{ign,small}})\,. \tag{2.15}$$

Models for $P_{\text{ign,small}}$ and $\tau_{\text{ign,small}}$ are described in later sections. In the following, these quantities are assumed to be known.

Numerical Implementation The probabilistic ignition model defined in Eqn. 2.15 is implemented in a time-dependent framework as follows. Consider a target cell (TC) that begins receiving firebrands at level-set solver time step t_n. Once firebrands are detected in TC, ignition attempts are performed at each subsequent time step, with a success probability governed by $P_{\text{ign,small}}$, $\tau_{\text{ign,small}}$, and the elapsed time.

For clarity, assume that $P_{\text{ign,small}}$, $\tau_{\text{ign,small}}$, and the time step Δt_n are constant, and that $\tau_{\text{ign,small}} = N\,\Delta t_n$, where N is an integer. Over the duration $\tau_{\text{ign,small}}$, a total of N independent ignition attempts are performed.

The probability of no ignition over the interval $0 \leq t \leq \tau_{\text{ign,small}}$ is given by:

$$(1 - P_{\text{ign,small}}) = (1 - P_{\text{ign,small},\Delta t_n})^N, \tag{2.16}$$

where $P_{\text{ign,small},\Delta t_n}$ is the ignition probability over a single time step Δt_n. Solving for $P_{\text{ign,small},\Delta t_n}$ yields:

$$P_{\text{ign,small},\Delta t_n} = 1 - (1 - P_{\text{ign,small}})^{\Delta t_n / \tau_{\text{ign,small}}}. \tag{2.17}$$

Ignition Algorithm Equation 2.17 provides the basis for simulating the stochastic ignition process:

1. For each unignited cell containing firebrands at time t_n, compute $P_{\text{ign,small},\Delta t_n}$ using Eqn. 2.17.
2. Draw a random number $R_0 \sim \text{Unif}[0, 1]$. If $R_0 \leq P_{\text{ign,small},\Delta t_n}$, the cell is considered ignited and the ignition time is recorded as $t_{\text{ign,small}} = t_n - \text{TOA}$. Otherwise, proceed to the next time step.
3. Repeat until ignition occurs or the simulation terminates.

This algorithm ensures that the cumulative probability of ignition increases monotonically and reaches $P_{\text{ign,small}}$ after $\tau_{\text{ign,small}}$.

Vegetative fuels For vegetative fuels, ignition is assumed to occur with high likelihood once firebrands are deposited, even for relatively small firebrand loads. This assumption is consistent with porous fuels such as grass and shrubs under favorable conditions, and with the present modeling framework, which primarily considers short-range firebrand transport.

Accordingly, $P_{\text{ign,small}}$ is prescribed as 0.9, and $\tau_{\text{ign,small}} = 0$ by default.

Structural fuels For structural fuels, pyrolysis simulations [19] and experimental data [7] are used to construct a simplified ignition model. The formulation considers dry, unprotected materials (e.g., Pressure-Treated Wood, PTW) and relates ignition to the local firebrand mass loading ψ and the near-surface air velocity v_{air}.

The firebrand mass loading is computed as:

$$\psi(t) = \frac{N(t)\, m_{\text{firebrand}}}{A_c},$$

where $N(t)$ is the total number of firebrands within the cell at time t, A_c is the surface area of the cell, and $m_{\text{firebrand}}$ is the average mass of a single firebrand. In the current model, $m_{\text{firebrand}}$ is assumed to be 200 g. The resulting ψ is expressed in units of $\text{g} \cdot \text{cm}^{-2}$.

The near-surface velocity is estimated using Albini's logarithmic profile:

$$\frac{v_{\text{air}}}{U_{H+20}} = \frac{0.555}{\sqrt{fH}\ln\left(\frac{H+20-0.64H}{0.13H}\right)}, \tag{2.18}$$

where H is canopy height (ft) and f is canopy aspect ratio.

Ignition is predicted using:

$$\psi_{\text{crit}}(v_{\text{air}}) = \frac{C}{(v_{\text{air}} - v_{\min})(v_{\max} - v_{\text{air}})}, \tag{2.19}$$

with $v_{\min} = -0.073\ \text{m}\,\text{s}^{-1}$, $v_{\max} = 4.111\ \text{m}\,\text{s}^{-1}$, and $C = 0.211$.

The correlation is derived from a regression analysis of comprehensive pyrolysis simulations presented in Ref. [19]. Ignition is predicted to occur when $\psi \geq \psi_{\text{crit}}(v_{\text{air}})$.

Material variability, fuel moisture content, and protective layers are not explicitly considered.

To reflect near-binary ignition behavior, $P_{\text{ign,small}}$ is set to 0.9 when the criterion is satisfied and 0 otherwise.

Modeling of $t_{\text{ign,large}}$ The time required for a small flame to transition into a fully developed fire is highly uncertain. A simplified approach based on compartment fire theory is adopted.

The fire is assumed to grow from 10 W to 1 MW following:

$$HRR = \left(\frac{t}{t_{1\text{MW}}}\right)^2.$$

Thus,

$$t_{\text{ign,large}} = t_{1\text{MW}}.$$

Default values are:

$$t_{\text{ign,large}} = \begin{cases} 1000\ \text{s}, & \text{vegetative fuels}, \\ 300\ \text{s}, & \text{structural fuels}. \end{cases}$$

The structural value corresponds to a medium-growth fire; the vegetative value is an initial estimate subject to refinement.

2.3.4 Decay and Consumption

For completeness, the UMD spotting model also accounts for the consumption of active firebrands following deposition. This aspect is important because the current modeling framework has thus far considered only the accumulation of deposited firebrand particles, without accounting for their finite lifetimes. Although ignition events predicted by the model occur on characteristic timescales of several minutes, the accumulation of

sufficient firebrand mass to meet ignition thresholds may require tens of minutes. In contrast, the lifetime of a typical firebrand—measured from deposition on the substrate to complete consumption of its combustible mass—has been observed in wildland fire events to be on the order of a few minutes. Once consumed, a firebrand can no longer deliver heat to the target surface and therefore ceases to contribute to ignition.

To address this limitation, a first-order approximation of firebrand consumption after deposition is proposed in Qin's thesis [19]. In this formulation, deposited firebrands are categorized as *active* or *inactive*, with only active firebrands contributing to the heat flux imparted to the substrate.

Rather than directly tracking the decay in firebrand number, a more physically consistent approach is to model the evolution of firebrand mass. As a first-order approximation, the local mass of active firebrands is assumed to decay proportionally to its current value. Accordingly, the temporal evolution of the local firebrand mass load is expressed as:

$$\frac{\partial \psi}{\partial t} = -\frac{\psi}{t_{\text{fb,live}}}, \tag{2.20}$$

where ψ is the local firebrand mass per unit area (in $\text{g} \cdot \text{cm}^{-2}$, consistent with previous section), and $t_{\text{fb,live}}$ is the effective firebrand lifetime. Since ψ is the evolving quantity, an appropriate parameterization for $t_{\text{fb,live}}$ is required.

The physics-based ignition model introduces an empirical correlation for the incident heat flux generated by a pile of deposited firebrands, expressed as a function of the local firebrand mass loading ψ and the ambient air velocity v_{air}, following Ref. [7]. According to this model, the heat flux initially increases, reaches a peak, subsequently decreases, and eventually approaches a constant value. The temporal evolution of the heat flux is given by:

$$HF(t) = \begin{cases} HF_{\text{Rise}} \times (t + 12), & 0 \le t < t_{\text{rise}} \\ HF_{\text{Rise}} \times (t_{\text{rise}} + 12) - HF_{\text{Decay}} \times (t - t_{\text{rise}}), & t_{\text{rise}} \le t < t_{\text{rise}} + t_{\text{decay}} \\ HF_{\text{Final}}, & t \ge t_{\text{rise}} + t_{\text{decay}} \end{cases} \tag{2.21}$$

The individual components are defined as:

$$\begin{aligned} t_{\text{rise}} &= \max\left(-0.58\, v_{\text{air}}^2 - 0.50\, v_{\text{air}} + 43,\ 0\right), \\ HF_{\text{Rise}} &= \max\left[\left(-0.29\, v_{\text{air}}^2 + 1.49\, v_{\text{air}} + 0.1\right) \tanh(12\, \psi),\ 0\right], \\ HF_{\text{Decay}} &= \min\left[\left(-0.02\, v_{\text{air}}^2 - 0.13\, v_{\text{air}} - 0.01\right) \tanh(12\, \psi),\ 0\right], \\ HF_{\text{Final}} &= 8\, \tanh(12\, \psi). \end{aligned} \tag{2.22}$$

Here, v_{air} is the ambient air velocity (in $\text{m} \cdot \text{s}^{-1}$), and ψ is the firebrand mass loading, defined as the total mass of firebrand particles per unit area (in $\text{g} \cdot \text{cm}^{-2}$).

Because only firebrands capable of delivering effective heat release are considered active, a *critical incident heat flux*, $HF_{\text{fb,live}}$, is used to distinguish between active and inactive firebrands ($HF_{\text{fb,live}} = 10\ \text{kW} \cdot \text{m}^{-2}$ by default). Using this threshold, the effective heating duration of a firebrand pile can be inferred and interpreted as the firebrand lifetime.

Based on the empirical heat flux model, the firebrand lifetime is estimated as:

$$t_{\mathrm{fb,live}} = \min\left[(t_{\mathrm{decay}} + t_{\mathrm{rise}}), \frac{HF_{\mathrm{Rise}}\,(t_{\mathrm{rise}} + 12) - HF_{\mathrm{fb,live}}}{HF_{\mathrm{Decay}}} + t_{\mathrm{rise}}\right] - \left(\frac{HF_{\mathrm{fb,live}}}{HF_{\mathrm{Rise}}} - 12\right), \tag{2.23}$$

where the minimum function ensures a finite lifetime, given that the empirical heat flux profile approaches a nonzero asymptotic value, HF_{Final}.

Figure 2.2 illustrates the predicted firebrand lifetime $t_{\mathrm{fb,live}}$ as a function of ambient air velocity ($\mathrm{m \cdot s^{-1}}$) and local firebrand mass loading ψ ($\mathrm{g \cdot cm^{-2}}$). The results indicate that the firebrand lifetime initially increases and then decreases with increasing air velocity, consistent with the competing effects of enhanced convective heat transfer and accelerated burnout. In addition, the lifetime decreases with decreasing firebrand mass loading, as lower ψ values are insufficient to sustain the required heat flux over time.

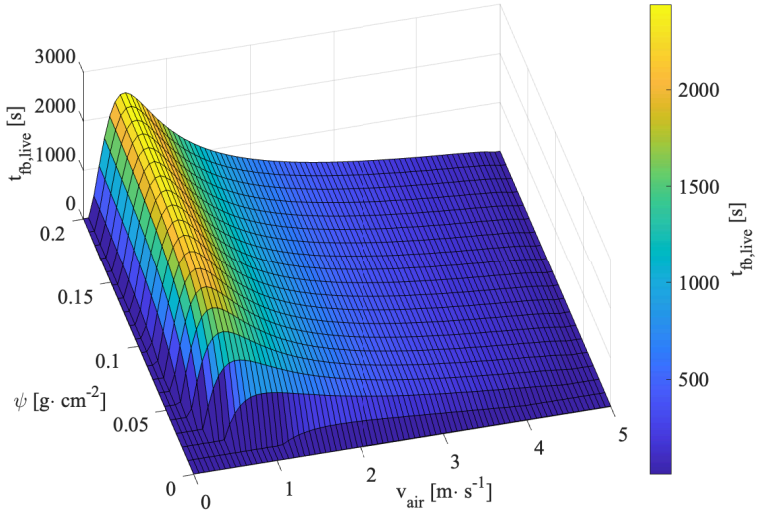


Figure 2.2: Predicted firebrand lifetime $t_{\mathrm{fb,live}}$ as a function of ambient air velocity and local firebrand mass load ψ. ($HF_{\mathrm{fb,live}} = 10\ \mathrm{kW \cdot m^{-2}}$ and a minimum $t_{\mathrm{fb,live}} = 10$ s are assumed) [19].

To incorporate this formulation into ELMFIRE, a first-order explicit time integration scheme is applied, in which the local firebrand mass is updated simultaneously with the level-set variable ϕ:

$$\psi(t + \Delta t_l) = \psi(t) - \frac{\psi(t)}{t_{\mathrm{fb,live}}(\psi(t),\, v_{\mathrm{air}}(t))}\,\Delta t_l. \tag{2.24}$$

Here, Δt_l denotes the time step used by the level-set equation solver.

2.4 Smoke

Estimating the smoke output of a specific wildfire scenario can be critical in assessing the health and visibility effects to nearby evacuees or to the wider populations through long term dispersion. To quantify the amount of smoke that people might be exposed to, a model is needed to estimate the dispersion of the wildfire pollutants (typically PM2.5) across the atmosphere. ELMFIRE does not contain such a model, but can

prepare inputs for smoke dispersion models such as HYSPLIT. Models such as HYSPLIT require inputs relating to the amount of pollutants released by the wildfire, and the power of the plume convective column (which they use to estimate the rise height of the smoke plume). While the plume power can easily be inferred from the total heat release rate of the fire, quantifying the emissions of a wildfire is more challenging.

The emissions of burning vegetation depend on a variety of factors, including the fuel loading, vegetation type, total consumption, fuel moisture content, and burning mode (flaming or smouldering). The information required for an accurate estimation of emissions is typically more than what fire behavior fuel types (Scott and Burgan, Andersen, NFDRS etc.) can provide. In the US, the FCCS fuel classification system was introduced to provide more information regarding the composition of a parcel of wildland [17]. FCCS can be procured through the LANDFIRE database, similarly to fuel models, canopy information and digitised terrain data. This data can then be used in combination with emission estimation models such as FOFEM and FuelFireTools (through the coupling of FCCS model, CONSUME model and FEPS model). Each of these models can return an estimation of emissions per fuel stratum, burning mode, and the duration of each phase.

Each of these models requires inputs from the user that would be difficult to obtain for theoretical fires. Parameters such as percent consumption of the canopy, percent sound and percent rotten wood are required by these models to provide accurate outputs. As such, and to ensure that ELMFIRE can be applied to global cases, a simplified method of estimating emissions has been implemented.

Emission factors are experimental constants that relate the emission of specific products to the total consumed biomass. They can be found in the literature, estimated under lab conditions or from real burns. The values from [22] are quoted below:

- Flaming PM2.5: 17.4 ± 7.2 g/kg
- Smouldering PM2.5: 49.8 ± 32.1 g/kg

Note the significant error margins, reflecting the heterogeneity of actual emission factors depending on the local conditions and the burning vegetation. Compare these values with another source, providing literature review values of emission factors for US forests [29]:

- Flaming PM2.5 (Northwest conifer forests): 23.2 ± 10.4 g/kg
- Flaming PM2.5 (Boreal forests): 21.5 ± 4.8 g/kg
- Smoldering PM2.5 (Stumps and Logs) 33 ± 20 g/kg
- Smoldering PM2.5 (Temperate forest duff) 50 ± 16 g/kg
- Smoldering PM2.5 (Boreal forest duff) 20.6 ± 20.6 g/kg

The values are similar but show the difference between vegetation type and overall prevailing conditions. For ELMFIRE, the values of [22] are used. The simplification is made that all fuel types (shrubs, grasses, timber litter etc.) have the same emission factors (and further down the section, calorific values). This simplification is maintained by anecdotal evidence that woody fuels produce the majority of PM2.5 emissions during a wildfire.

The emission factors can be converted from grams per kilogram of burned biomass to grams per kiloJoule of emitted heat through the calorific value of wood. The net calorific value of wood depends strongly on the moisture content, as any water evaporation that occurs may drop the overall net heat release. Using the oven-dry wood calorific value, the following equation can be used:

$$Q_M = Q_0(1 - M) - 2.44M$$

With Q_M being the net calorific value of wood with M% moisture content, and Q_0 being the oven dry

wood calorific value. According to literature, Q_0 is equal to 19 MJ/kg [8].

For a specific pixel burning in the landscape, the total energy release during the flaming phase is:

$$E = 0.06\ I_b\ C^2\ R^{-1}$$

With E being the total energy release (MJ), I_b being the fireline intensity (kW/m), C being the cell size (m), and R being the rate of spread (m/min).

Thus for the flaming phase, the total emissions in a cell is equal to:

$$PM_f = \frac{E\ \eta_f}{Q_M}$$

With η_f being the flaming emission factor.

For smoldering combustion the above equation cannot be used, as the time scale for smouldering cannot be determined from a fireline intensity and rate of spread value. Smoldering is the more important burning mode during a wildfire with respects to harmful PM2.5 emissions, due to its incomplete combustion processes [21]. The same source cites that more than 50% of a forest's biomass can be burned during smoldering, depending on the ratios of the surface, canopy and ground fuel loading. In ELMFIRE, a more average value of 30% is used, meaning that 70% of the total burned biomass in a wildfire emits flaming-mode emissions and 30% emits smodlering-mode emissions. Regarding the timespan of the smoldering phase of a wildfire, while no literature consensus exists, models like FOFEM predict values ranging from 30 to 60 minutes.

Using the above, the smouldering emissions can be estimated through:

$$PM_s = \frac{0.43E\ \eta_f}{Q_M}$$

Typically, smoke outputs are saved hourly. While the flaming phase lasts about 5 minutes within a 30 x 30 m cell, the smouldering phase can last up to 60 minutes. If the flaming and smouldering emissions are added instantaneously to the total wildfire emissions, any transient effects would be ignored and the overall emission output would be over- or underestimated. To account for this, a subgrid scale model can be used based on the work in [15]. There, the total emissions for every time step are adjusted depending on the relative fire progression within the cell. In ELMFIRE a similar subgrid scale model is introduced, made different because of the different way of estimating emissions. For simulation time T and cell ignition time T_i, the burning time of each cell is $T_c = T - T_i$. Then the relative total emissions of each cell can be linearly interpolated based on the total flaming time ($T_f = C/R$), and the total smoldering time ($T_s = 60$ min).

2.5 Suppression

The suppression module of ELMFIRE is experimental and still under development. It does not attempt to model individual firefighter actions at specific times or places, but instead represents the aggregate effects of suppression on wildfire containment and fire growth.

There are two parts to the suppression module of ELMFIRE. The first estimates the probability of success of an initial containment attempt based on fire size and fireline intensity. The second represents extended attack. Two alternative formulations are available for extended attack: an Area-Growth-Based Containment Model and a Spatially Explicit Suppression Model. The former estimates an aggregate target containment percentage, whereas the latter represents suppression directly on the evolving fire perimeter through direct

and indirect attack.

2.5.1 Initial Attack

The approach used here to quantify initial attack probability of containment is based on the analysis of Hirsch et al. [13], who leveraged expert judgment to quantify initial attack effectiveness as a function of fire size and head-fire fireline intensity, i.e., intensity at the main advancing fire front at the time of initial attack commencement. The authors developed an expression for probability of containment (POC) as a function of fire size (A) and fireline intensity (I_b):

$$POC = \frac{E}{1+E},$$

where

$$\ln E = 4.6835 - 0.7043A - 0.00041I_b - 0.000052\,AI_b.$$

In the equations above, A is in hectares and I_b is in kW/m. Since trends in probability of containment are not immediately apparent upon inspection of the equation, probability of containment calculated from the equation is tabulated in Fig. 2.3 as a function of fire size and head-fire fireline intensity at the time of initial attack. Although the qualitative trends are logical, i.e., containment probability increases with smaller fires, lower intensity, or both, the Hirsch et al. study was based on expert opinion from Canadian firefighters, so differences in suppression tactics between Canadian and U.S. agencies are not reflected in the formulation.

Fire size (hectacres)	1000	2000	3000	4000	5000	6000	7000	8000	9000	10000
10	77%	64%	49%	34%	22%	13%	8%	4%	2%	1%
9	82%	71%	58%	43%	29%	19%	11%	6%	4%	2%
8	86%	78%	66%	52%	38%	26%	16%	10%	6%	3%
7	89%	83%	73%	61%	48%	34%	23%	15%	9%	5%
6	92%	87%	80%	70%	57%	44%	31%	21%	14%	8%
5	94%	90%	85%	77%	66%	54%	41%	30%	20%	13%
4	95%	93%	89%	83%	74%	64%	52%	40%	29%	20%
3	97%	95%	92%	87%	81%	73%	63%	51%	39%	29%
2	97%	96%	94%	91%	86%	80%	72%	62%	51%	40%
1	98%	97%	96%	94%	90%	86%	80%	72%	63%	52%
	Fireline Intensity (kW/m)									

Figure 2.3: Probability of successful initial attack calculated using the probability-of-containment formulation.

2.5.2 Extended Attack

Extended attack in ELMFIRE can be represented using two alternative formulations. The Area-Growth-Based Containment Model estimates a target containment fraction from fire growth and suppression difficulty and subsequently distributes containment around the fire perimeter. The Spatially Explicit Suppression Model instead evaluates suppression directly on the evolving fireline and represents direct attack and indirect control-location holding dynamically during fire propagation.

Area-Growth-Based Containment Model

The aim of the Area-Growth-Based Containment Model is to estimate the target containment of the fire at each suppression time step and compare it with the estimated actual containment. It was developed based on a series of expert judgment exercises, with quantification of containment per day depending on the total containment capacity and daily progression of the fire.

At any given time step t during the level-set fire-spread calculation, the change in burning area can be calculated as

$$\frac{\delta A}{\delta t} = \frac{A_t - A_{t-1}}{\Delta t}.$$

For this equation and the remainder of this subsection, $\delta A/\delta t$ is expressed in acres per day.

A critical parameter of extended suppression is the Suppression Difficulty Index (SDI), a measure of the cumulative suppression difficulty of an area, including the influence of factors such as terrain and accessibility. Based on this value, total suppression difficulty can be represented through the SDI-informed total fire acreage, calculated as the sum of all burning points on the landscape multiplied by their corresponding SDI values:

$$A_{SDI} = \sum_i A_i SDI_i.$$

The change in cumulative suppression difficulty can then be calculated as

$$\frac{\delta A_{SDI}}{\delta t} = \frac{A_{SDI,t} - A_{SDI,t-1}}{\Delta t}.$$

The mean SDI at each time is subsequently calculated as

$$\overline{SDI}_t = \frac{\delta A_T/\delta T}{\delta A_{SDI,T}/\delta T} - 1.$$

The δT terms are expressed in acres per day. $\overline{SDI}_t$ is constrained to values between 0 and 3.

From this value, the change in target containment per day can be estimated based on expert judgment as

$$\delta C_{T,t} = 0.01 \psi C_m J \exp\left(-\left|B_{SDI}\overline{SDI}_t\right|\right),$$

where ψ is the diurnal adjustment factor, C_m is the maximum containment per day, B_{SDI} is a calibration constant, and J represents the influence of fire-area growth. The latter can be represented by either a linear or logarithmic relationship:

$$J = 1 - \frac{\delta A/\delta t}{A_{nc}},$$

or

$$J = 1 - \frac{\log(\delta A/\delta t)}{\log(A_{nc})},$$

where A_{nc} is the area-growth rate associated with no containment change per day. In broad terms, the change in containment is primarily controlled by the rate of fire-area increase relative to the specified area-growth threshold.

The total target containment at the current time step is then calculated as

$$C_{T,t} = C_{T,t-1} + \delta C_{T,t}.$$

The target containment is subsequently compared with the predicted actual containment of the fire. The analysis evaluates cumulative containment by dividing the fire front into angular bins relative to the centroid of the burning area.

For a given angular bin i, the velocities of all points within the bin form the set V_i, and the total number of cells within that bin is N_i. The average velocity is therefore

$$\overline{V}_i = \frac{\sum V_i}{N_i}.$$

The suppressed fraction s_i represents the fraction of previously suppressed points within angular bin i.

The current containment before newly burning points are considered is estimated as

$$C_{C,p} = \sum_i s_i \frac{N_i}{N},$$

where N is the total number of burning or contained points over all angular bins.

An intermediate step smooths the fire-front velocities by averaging the velocities of neighboring angular bins. The effect of newly burning points is then included until the current containment reaches the target containment:

$$C_C = C_{C,p} + \sum_i (1 - s_i) \frac{N_i}{N}.$$

Spatially Explicit Suppression Model

The Spatially Explicit Suppression Model represents suppression directly on the evolving fire perimeter. Rather than prescribing a target containment percentage for the entire fire, suppression develops through interactions between local fire behavior, suppression difficulty, available suppression capacity, and favorable control locations.

The model consists of two linked components. Direct attack represents suppression conducted at or immediately adjacent to the active fire edge, whereas indirect attack represents suppression opportunities at favorable control locations ahead of the advancing fire. The resulting suppressed portions of the fireline affect subsequent fire propagation.

Suppression Capacity and Deployment: Suppression capability is represented by the available suppression capacity, $C_s(t)$, expressed as the length of fireline that can be treated per unit time. The maximum available capacity is denoted by $C_{s,\max}$.

The model accounts for the delay associated with mobilization and deployment of suppression resources. A fixed deployment time, T_d, can first be prescribed. Under this formulation,

$$C_s(t) = \begin{cases} 0, & t < T_d, \\ C_{s,\max}, & t \geq T_d. \end{cases}$$

Thus, no extended-attack capacity is available before the prescribed deployment time, after which the full suppression capacity becomes available.

Alternatively, the deployment time can be estimated from the early development of the active fireline. Let $L(t)$ denote the active fireline length. The mean fireline-growth rate during the first day of the simulation is estimated as

$$\overline{G}_L = \frac{1}{N} \sum_{k=1}^{N} \frac{L(t_k) - L(t_{k-1})}{t_k - t_{k-1}},$$

where N is the number of fireline-growth intervals evaluated during the first day.

A reference fireline length, L_{ref}, represents the expected fireline size at which full suppression capacity is deployed. The corresponding deployment time is estimated as

$$T_d = \frac{L_{\text{ref}}}{\overline{G}_L}.$$

For this approach, suppression capacity increases progressively from zero toward the maximum available capacity during the deployment period. This relationship can be expressed generally as

$$C_s(t) = C_{s,\max} f_\kappa \left(\frac{t}{T_d} \right), \qquad 0 \le t < T_d,$$

where

$$f_\kappa(0) = 0, \qquad f_\kappa(1) = 1.$$

Once the deployment time is reached,

$$C_s(t) = C_{s,\max}, \qquad t \ge T_d.$$

The parameter κ controls the shape of the capacity-development curve. Larger values delay a greater fraction of the available capacity until later in the deployment period and produce a sharper increase as the full-capacity time is approached.

Direct Attack: Direct attack begins by identifying active fireline cells. An active fireline cell is a burning cell adjacent to one or more unburned cells and therefore represents a portion of the perimeter capable of continued propagation. Isolated fireline cells are excluded to reduce grid-scale artifacts.

Each active fireline cell is classified according to fireline type as heading, flanking, or backing fire. The classification follows the Huygens fire-spread representation. A local fire-spread ellipse is associated with each active fireline location, with its orientation and geometry determined by the combined effects of wind, slope, aspect, and rate of spread. The position of the fireline cell relative to the ellipse is then used to distinguish heading, flanking, and backing portions of the fire.

The active fireline is subsequently divided into spatially connected segments having similar fire behavior and suppression conditions. For neighboring cells j and k, similarity is evaluated using fireline type, rate of spread (ROS), flame length (FL), SDI, and PCL. The continuous-variable criteria can be written as

$$|ROS_j - ROS_k| \le \Delta ROS,$$

$$|FL_j - FL_k| \leq \Delta FL,$$

$$|SDI_j - SDI_k| \leq \Delta SDI,$$

and

$$|PCL_j - PCL_k| \leq \Delta PCL.$$

Very small segments are subsequently absorbed into neighboring segments to reduce fragmentation and avoid highly irregular suppression patterns.

For each resulting segment i, the mean flame length and SDI are

$$\overline{FL}_i = \frac{1}{N_i} \sum_{j=1}^{N_i} FL_{ij},$$

and

$$\overline{SDI}_i = \frac{1}{N_i} \sum_{j=1}^{N_i} SDI_{ij},$$

where N_i is the number of cells in segment i.

These quantities are normalized as

$$FL_{n,i} = \frac{\overline{FL}_i}{FL_{\max}},$$

and

$$SDI_{n,i} = \frac{\overline{SDI}_i}{SDI_{\max}},$$

where $FL_{\max}$ and $SDI_{\max}$ are reference values representing practical limits for direct attack.

The relative feasibility and difficulty of direct attack are represented using the Suppression Type Score (STS):

$$STS_i = 0.7FL_{n,i} + 0.3SDI_{n,i}.$$

The larger weighting assigned to flame length emphasizes the current fire behavior at the active perimeter, while the SDI term accounts for additional landscape-related suppression difficulty.

A fireline segment is considered eligible for direct attack when

$$STS_i < 1.$$

Segments satisfying

$$STS_i \geq 1$$

are not considered suitable for direct attack. Among eligible segments, STS_i also provides a relative measure of difficulty. Values approaching zero represent favorable suppression conditions, whereas values approaching one represent increasingly difficult conditions. Eligible segments are therefore prioritized from the lowest to the highest STS_i.

Suppression difficulty also modifies the amount of capacity required to treat a segment. Let L_i denote the physical length of segment i. The effective suppression requirement is calculated as

$$L_{\text{req},i} = \frac{L_i}{1 - STS_i}.$$

For relatively favorable conditions,

$$STS_i \to 0 \quad \Longrightarrow \quad L_{\text{req},i} \to L_i.$$

As suppression conditions become increasingly difficult,

$$STS_i \to 1 \quad \Longrightarrow \quad L_{\text{req},i} \to \infty.$$

The formulation therefore causes difficult segments to consume a larger portion of the available suppression capacity than equal-length segments under more favorable conditions.

For a suppression interval Δt_s, the total length that can be treated is

$$L_{\text{avail}} = C_s(t)\Delta t_s,$$

when time and suppression-capacity units are consistent. If suppression capacity is expressed in m/hr and the suppression interval is expressed in seconds, this becomes

$$L_{\text{avail}} = C_s(t)\frac{\Delta t_s}{3600}.$$

Suppression is progressively applied to eligible segments until the available capacity for the current suppression interval is exhausted.

Ranking segments only by STS_i could cause suppression to alternate between widely separated portions of the fire perimeter. The model therefore combines the STS_i ranking with a spatial search around the initial suppressed segment. After the first segment is selected, subsequent attack locations are identified through a 360° angular search around the existing suppression location, with preference given to nearby eligible segments having relatively low STS_i. This allows suppression to progress more continuously along the active fireline.

Indirect Attack: Indirect attack represents suppression at favorable control locations situated ahead of the active fire edge. Potential Control Location (PCL) information is used to identify areas where the advancing fire may be more likely to be stopped.

A cell j is considered a candidate control location when

$$PCL_j \geq PCL_{\text{th}},$$

where PCL_{th} is the minimum PCL value required for inclusion in the candidate control network.

Spatially connected candidate cells are grouped into control-location segments. The representative PCL value of segment i is calculated as the mean PCL of its constituent cells:

$$\overline{PCL}_i = \frac{1}{N_i} \sum_{j=1}^{N_i} PCL_{ij},$$

where N_i is the number of cells belonging to control-location segment i.

For PCL represented on a scale from 0 to 100, the normalized control-location suitability is

$$PCL_{n,i} = \frac{\overline{PCL}_i}{100}.$$

The probability that a control location successfully holds also depends on the fire behavior present when the advancing fire encounters that location. Flame length is used to represent this effect. The flame-length adjustment factor is

$$F_{FL} = \frac{1}{1 + \dfrac{FL}{FL_{\max}}},$$

where FL is the local flame length at the time of encounter and $FL_{\max}$ is the same flame-length reference used in the direct-attack formulation.

For low flame lengths,

$$F_{FL} \to 1,$$

whereas increasing flame length progressively reduces F_{FL}. The formulation therefore decreases the effectiveness of a potential control location as local fire behavior becomes more severe.

The probability that control-location segment i successfully holds is

$$P_{\text{hold},i} = PCL_{n,i} F_{FL}.$$

The corresponding probability that the control location is breached is

$$P_{\text{breach},i} = 1 - P_{\text{hold},i}.$$

When the advancing fire reaches a candidate control location, the outcome is determined stochastically. A random value R is drawn from a uniform distribution,

$$R \sim U(0, 1).$$

The control location successfully holds when

$$R \leq P_{\text{hold},i}.$$

In this case, propagation through the affected portion of the control location is prevented. Conversely, if

$$R > P_{\text{hold},i},$$

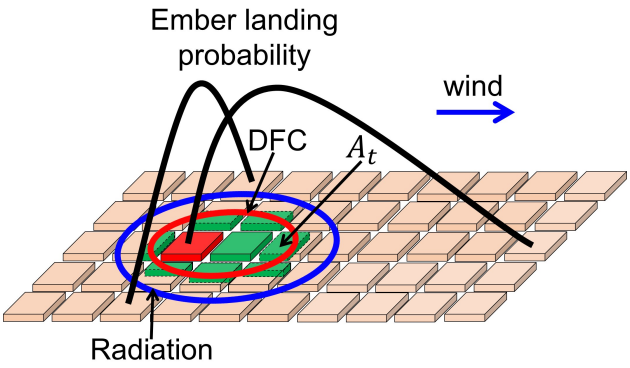


Figure 2.4: A schematic of the WU-E heat spread mechanisms

the control location is considered breached and the fire continues to propagate according to the underlying ELMFIRE fire-spread calculation.

2.6 Urban Fire Spread

The urban fire spread model adapted in ELMFIRE is WU-E, taken from Purnomo et al [18], and describes a semi-physical model to estimate the spread of fire though a built-up area, which up to now was considered nonburnable by Rothermel-based fire models.

The heat transfer from a fire front to a nearby building is broken down into direct flame contact (DFC), radiation and firebrands. Each of the three heat transfer mechanisms is assumed independent, although areas with direct flame contact do not calculate radiation as well. The heat transfer mechanisms considered here are shown in Fig. 2.4.

The model operates by simulating multiple modes of fire spread. Heat transfer to cells via DFC is determined by the heat release rate (HRR) and the relative position of the target cell to the burning structure. Currently, this component of the model is based on regression data from the older Hamada model [11], as fine-scale data for structure-to-structure heat fluxes remains limited. For direct flame contact, the Hamada model [11] is used to estimate its range of effect. Due to the rarity of digital copies of this work, the Hamada model as explained by Qin [19] has been repeated below.

For radiative heat transfer, the model employs a point-source ignition approach to account for radiation effects. Consequently, fire spread via radiation is also influenced by HRR. Structures ignite when they are exposed to a critical heat flux for a specified duration, defined as a flux-time product, which represents the ignition characteristics of the material.

2.6.1 The Hamada Model

The Hamada model was developed for a point source building ignition, in a uniform community. It calculates the fire spreads based on an elliptical assumption, explicitly calculating the spread of the fire downwind (K_d), upwind (K_u), and crosswind (K_s). These parameters are enough to get a Length over Width (LoW) measure, that can be directly applied using the elliptical fire spread equations we have already established.

The original elliptical dimensions are:

$$K_d = \frac{(a_0 + d)\, t}{T_d},$$

$$K_s = \frac{a_0}{2} + d + \frac{(a_0 + d)(t - T_s)}{T_s},$$

$$K_u = \frac{a_0}{2} + d + \frac{(a_0 + d)(t - T_u)}{T_u}.$$

With a_0 denoting average building footprint dimension, d is average building separation, and t is the elapsed time. A later correction by Hazus was introduced to correct for an overestimation of fire spread under zero wind:

$$K'_d = K_d \frac{V}{10} + \left(1 - \frac{V}{10}\right) \sqrt{\left(\frac{K_d + K_u}{2}\right) K_s},$$

$$K'_s = K_s \frac{V}{10} + \left(1 - \frac{V}{10}\right) \sqrt{\left(\frac{K_d + K_u}{2}\right) K_s},$$

$$K'_u = K_u \frac{V}{10} + \left(1 - \frac{V}{10}\right) \sqrt{\left(\frac{K_d + K_u}{2}\right) K_s}.$$

Where V is the wind speed. The above equations can be rearranged to solve for the time taken for the fire to reach adjacent buildings.

$$T_i = (1 - f_b) \left(3 + 0.375\, a_0 + \frac{8d}{c_{4i} + c_{5i} V}\right) + \frac{f_b}{C(V)} \left(5 + 0.625\, a_0 + \frac{16d}{c_{4i} + c_{5i} V}\right),$$

$$C(V) = c_{1i}\big(1 + c_{2i} V + c_{4i} V^2\big).$$

where i indexes wind directions (downwind, crosswind, upwind), and f_b is the combustible fraction. Empirical constants c_{ki} are tabulated below:

i	c_{1i}	c_{2i}	c_{3i}	c_{4i}	c_{5i}
d	1.6	0.1	0.007	25.0	2.5
s	1.0	0.0	0.005	5.0	0.25
u	1.0	0.0	0.002	5.0	0.2

Then, the spread extent and spread time can be used to find the rate of spread of the fire towards each direction:

$$V_i = \frac{dK'_i}{dt} = \frac{V}{10} \frac{dK_i}{dt} + \frac{1}{4} \left(1 - \frac{V}{10}\right) \mathcal{A}, \qquad i \in \{d, s, u\}.$$

$$\mathcal{A} := \sqrt{\frac{2}{(K_d + K_u)\, K_s}} \left[\frac{dK_s}{dt}(K_d + K_u) + K_s \left(\frac{dK_d}{dt} + \frac{dK_u}{dt}\right)\right].$$

With the temporal derivatives:

$$\frac{dK_i}{dt} = \frac{a_0 + d}{T_i}$$

The results can then be used to calculate the LoB of the urban fire spread ellipse:

$$LB = \frac{V_u + V_d}{V_s}$$

2.6.2 Heat Transfer

DFC, originating from a burning house (red square), has limited influence, depicted by the red ellipse. Beyond this ellipse, DFC is negligible. Radiation effects are confined to a 100-m radius (blue ellipse) and are zero beyond this distance. When a cell receives DFC, only heat from DFC is considered, disabling radiation. For cells with both DFC and radiation, heat contributions are shown by green (DFC) and brown (radiation) areas. Firebrand propagation follows probabilistic distributions.

The heat transferred to cells via DFC is determined by the heat release rate (HRR) and the relative position of the cell to the burning house. This relationship is formulated as:

$$\dot{q}_c = \frac{HRR \cdot A_t}{A_{f,eff}}$$

Here, $\dot{q}_c(kW)$ is the heat from DFC, HRR (kW) is the heat release rate of a source cell, $A_t(m^2)$ is the area of a target cell affected by the flame, and $A_{f,eff}(m^2)$ is the effective flame area. The flame size is determined by regression based on Hamada's empirical model, considering wind speed, house size, and separation distance.

The HRR starts at zero upon ignition, grows linearly to peak HRR, stays constant, and then decays as fuel is consumed. Peak HRR per unit area of structure (HRRPUA) is 150 kW/m^2, calculated by dividing the reported HRR by the structure area. Stages include an early stage (5 min), a fully developed stage (1 min), and a decay stage (60 min).

The model adopts a point-source ignition method for radiation effects. The spread of fire through radiation depends on HRR and the distance of the target cell from the burning house. The incident heat flux resulting from radiation is calculated as:

$$\dot{q}''_r = \frac{0.3 \cdot HRR}{4\pi R^2}$$

Fire spread from urban cells to neighboring cells depends on heat transfer via DFC, radiation, or firebrands. Heat received by an unburned cell is given by:

$$\dot{q}_t = \alpha_c \dot{q}_c + \alpha_r \dot{q}''_r \Delta x^2$$

where α_c and α_r are fire transfer coefficients for DFC and radiation, and Δx^2 is the area of the burning house. The DFC coefficient α_c depends on house material and configuration, while the radiative coefficient α_r is a product of α_c and house radiation absorptivity. The DFC coefficient α_c is derived from:

$$\alpha_c = \frac{S_b + S_v}{S_t}$$

where S_b is the combustible material surface area, S_v is the vegetation area outside the house within the same cell, and S_t is the total surface area.

A structure ignites when subjected to a critical heat flux for a duration defined by the critical flux time product (FTP). The FTP value (kJ) is based on incident heat flux, adapted from Lee (2009). The model can account for different FTP values for various materials, though a uniform threshold is used here. The

model's flexibility allows easy integration of future improvements based on new research.

For urban fire spread rate, U represents the time Δt required to ignite a structure of size Δx, as shown by:

$\sum (\dot{q}_t \Delta t) \geq FTP$

$U = \frac{\sum \dot{q}_t}{FTP} \cdot \Delta x$

It can be expressed as a vector by applying decomposition through the elliptical point spread hypothesis:

$$a = \frac{K_d + K_u}{2},$$

$$c = \min\left(\frac{a}{2},\, a - K_u\right),$$

$$b = \frac{K_s}{\sqrt{1 - \left(\frac{c}{a}\right)^2}}.$$

And the directional rates of spread can be calculated downwind (V_d), upwind (V_u), and sidewind (V_S) of the source:

$$V_d = U \cdot \frac{(a + c)\,\Delta x}{2(a + b)},$$

$$V_s = U \cdot \frac{2b\,\Delta x}{2(a + b)},$$

$$V_u = U \cdot \frac{(a - c)\,\Delta x}{2(a + b)}.$$

The model also adapts for fire spread from vegetation into urban areas by converting the wildland fireline intensity output to HRR for modeling urban fire spread. However, a simplified version of this interface spread is introduced. This simplified version leverages distance from burning vegetation and fireline intensity threshold to determine ignition of structures caused by burning vegetation. The default values for these thresholds are now arbitrary selected as 1 cell for distance and 1000 kW/m for fireline intensity. These two interface spread options can be selected depending on the users and purposes.

Similarly, when fire spreads from urban to wildland areas, wildland cells at the urban interface receive heat from adjacent urban cells and ignite accordingly, using the same equations as for urban cells.

$HRR = I_f \Delta x$

A number of critical assumptions need to be taken into account when using this model.

- The heat received by the target is the heat flux times the area affected (At). It is assumed that the heat flux is uniform over the flame ellipse. Thus, the heat flux is HRR/area. It is assumed that the heat flux when the heat flux gauge is placed on the wall of the burning house is equal to the heat flux gauge reading when place further away from the burning house as long as the flame touches the heat flux gauge.
- Regression is performed on Hamada equations, adapting the method in Jiang, so that WU-E does not execute Hamada equation, which can be more computationally demanding, but instead calculates the more simple regression equation. From these calculations, flame reach at the downwind, upwind and sidewind directions are obtained. These flame reaches cannot be directly used to get the radius of

ellipse at a specific angle, that follow Kepler law of elliptical orbit.

- The ellipse area used for the calculation of DFC is based on effective ellipse area. Although the real flame ellipse might be larger, the heat dissipates more rapidly further from the source. By focusing on a smaller effective ellipse, where the majority of the heat is actually distributed, we assume uniform heat within this area and negligible heat outside it (but still within the real ellipse). This approach prioritizes the area closer to the source, effectively sacrificing the outer regions, formulated as:

$$\dot{q}_c = \frac{HRR \cdot A_t}{\pi \cdot a_{\text{eff}} \cdot b_{\text{eff}}}$$

  To accommodate this, a tunable parameter (`HRR_ELLIPSE_ADJ`) was introduced in the SIMULATOR namelist. This parameter is multiplied with the actual semi-major and semi-minor axes of the ellipse to obtain their effective values used in the calculation. The default value of `HRR_ELLIPSE_ADJ` is set to 0.5, but it is subject to calibration for improved accuracy.
- C%ABSOLUTE_U corresponds to the total heat available to form the flame ellipse. It is assumed that by determining the portion of this heat goes to downwind (C%VELOCITY_DMS), upwind (C%VBACK), and sidewind (V_S), ellipse shape can be formed. Thus, C%ABSOLUTE_U corresponds to the total of these portions (sum of downwind, upwind, and sidewind directions). The fraction of C%ABSOLUTE_U goes to downwind (C%VELOCITY_DMS), upwind (C%VBACK), and sidewind (V_S) are based the ratio of each length with the total length.

2.7 Diurnal Profiles

The diurnal factors, aiming at adjusting for the Rothermel model's known overestimation of night-time rate of spread, require an estimation of the sunrise and sunset times of the area of interest. The calculations used in ELMFIRE come from the NOAAs sunrise/sunset calculations, and will be repeated here:

The fractional year γ, in radians, is computed as:

$$\gamma = \frac{2\pi}{365}\left(\text{day\_of\_year} - 1 + \frac{\text{hour} - 12}{24}\right)$$

For leap years, replace 365 with 366.

Using γ, we compute the equation of time (in minutes):

$$\text{eqtime} = 229.18\,(0.000075 + 0.001868\cos\gamma - 0.032077\sin\gamma - 0.014615\cos(2\gamma) - 0.040849\sin(2\gamma))$$

The solar declination angle (in radians) is:

$$\text{decl} = 0.006918 - 0.399912\cos\gamma + 0.070257\sin\gamma - 0.006758\cos(2\gamma) + 0.000907\sin(2\gamma) - 0.002697\cos(3\gamma) + 0.00148\sin(3\gamma)$$

First compute the total time offset (minutes):

$$\text{time\_offset} = \text{eqtime} + 4(\text{longitude}) - 60(\text{timezone})$$

Longitude is in degrees, positive east of the prime meridian. Timezone is hours from UTC (e.g., U.S. Mountain Standard Time = -7).

Then the true solar time (minutes) is:

$$\text{tst} = \text{hr} \cdot 60 + \text{mn} + \frac{\text{sc}}{60} + \text{time\_offset}$$

The solar hour angle (degrees) is:

$$\text{ha} = \frac{\text{tst}}{4} - 180$$

The solar zenith angle ϕ is given by:

$$\cos\phi = \sin(\text{lat})\sin(\text{decl}) + \cos(\text{lat})\cos(\text{decl})\cos(\text{ha})$$

The solar azimuth angle θ (degrees clockwise from north) satisfies:

$$\cos(180 - \theta) = -\frac{\sin(\text{lat})\cos\phi - \sin(\text{decl})}{\cos(\text{lat})\sin\phi}$$

For sunrise or sunset, the zenith angle is set to:

$$z_0 = 90.833^\circ$$

accounting for atmospheric refraction and solar disk radius.

The corresponding hour angle is:

$$\text{ha} = \pm\arccos\left\{\frac{\cos z_0}{\cos(\text{lat})\cos(\text{decl})} - \tan(\text{lat})\tan(\text{decl})\right\}$$

The + sign corresponds to sunrise, − to sunset.

$$\text{sunrise} = 720 - 4(\text{longitude} + \text{ha}) - \text{eqtime}$$

(all angles in degrees, eqtime in minutes).

Solar noon occurs at:

$$\text{snoon} = 720 - 4(\text{longitude}) - \text{eqtime}$$

Chapter 3

Verification

3.1 Introduction

In the context of natural phenomena modelling, verification refers to the process of confirming that the final program's output mathematically corresponds to the analytically derived, expected outcome. Validation refers to the process of confirming that the model can, to some degree of accuracy, model the real phenomena it is meant to predict. In simplified terms, verification tests simplified cases that can be modelled both computationally and analytically, and compares the results. Validation tests real world, complex cases that can only be studied computationally and historically. These two processes give greater confidence (but not guarantee) that the model is performing as it is expected, and can reliably predict future events.

ASTM E1355 Standard Guide for Evaluating the Predictive Capability of Deterministic Fire Models (2018) defines model verification as "the process of determining that the implementation of a calculation method accurately represents the developer's conceptual description of the calculation method and solution to the calculation method." Slightly different definitions of model verification are used by other standards in other industries, but in general verification involves assessing whether the underlying mathematical formulation is correctly implemented in the source code - or "is the math right". ASTM E1355 Standard Guide for Evaluating the Predictive Capability of Deterministic Fire Models (2018) defines model validation as "the process of determining the degree to which a calculation method is an accurate representation of the real-world from the perspective of the intended uses of the calculation method."

Verification and validation of wildfire spread models has its own intricacies and particularities, as the range of methodologies that can be employed to model wildfire spread are broad and varied. Sullivan categorises the plethora of models in the literature as spanning a spectrum between empirical (purely relying on experimental correlations and interpolations) and physical (purely relying on conservation laws and heat and mass transfer laws). Most wildfire models are somewhere inbetween, with some degree of physical reasoning and calibration required by all models. ELMFIRE would be classified as a semi-empirical model, as it primarily uses physically-derived equation with empirical, experiment-driven parameterization.

Model verification is typically approached by simulating simple canonical problems and comparing model outputs to exact solutions, where available. ELMFIRE has been subjected to extensive model verification exercises since development began in 2010. In an effort to promote transparency and authenticity, ELMFIRE's mathematical formulation is documented in the Technical Reference, its source code is maintained in a public Git repository, and several model verification test cases.

This guide will present the results of a verification and validation study of ELMFIRE. A series of simplified cases will first be presented, with quantitative and qualitative comparison of the computational and analytical results. Following that, a series of historical wildfires will be modelled in ELMFIRE. The results will be compared to establish ELMFIRE's ability to work on real, complex landscapes.

3.2 Verification Test Cases

The verification of ELMFIRE will be based on the WUI-NITY project verification guide (WUI-NITY 2), where a wildfire spread model verification method is described, and later applied in a benchmarking study. The verification test suggests small landscapes with five simplified scenarios:

- **"Point"**: Uniform fuel, flat landscape, zero wind, ignition at the center of the landscape.
- **"Windy"**: Uniform fuel, flat landscape, northern wind, ignition at the north part of the landscape.
- **"Valley"**: Uniform fuel, a shallow landscape rise to the east and a steep rise to the west, zero wind, ignition at the center of the landscape.
- **"Fuel Quadrant"**: Different fuels at each quadrant of the landscape, flat landscape, zero wind, ignition at the center of the landscape.
- **"Complex"**: (an amalgamation of the previous cases) Different fuels at each quadrant of the landscape, a shallow landscape rise to the east and a steep rise to the west, northern wind, ignition at the center.

Each of these tests verifies the model against a separate parameter, with the final one superimposing the previous cases to test their interaction. For this verification guide, additional tests are proposed to expand the parameters being examined and verified. These extra tests follow from the submodel breakdown identified in the main guide.

- **"Moisture Quadrant"**: Uniform fuel, with different fuel moisture values at each landscape quadrant.
- **"Canopy"**: Same as "Windy" case, with canopy values included.
- **"Firebrands"**: Uniform fuel, flat landscape, northern wind, line ignition at the north, with firebrand generation included.
- **"Overnight"**: Same as "Point" with an overnight burn duration.
- **"Suppression"**: Same as "Point" with suppression to the south of the ignition.

The expected outcomes of each case are shown in Fig. 3.1. The original five cases are colored orange, and the four new cases are colored blue. Some of the original figures have been altered compared to the original source. Based on these cases, appropriate weather, topography and fuel files were created to run each simulation.

The details of each case are described in Tab. 3.1. All the raster inputs are 126 x 126 cells with 30m cell size. Wind magnitude and direction is spatially constant. Simulation DT is 1 minute. Wind is specified at 10m height. 10-hour and 100-hour fuel moistures are 7% and 8% for all fuels and cases, live herbaceous fuel moisture is set to 90% and live woody fuel moisture is set to 60%. Where analytical solutions for the rate of spread can be acquired, they are calculated through the Rothermel model and the BEHAVE program. All isochrones shown in the later sections are hourly.

To maintain accuracy between versions, and ensure that subsequent changes do not affect the verification results, we have included scripts to automatically run and verify all cases below. The `verification/GUIDE/run_verification.sh` script runs all cases and compares them with the analytically derived results, highlighting discrepancies and similarities. The output of this script is a table summarising the critical values of each case.

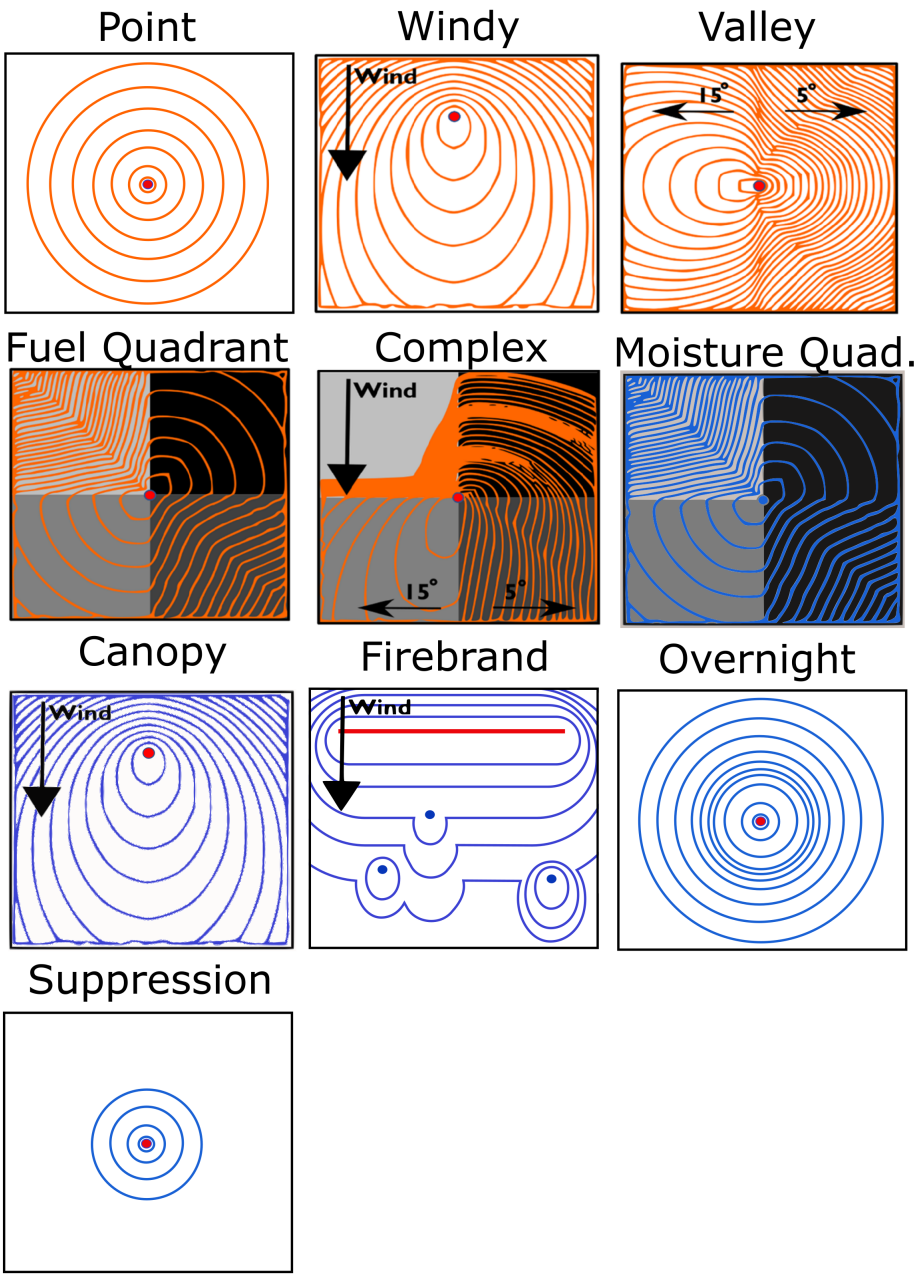


Figure 3.1: The nine verification cases of this study. The orange fire fronts represent the original five cases, and the blue fire fronts represent the new cases. Adapted from WUI-NITY 2.

Case	Metric	Target	ELMFIRE	%err	Result
Point	head ROS	1.510	1.512	0.1	PASS
Windy	head ROS	3.635	3.644	0.3	PASS
Windy	L/W ratio	1.240	1.248	0.6	PASS
Valley	head ROS (5 deg, east)	1.920	1.924	0.2	PASS
Valley	head ROS (15 deg, west)	5.370	5.374	0.1	PASS

Quadrant	FB8 head ROS	0.080	0.080	-0.6	PASS
Quadrant	FB7 head ROS	0.472	0.471	-0.3	PASS
Quadrant	FB4 head ROS	1.492	1.493	0.1	PASS
Quadrant	FB2 head ROS	0.871	0.872	0.1	PASS
MoistureQuad	ROS @ 6%	1.511	1.512	0.1	PASS
MoistureQuad	ROS @ 12%	1.089	1.090	0.1	PASS
MoistureQuad	ROS @ 18%	0.801	0.801	0.0	PASS
MoistureQuad	ROS @ 25%	0.000	0.000	-	PASS
Canopy-1mph	crown class	0.000	0.000	-	PASS
Canopy-1mph	max ROS	1.809	1.745	-3.5	PASS
Canopy-5mph	crown class	1.000	1.000	0.0	PASS
Canopy-5mph	max ROS	10.060	10.084	0.2	PASS
Canopy-10mph	crown class	2.000	2.000	0.0	PASS
Canopy-10mph	max ROS	35.300	35.443	0.4	PASS
Complex	max ROS (qualitative)	-	12.896	-	n/a
Firebrands	firebrands (cell)	8450.000	12198.000	44.4	FAIL
Firebrands	ln-distance mu	2.420	2.411	-0.4	PASS
Firebrands	ln-distance sigma	1.310	1.300	-0.7	PASS
Overnight	reduction starts (hr)	17.500	17.750	1.4	PASS
Overnight	reduction ends (hr)	7.500	7.750	3.3	PASS
Suppression-extendedtime at simulation end		88200.000	83988.000	-4.8	PASS
Suppression-initial fraction contained		0.540	0.569	5.4	PASS

Table 3.1: Simulation parameters of each verification case. Changes in fuel types between cases occur to generate faster or slower wildfires, for better visualisation or comparison.

Case	**Topography**	**Fuel (FBFM13)**	**CC (%)**	**CBH (m×10)**	**CBD (kg/m³×100)**	**CH (m×10)**	**Wind (mph)**	**1h FMC**	**Firebrands**	**Diurnal**	**Suppression**
Point	Flat	3	0	0	0	0	0.0	6	off	off	off
Windy	Flat	5	0	0	0	0	5	6	off	off	off
Valley	5° east, 15° west	3	0	0	0	0	0.0	6	off	off	off
Fuel Quadrant	Flat	8, 7, 4, 2	0	0	0	0	0.0	6	off	off	off
Complex	5° east, 15° west	8, 7, 4, 2	0	0	0	0	5	6	off	off	off
Moisture Quad.	Flat	3	0	0	0	0	0.0	6, 12, 18, 25	off	off	off
Canopy	Flat	5	80	16	56	55	1, 6, 12	6	off	off	off
Firebrand	Flat	3	0	0	0	0	5	6	on	off	off
Overnight	Flat	3	0	0	0	0	0.0	6	off	72 h runtime	off
Suppression	Flat	3	0	0	0	0	0.0	6	off	off	on

3.2.1 Point

Goal. Verify baseline isotropic spread in the absence of wind and slope.

Setup. Flat terrain and no wind. BEHAVE (Rothermel surface model) inputs:

- Fuel Model: FB3 (3) Tall Grass (static)
- 1-h fuel moisture: 6%
- 20-ft wind speed: 0 mph
- Slope steepness: 0%

Expected outcome. Circular isochrones with constant rate of spread.

Result. Figure 3.2 shows circular fire fronts. The estimated rate of spread is 1.51 m/min, matching BEHAVE.

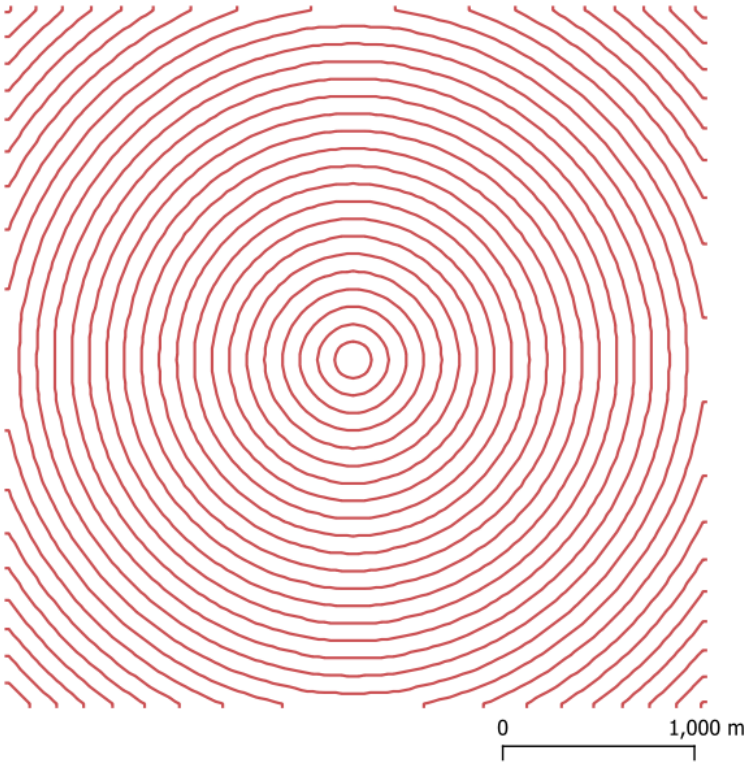


Figure 3.2: ELMFIRE simulation of verification case "Point"

3.2.2 Windy

Goal. Verify wind-driven elliptical spread and agreement with an analytical/BEHAVE-based isochrone.

Method. Compute the wind adjustment factor (WAF) to convert 20-ft wind to midflame wind, then compare BEHAVE-derived spread metrics and analytical isochrones against ELMFIRE.

Wind adjustment factor. For uncovered fuels (no or sparse canopy):

$$WAF_u = \frac{1.83}{\ln\left(\frac{20+0.36H}{0.13H}\right)} \tag{3.1}$$

Where H is the fuel depth height in feet. If canopy is present, the sheltered WAF is used. All WAF values are constrained between 0.1 and 1.

$$WAF_s = \frac{0.555}{\sqrt{\frac{H(CH-CBH)}{CH}}\ln\left(\frac{20+0.36H}{0.13H}\right)} \tag{3.2}$$

Recall that the point fire spread can be characterised by an ellipse whose Length over Width ratio can be found by:

$$\frac{L}{W} = \min\left[0.936\,e^{0.1147\,U_{mf,e}} + 0.461\,e^{-0.0692\,U_{mf,e}} - 0.397,\ 8\right] \tag{3.3}$$

Setup. Uniform topography and constant wind. BEHAVE inputs:

- Fuel Model: FB5 (5) Brush (static)
- 1-h fuel moisture: 6%
- 10-h fuel moisture: 7%
- 20ft wind speed: 6 mph
- Slope steepness: 0%

Expected (BEHAVE/analytical).

- Wind adjustment factor: 0.42
- Head rate of spread: 3.66 m/min
- Midflame windspeed: 2.5 km/h
- L/W: 1.24

Result (ELMFIRE).

- Head rate of spread: 3.644 m/min
- L/W: 1.248

Comparison. The match is almost identical, supporting correct implementation of wind-driven elliptical spread.

3.2.3 Valley

Goal. Verify slope-driven spread and the Huygens construction (successive isochrones consistent with point-source expansions).

Setup. Two slopes (5% and 15%), no wind. BEHAVE inputs:

- Fuel Model: FB3 (3) Tall Grass (static)
- 1-h fuel moisture: 6%
- 20ft wind speed: 0 mph
- Slope steepness: 5% / 15%

Expected (BEHAVE).

- Head rate of spread (5%): 1.92 m/min
- L/W (5%): 1.01
- Head rate of spread (15%): 5.37 m/min
- L/W (15%): 1.07

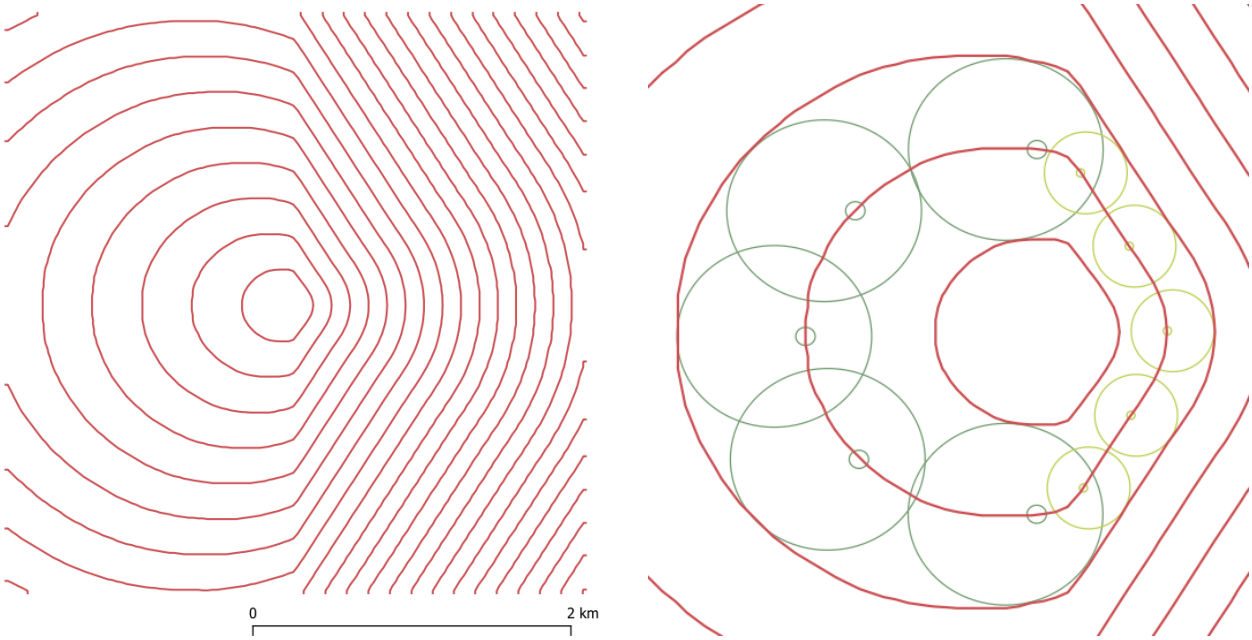
Result (ELMFIRE).

- Head rate of spread (5%): 1.92 m/min
- Head rate of spread (15%): 5.37 m/min

Huygens check. Rather than estimating L/W directly from the composite isochrones, BEHAVE-derived L/W values are used to draw 1-hour point-spread ellipses for each slope and overlay them on a given isochrone.

- Head L/W (5%): 1.01
- Head L/W (15%): 1.07

Comparison. Overlaying the 1-hour point spreads onto an isochrone aligns well with the next ELMFIRE outline (Figure 3.3).



(a) (b)

Figure 3.3: Left: The "Valley" verification case in ELMFIRE. Right: the 5° and 15° incline point ignition ellipses overlapped between two isochrones, with great match.

3.2.4 Quadrant

Goal. Verify that ELMFIRE handles spatially varying fuel models correctly by checking (i) that the composite fire perimeter remains physically reasonable when it crosses fuel boundaries and (ii) that the head rate of spread within each fuel matches BEHAVE.

Setup. Four fuel models arranged in quadrants, no wind, no slope. BEHAVE inputs:

- Fuel Models:
  1. FB2 (2) Timber (grass and understory)
  2. FB4 (4) Chaparral (6 feet)
  3. FB7 (7) Southern Rough
  4. FB8 (8) Closed Timber Litter
- 1-h fuel moisture: 6%
- 10-h fuel moisture: 7%
- 100-h fuel moisture: 8%
- Live herbaceous fuel moisture: 60%
- Live woody fuel moisture: 90%
- 20ft wind speed: 0 mph
- Slope steepness: 0%

Expected (BEHAVE).

- FB2 head rate of spread: 0.871 m/min
- FB4 head rate of spread: 1.492 m/min
- FB7 head rate of spread: 0.472 m/min
- FB8 head rate of spread: 0.080 m/min

Result (ELMFIRE).

- FB2 head rate of spread: 0.872 m/min
- FB4 head rate of spread: 1.974 m/min
- FB7 head rate of spread: 0.551 m/min
- FB8 head rate of spread: 0.080 m/min

Comparison. ELMFIRE matches BEHAVE closely for FB2 and FB8, while FB4 and FB7 exhibit higher ELMFIRE spread rates under the same inputs (discrepancy is currently being investigated). Qualitatively, the perimeter shape remains consistent across fuel boundaries (no nonphysical kinks or discontinuities), indicating that the multi-fuel coupling behaves sensibly; remaining quantitative differences are investigated further in Section 3.2.5.

3.2.5 Complex

The complex case is beyond analytical solution or comparison to an ideal solution and is the primary reason why one needs wildfire spread models. This case is added for qualitative review, to showcase the compilation of previous cases and investigate whether the results do linearly add to one another. As such, it is expected to see fast southward spread, with higher spread eastward (toward the higher slope) and towards the faster-spread fuel model. Indeed, that is exactly what ELMFIRE predicts.

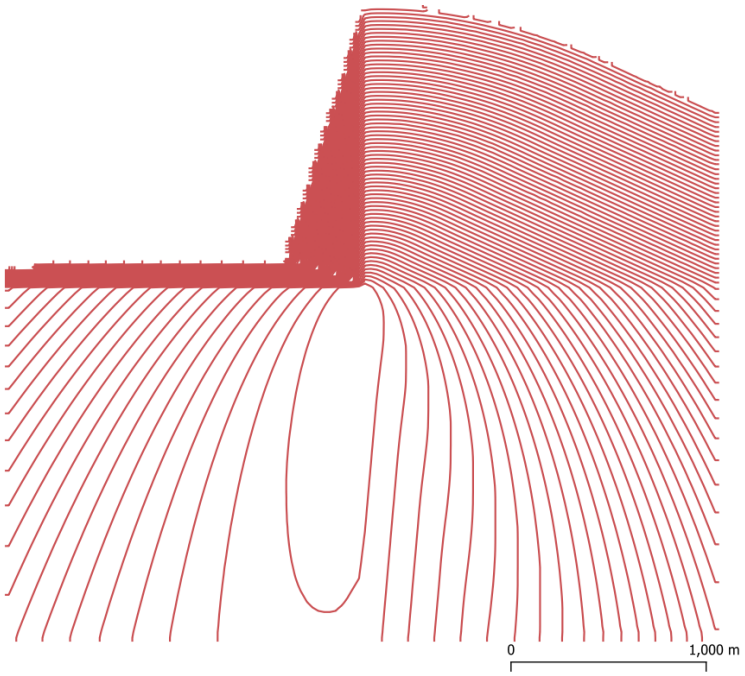


Figure 3.4: The "Complex" verification case results from ELMFIRE.

3.2.6 Moisture Quad

Goal. Verify that spatially varying fuel moisture inputs are parsed and applied correctly in ELMFIRE.

Setup. Uniform fuel model (FB3) with quadrant-wise 1-h fuel moisture (6%, 12%, 18%, 25%), no wind, no slope. BEHAVE inputs:

- Fuel Model: FB3 (3) Tall grass (static)
- 1-h fuel moisture: 6%, 12%, 18%, 25%
- 20ft wind speed: 0 mph
- Slope steepness: 0%

Expected (BEHAVE).

- 6%: 1.511 m/min
- 12%: 1.089 m/min
- 18%: 0.801 m/min
- 25%: 0 m/min

Result (ELMFIRE).

- 6%: 1.511 m/min
- 12%: 1.089 m/min
- 18%: 0.801 m/min
- 25%: 0 m/min

Comparison. ELMFIRE matches BEHAVE for all four moisture values. In particular, the 25% quadrant (approximately the moisture of extinction for fuel model 3) correctly yields zero spread, verifying correct interpretation and application of spatially varying fine fuel moisture.

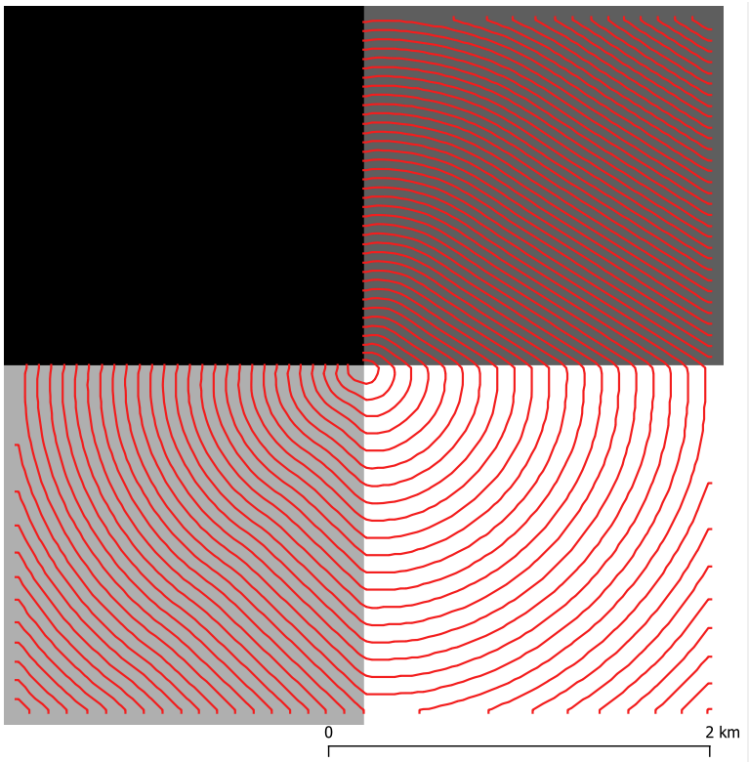


Figure 3.5: The "Moisture Quadrant" verification case results in ELMFIRE. Fuel moisture is plotted on the background with values ranging from 6 (white) to 25 (black).

3.2.7 Canopy

Goal. Verify crown fire initiation/classification (surface vs. passive vs. active crown fire) and crown spread rate calculations in ELMFIRE under nonzero canopy conditions.

Setup. A uniform domain with canopy parameters enabled. BEHAVE is used to obtain surface fire spread rate and fireline intensity, then the crown fire equations in Section 2.2.4 are applied analytically for comparison. Inputs:

- Fuel Model: FB3 (3) Tall Grass (static)
- 1-h fuel moisture: 6%
- 20ft wind speed: 1, 5, 10 mph
- Slope steepness: 0%
- Foliar Moisture Content: 90%
- Canopy Base Height: 1.6 m
- Canopy Height: 5.5 m
- Canopy Bulk Density: 0.10 $kg\,m^{-3}$
- Canopy Cover: 80%

Expected (analytical, using BEHAVE surface fire outputs). 1 mph.

- Head rate of spread (surface): 1.809 m/min
- Critical crown rate of spread R_0: 30.0 m/min
- Predicted active crown rate of spread R_C:4.44 m/min
- Criteria for active crowning CAC: 0.15
- Critical Fireline Intensity I_0: 477.3 kW/m
- Predicted Fireline Intensity I: 254.4 kW/m
- Crown Fire: None
- Predicted final rate of spread R: 1.809 m/min

5 mph.

- Head rate of spread (surface): 3.44 m/min
- Critical crown rate of spread R_0: 30.0 m/min
- Predicted active crown rate of spread R_C: 18.9 m/min
- Criteria for active crowning CAC: 0.63
- Critical Fireline Intensity I_0: 477.3 kW/m
- Predicted Fireline Intensity I: 449.0 kW/m
- Crown Fire: Passive
- Predicted final rate of spread R: 10.06 m/min

10 mph.

- Head rate of spread (surface): 5.26 m/min
- Critical crown rate of spread R_0: 30.0 m/min
- Predicted active crown rate of spread R_C: 35.3 m/min
- Criteria for active crowning CAC: 1.18
- Critical Fireline Intensity I_0: 477.3 kW/m
- Predicted Fireline Intensity I: 685.8 kW/m
- Crown Fire: Active

- Predicted final rate of spread R: 35.3 m/min

Result (ELMFIRE). 1 mph.

- Head rate of spread: 1.745 m/min
- Crown Fire: None

5 mph.

- Head rate of spread: 10.08 m/min
- Crown Fire: Passive

10 mph.

- Head rate of spread: 35.4 m/min
- Crown Fire: Active

Comparison. The 5 mph case transitions to passive crown fire with a spread rate consistent with the analytical crown prediction. The 1 mph case remains a surface fire with ELMFIRE head rate of spread and fireline intensity in close agreement with the surface-fire expectation. The 10 mph case produces active crown fire at the head fire.

As such, ELMFIRE can be considered verified for crown fire spread classification and calculation.

3.2.8 Firebrand

Goal. Verify that ELMFIRE correctly (i) samples user-defined lognormal firebrand transport distances and (ii) computes Eulerian ember flux magnitudes consistent with the analytical formulation.

Setup. Firebrand generation, transport, and ignition ("spotting") are supported in multiple modes in ELMFIRE. In this case, the most physical options are chosen, limiting the required user inputs. This means that spotting is setup as:

```
    &SPOTTING
ENABLE_SPOTTING                       = .TRUE.
USE_SUPERSEDED_SPOTTING  = .FALSE.
GENERATION_MODEL  = 'PER-MW'
SPOTTING_DISTANCE_MODEL  = 'EMPIRICAL'
ACCUMULATION_MODEL  = 'LAGRANGIAN'
IGNITION_MODEL  = 'DIRECT'
ENABLE_SURFACE_FIRE_SPOTTING          = .TRUE.
SURFACE_FIRE_SPOTTING_PERCENT(:)      = 100
CRITICAL_SPOTTING_FIRELINE_INTENSITY = 200.0
PIGN                                  = 1.0
/
```

Expected (analytical). (1) Lagrangian lognormal transport sampling. For a run using the predefined empirical (Sardoy) lognormal distribution, the distance sampling procedure described in Section 1.11.2 is applied. A simulation that includes firebrands produces `spotting_stats_0000001.csv`, logging each firebrand trajectory and, critically here, its calculated travel distance.

A centerline cell in a symmetric domain is used so the verification analogue is effectively 1D.

The (representative) per-cell inputs are:

- 20-ft wind speed (U_{20}): 5 mph
- Fireline intensity (I_b): 2035 kW/m
- Fuel Model: 3
- Cell size (C): 30 m
- Rate of Spread: 14.4 m/min

The total ember emitting time from this cell is the maximum between the total residence time as per the Rothermel model, and the total time taken for the fire to cross the cell. For fuel model 3:

$$\Delta t_R = 15.3s$$

$$\Delta T_{ROS} = \frac{30 \times 60}{14.4} = 125s$$

The expected total ember count emitted from the cell per step is:

$$N = 33.3 \times I_{b,MW} \times \Delta t = 8450$$

To compute the predicted travel distance, we refer to Section 2.3.2. The Froude number is computed by:

$$Fr = \frac{U_{20}}{\sqrt{gL_c}} \qquad L_c = \frac{I_b}{\rho_\infty C_{pg} \tau_\infty \sqrt{g}}$$

(using constants $g = 9.81$, $\rho_\infty = 1.1$, $C_{pg} = 1$, and $\tau_\infty = 300$), giving $L_c = 1.56\,\text{m}$ and $Fr = 0.56$.

Downwind and spanwise distribution parameters then are:

$$\mu_d = 1.47 \frac{(0.001 I_b)^{0.54}}{U_{10}^{0.55}} + 1.14$$

$$\sigma_d = 0.86 \frac{U_{10}^{0.44}}{(0.001 I_b)^{0.21}} + 0.19$$

$$\mu_s = 0$$

$$\sigma_s = 0.92 L_c$$

As such, we expect a transport distance distribution that follows a lognormal distribution with:

$$\mu_d = 2.42$$

$$\sigma_d = 1.31$$

Comparison. The logged Lagrangian lognormal transport distances match the independently calculated values exactly, validating the lognormal sampling implementation. For the Eulerian mode, the predicted and simulated ember flux values are of similar magnitude and follow the expected downwind decay; differences are largest for the highest-flux cell and are attributable to using averaged/representative I_b in the analytical calculation and to per-cell variability in the simulated fireline intensity. Overall, the transport component of the Eulerian spotting model and its associated submodules are considered verified.

Notes on ignition. For the Lagrangian model, ignition is a direct per-firebrand probability and does not require verification. For the Eulerian model, the ignition probability depends on coverage density (physical model) or a user-specified probability (simple model). A dedicated verification method for the Eulerian ignition criterion is not included at this time.

3.2.9 Overnight

Goal. Verify that ELMFIRE correctly computes sunrise/sunset timing and applies the diurnal (overnight) rate-of-spread reduction during the intended hours.

Setup. The verification cases are set at the equator and span multiple simulated hours; the "Point" case in particular is configured to run for 2 simulated days. ELMFIRE does not adjust 1-h fine fuel moisture dynamically once it is read as input. To retain the effect of night-time slowing down, a constant factor of 0.1 is used to scale down the predicted rate of spread during the hours between sunset and sunrise. These hours are calculated in ELMFIRE through the NOAA Solar Position Calculations (too extensive to repeat here).

For our test case, located at the equator, the sunrise and sunset times should be at 06:00 and 18:00, regardless of the time of year. Converted to UTC, since this raster is at longitude 22.5E, yields UTC times of 04:30 and 16:30 respectively The simulation is made to start at 10:00. With a burn period length of 10 hours and a center fraction of 0.667, the expected hours that the overnight adjustment factor applies are between 17:30 and 07:30.

Result (ELMFIRE). The resulting isochrones show the expected night-time slowdown and daytime speed-up (Figure 3.6).

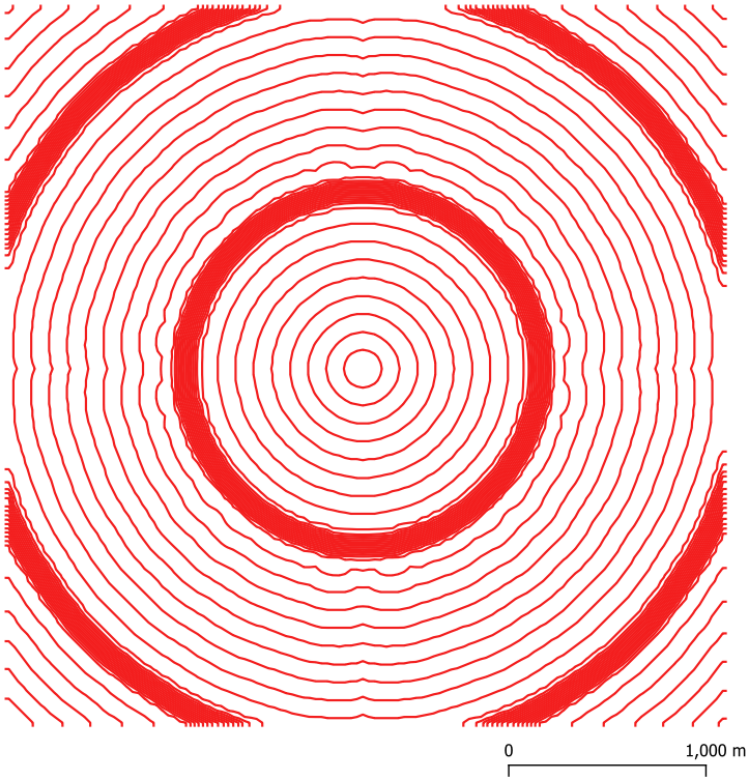


Figure 3.6: The results of the "Overnight" verification case.

Comparison. To confirm the timing, each isochrone is counted and annotated (Figure 3.7). The observed transition hours closely match the expected slowdown and speed-up times, verifying that ELMFIRE correctly calculates the sunrise and sunset hours and applies the diurnal correction factor.

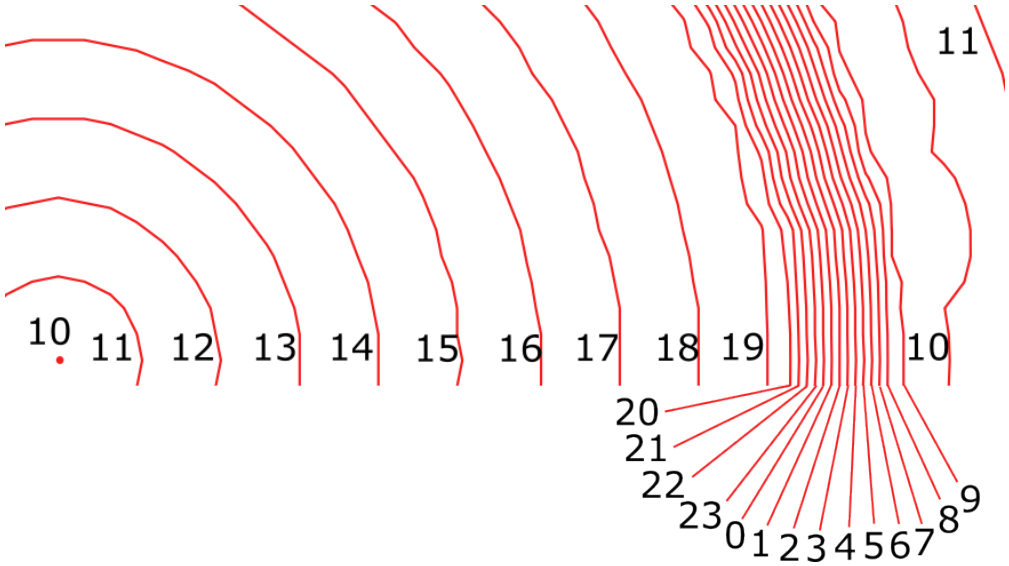


Figure 3.7: An isochrone count of the Overnight verification case, with hours of arrival labelled on.

3.2.10 Suppression

Goal. Verify the probabilistic initial attack containment logic and the containment evolution of both extended attack formulations.

Setup. The "Point" case is reused as a simple geometry where spread and containment can be analyzed directly.

Initial attack (stochastic). Expected. The initial attack submodel prescribes a probability of successful containment at the first attack opportunity. For this case, the initial attack time is set to 4800 seconds. Using a head rate of spread of 1.51 m/min, the (continuous) fire area at 4800 s is 45800 m^2 (11.3 acres). Because ELMFIRE uses a rasterized fire area, the area used internally is computed from the time-of-arrival raster: 69 burned cells at 30 m resolution, giving 62100 m^2 (15.3 acres, 6.2 hectares). The fireline intensity is 212.5 kW/m. With these values, the expected probability of containment is 54%.

Result. The verification case was run for 20000 realizations with different random seeds. About 52% of the simulated fires are contained at the initial attack time.

Comparison. The observed containment frequency (52%) is close to the expected probability (54%), verifying correct implementation of the initial attack success logic.

Extended attack – Area-Growth-Based Containment Model (deterministic). Expected. The "Point" case is rerun with `ENABLE_EXTENDED_ATTACK = .TRUE.` using the Area-Growth-Based Containment Model and default parameters, except for `DT_EXTENDED_ATTACK = 300`. The fuel model is changed to Fuel Model 8, yielding a head ROS of approximately 0.08 m/min. For this simple geometry and uniform suppression difficulty, the expected containment progression can be evaluated independently. Full containment is expected at approximately 88200 s.

Result. The simulation stops at 83988 seconds with final containment of 1.0. The end time is taken from the fire stats output.

Comparison. The simulated containment time is close to the independent expectation, verifying the containment evolution of the Area-Growth-Based Containment Model for this simple geometry.

Extended attack – Spatially Explicit Suppression Model (deterministic). Expected. The "Point" case is also used to verify the Spatially Explicit Suppression Model. The test uses a point ignition, no wind, flat terrain, uniform Fuel Model 8, and a head ROS of approximately 0.08 m/min. Full suppression capacity becomes available at 100000 seconds, with an available suppression capacity of 700 m/hr and an extended attack update interval of 3600 seconds. Indirect attack is disabled so that the test isolates the direct attack component. For this idealized case, full suppression is expected during the third direct-attack operation, at approximately 110800 s.

Result. ELMFIRE reaches full suppression at 100800 seconds, during the third direct-attack operation.

Comparison. The fire is fully suppressed during the third direct-attack operation at 100800 seconds, consistent with the expected behavior for this idealized case. The result verifies that the Spatially Explicit Suppression Model progressively allocates the available suppression capacity to eligible fireline segments until the active fireline is fully suppressed.

3.3 FBP Test Cases

The 2009 update to the FBP system [32] contained a set of 20 verification cases, to specifically test implementations of FBP in new models or fire spread codes. Each test varies the fuel model, wind and slope vectors, fine fuel moisture contents, and other parameters to test the correct implementation of each part of the system. The 2009* update also provides final and intermediate scalar values of the fire parameters (spread rate, fireline intensity, and fuel consumption of the surface and canopy respectively). These provided tests, and the accompanying tables of results, were instrumental in the implementation and testing of the FBP module of ELMFIRE, and the developers want to thank the authors for their foresight and for providing these cases.

As only the head fire model has been implemented in ELMFIRE, the results ignore the flank and backfire part of the FBP model, as it was not implemented. The 20 cases have been implemented in ELMFIRE and can be ran automatically, with a final verdict on the similarity of the results. At present, the maximum head rate of spread and fireline intensity are tested. a 10% error window has been introduced to allow for the effect of rounding errors (e.g. between the listed slope rise given in percent and the slope input ELMFIRE accepts in nearest integer degree).

The results are given in the table below. All tests can be found in the verification folder of ELMFIRE. To run these tests, navigate to `\elmfire\verification\FBP_tests\` and run `./runme.sh`. The script will clear existing results, produce the ELMFIRE input files and case folders, run ELMFIRE for each case, and compare the results with the ones in the 2009 Update. A 10% margin of error is included to account for differences in slope definition and other accuracy differences between ELMFIRE integer inputs and the float inputs. As standard, the level set mode results will be displayed, but the fire potential mode results can also be compared by running `python compare.py --mode 2` (some python packages might need to be installed through pip / conda).

Run	ROS_{target}	ROS_{sim}	$ROS_{error}\%$	FI_{target}	FI_{sim}	$FI_{error}\%$	ROS_{match}	FI_{match}
01	5.6	5.3	-4.6	3147.8	3007.8	-4.4	True	True
02	121.1	125.1	3.3	164491.3	172015.0	4.6	True	True
03	80.5	79.7	-1.0	42314.9	42367.6	0.1	True	True
04	16.7	17.6	5.4	18512.3	19853.1	7.2	True	True
05	0.0	0.0	-0.9	17.9	17.9	0.2	True	True
06	42.8	41.4	-3.2	36704.9	35544.4	-3.2	True	True
07	2.4	2.4	3.3	1490.6	1557.8	4.5	True	True
08	31.2	31.0	-0.6	11792.2	11870.6	0.7	True	True
09	13.3	13.0	-2.1	6817.1	6754.8	-0.9	True	True
10	18.1	18.0	-0.6	9474.8	9537.1	0.7	True	True
11	9.4	9.6	2.2	4010.2	4227.7	5.4	True	True
12	36.1	36.6	1.4	35166.5	36135.2	2.8	True	True
13	134.7	140.8	4.5	40408.0	14972.1	-62.9	True	False
14	0.4	0.4	8.1	24.1	46.2	91.4	True	False
15	13.3	13.4	0.5	3989.6	1420.5	-64.4	True	False
16	45.1	43.2	-4.2	105514.8	102373.2	-3.0	True	True
17	1.0	1.0	1.4	3478.6	3571.0	2.7	True	True
18	1.4	1.4	-1.4	4301.9	4318.7	0.4	True	True
19	4.4	4.3	-3.1	2687.6	2642.2	-1.7	True	True
20	17.6	17.0	-3.4	20766.5	20063.6	-3.4	True	True

Table 3.2: Comparison of target and maximum ROS and FI values across FBP verification runs. Note that cases 13, 14, and 15 are all grass model fires, with the test cases prescribing a different grass fuel load than that hardcoded in the FBP model ($3kg/m^2$). Repeating the calculations with the modified surface fuel load yields the expected FI values.

Chapter 4

Validation

4.1 CONUS: Automated Validation

ASTM E1355 Standard Guide for Evaluating the Predictive Capability of Deterministic Fire Models (2018) defines model validation as "the process of determining the degree to which a calculation method is an accurate representation of the real-world from the perspective of the intended uses of the calculation method". Model validation ("How closely do model calculations represent reality") is assessed by comparing modeled fire perimeters to observed fire perimeters.

Obtaining the input data required for validation case studies is a cumbersome process, resulting in most models being validated against one or two fires. In many cases, historic fires are picked that the model performs well in. Some models even use optimisation and parameterisation against the historic fire they use as validation, to prove that the model has the capability of producing accurate results. This, coupled with the fact that each model chooses different historic fires and different validation metrics, makes actual validation of wildfire spread models and comparison between them almost impossible.

To this end, Wildfire AV has been developed, the first publically available, large scale, historic wildfire validation pipeline (https://github.com/nick-cloudfire/WildfireAV). WildfireAV generates a large number of historic wildfire cases, and the input parameters required to simulate them. The pipeline creates the input files, runs ELMFIRE automatically, compares each simulation to the actual burn scar, computes similarity scores and compiles all the cases into sets for data analysis. A detailed explanation of the database, its sources, and its capabilities and limitations has been submitted for publication.

The complete validation PDF report, comparing both ELMFIRE and FARSITE (to provide an established metric for comparison), has been included in the validation/ folder. The results are summarised here. All the results below use simple fire spread with canopy effects, no firebrands, barriers, no suppression.

Overall, ELMFIRE seems to have a more balanced output with regards to over- or underestimation of total fire area compared to Farsite, which may be attributed to its breaching criterion (Farsite hard stops fire spread at barriers).

In terms of overall similarity score, ELMFIRE and Farsite obtain broadly similar scores, with ELMFIRE having a mean Jaccard coefficient of 0.178, a mean Sorensen Coefficient of 0.278 and a mean Cohen's kappa of 0.241. THe equivalent Farsite values are 0.176, 0.274 and 0.249. The distributions also look similar, with the elmfire curves generally skewing towards greater bulk similarity indices.

A sample fire output is given below. For clarity, the chosen wildfire is the most accurate fire (according

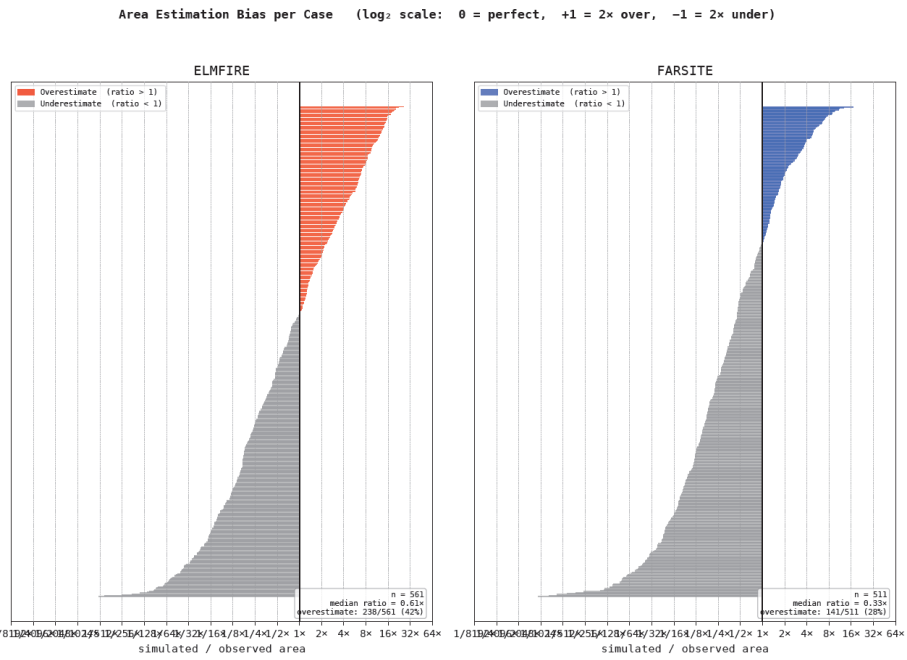


Figure 4.1: A bias estimation plot, with the over- and underestimation of each case plotted (through simple total area comparisons.

to its Jaccard similarity) for ELMFIRE, and shows the advantage of a conditional breaching criterion. All wildfire cases are accompanied with such a figure.

Finally, for completeness, below are the top and bottom performing wildfire simulations of the dataset.

4.2 Canada: Dogrib Fire

The Dogrib Fire is the standard provided validation case of the Prometheus program, the main CFFDRS fire spread model implementation [28]. The Dogrib fire (ID RWF085) ignited on September 25th 2001 with two main growth spurts. The first growth spurt occured between ignition and October 4th, at which point the fire was classified under control. On October 16th a wind event caused a major fire run, significantly increasing the total area of the fire. The second growth spurt of the fire was the focus of the Prometheus validation, shown in Fig. 4.5, and has been recreated in ELMFIRE for the purposes of comparing the outputs of the ELMFIRE CFFDRS implementation and the real fire / Prometheus prediction. The simulation will focus on the fire spread between October 16th 13:00 local time and 19:42 local time (specified as scenario 2 in the Prometheus validation document).

The supporting document of the Prometheus verification (that can be found in the Prometheus online distribution) describe the Dogrib fire extensively, along with the input data and parameters used in the simulation. The simulation used four wind scenarios:

1. Weather station wind: magnitude and direction are taken directly from the local weather station. Wind magnitude is given in kilometers per hour, 10m up from vegetation canopy. Wind direction is quantized in 45 degree increments.
2. Modified weather station wind: wind magnitude is unchanged. Wind direction gets 15 degrees added to make the wind vector more consistent with fire spread (to account for 45 degree weather station

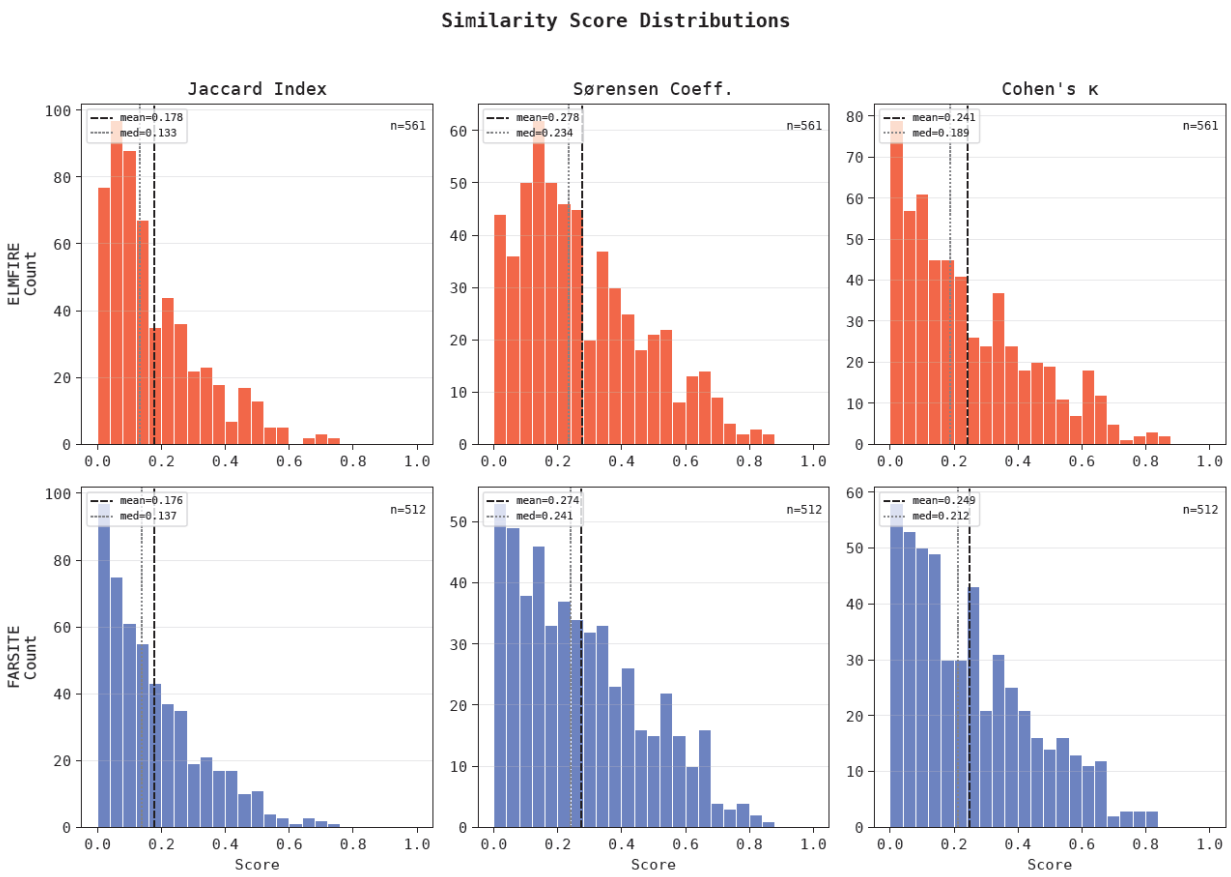


Figure 4.2: The similarity coefficient histograms of ELMFIRE and Farsite, with mean and median values superimposed.

step range). Wind is also defaulted to 280 within the river area to simulate wind channeling caused by the river valley

3. Windninja simulation with conservation of mass, using the weather station data as inputs.
4. Windninja simulation with conservation of mass and momentum, using the weather station data as inputs.

The results with each set of wind inputs are given in detail in the Prometheus validation document, with the end result being that scenario 2 (modified weather station wind) produces the most accurate burn scar. As such, the ELMFIRE simulation will use scenario 2's wind magnitude and direction.

In terms of the input parameters, the ones that remain unchanged between Prometheus and ELMFIRE are:

- Elevation: DEM file provided in Prometheus
- Slope: Calculated with GDAL from the DEM file
- Aspect: Calculated with GDAL from the DEM file
- FBP Fuel Map: Provided in Prometheus
- Starting DMC: 64 calculated from the Prometheus weather stream
- Starting DC: 535 calculated from the Prometheus weather stream
- Ignition: Specified with coordinates in Prometheus

The following input parameters were given in the Prometheus case but had to be converted to be used in

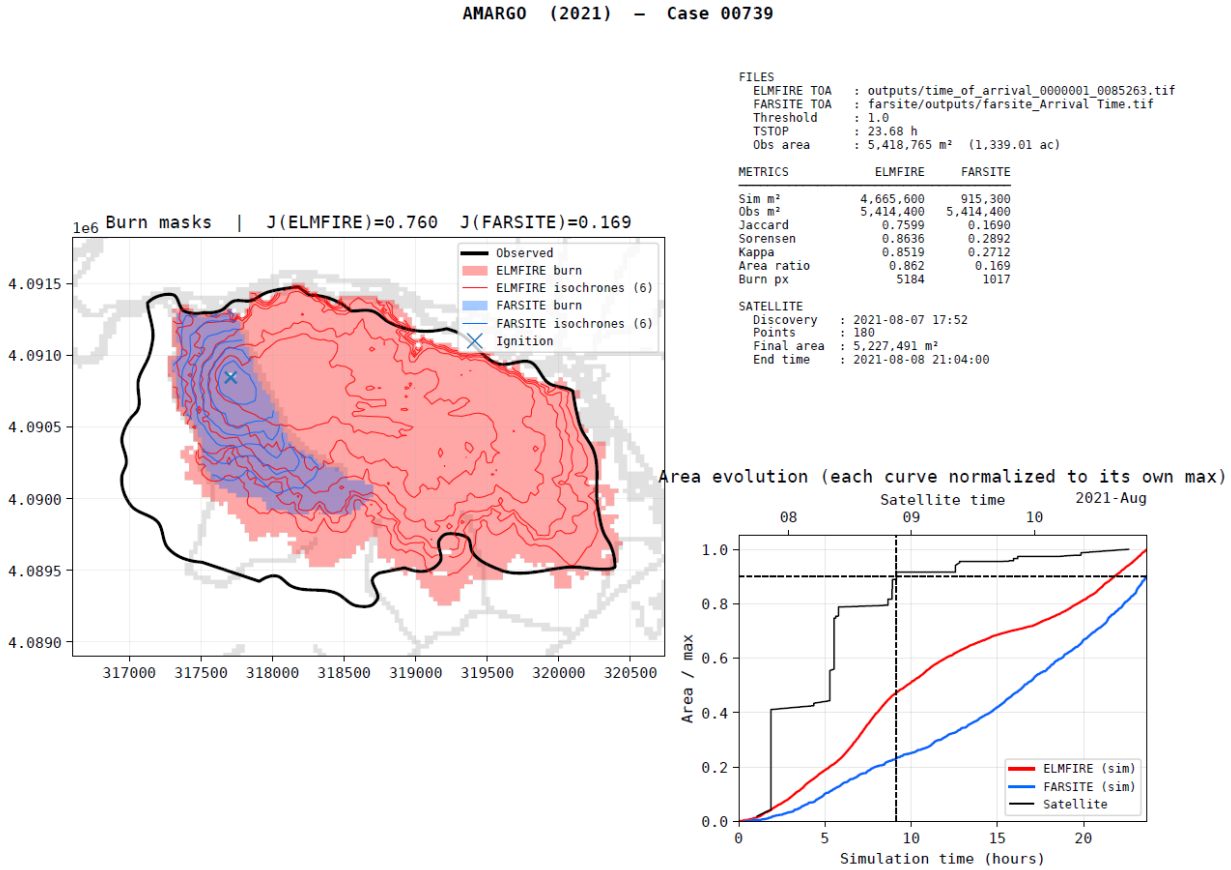


Figure 4.3: An exmaple output of a WildfireAV case, in this case the 2021 Amargo fire, the best performing ELMFIRE simulation.

ELMFIRE:

- Wind magnitude: given in the weather stream in kph, converted to mph
- Wind direction: taken from the weather stream and then modified in QGIS to include the weather modifications of Prometheus.
- Fuel Moisture Content: Prometheus calculates Hourly Fine Fuel Moisture Code (HFFMC) from the weather stream. ELMFIRE requires fuel moisture content (1-hour) as an input. The 1-h fuel moisture content of the area can be calculated through the nelson model (as is done in the US), using the same weather stream that is provided by Prometheus. Doing so yields fuel moisture values of 18% to 20%. The Prometheus-derived HFFMC values translate to 9% to 11% fuel moisture content. For the validation case, the HFFMC-derived fuel moisture content values are used (spatially consistent). M10 and M100 are irrelevant to CFFDRS calculations and are taken from the Nelson model results.
- Degree of curing: Prometheus specifies 100% degree of curing for all grass type fuels. ELMFIRE translates this input as a Live Herbaceous moisture content of 30%.
- FWI Interpolation: The dogrib simulation provided with the Prometheus installation is ran with the Lawson diurnal FWI interpolation model. The Lawson model interpolates hourly fine fuel moisture code (HFFMC) and Hourly Initial Spread Index (HISI), the two most critical parameters of fire spread, from a diurnal drying and wetting profile, using the daily midday weather values. The alternative method is the Van Wagner method which relies on hourly weather values to calculate the HFFMC and

Validation Report — Summary

Total cases: 1295 Successful: 561 Non-ignited (Jaccard < 0.01): 36 Failed (no outputs): 698

ELMFIRE — Top 10 by Jaccard

Case	Fire	Jaccard	Sørensen	κ
00739	AMARGO	0.7599	0.8636	0.8519
01457	MOSS RANCH	0.7560	0.8611	0.8532
00738	JACK	0.7183	0.8361	0.8212
01232	WOODWARD	0.7098	0.8303	0.8141
00798	BEAR	0.6854	0.8134	0.8003
00732	TURNER RIVER	0.6465	0.7853	0.7707
00734	SHAMROCK	0.6450	0.7842	0.7643
01139	ENCINOS	0.5966	0.7473	0.7301
01321	LIME	0.5762	0.7311	0.7029
01441	KUGRUK RIVER 1	0.5730	0.7285	0.7074

FARSITE — Top 10 by Jaccard

Case	Fire	Jaccard	Sørensen	κ
00738	JACK	0.7306	0.8443	0.8306
01355	BOUNDARY RIVER	0.7104	0.8307	0.8182
01166	TADPOLE	0.7059	0.8276	0.8131
01287	LITTLE BEAR	0.6531	0.7902	0.7745
00798	BEAR	0.6486	0.7868	0.7751
01362	PETE ANDREWS CRE	0.6413	0.7815	0.7704
00734	SHAMROCK	0.6343	0.7762	0.7562
00242	GOVERNMENT	0.5924	0.7441	0.7191
00732	TURNER RIVER	0.5888	0.7412	0.7248
01098	BOZARTH	0.5814	0.7353	0.7271

ELMFIRE — Bottom 10 by Jaccard

Case	Fire	Jaccard	Sørensen	κ
01227	NIMIUKTUK RIVER	0.0019	0.0037	0.0035
00689	HUGHES	0.0022	0.0045	0.0043
00333	WILLOW CREEK	0.0032	0.0065	0.0056
00192	COCONUT	0.0039	0.0077	0.0072
00902	SARPY	0.0053	0.0105	0.0100
01331	POT HOLE	0.0055	0.0110	0.0099
01485	SUN	0.0067	0.0134	0.0115
01341	MOUNTAIN	0.0085	0.0168	0.0152
00901	PEARL HILL	0.0085	0.0169	0.0160
00753	COPPER CANYON	0.0088	0.0174	0.0003

FARSITE — Bottom 10 by Jaccard

Case	Fire	Jaccard	Sørensen	κ
01317	WALKER	0.0000	0.0000	0.0000
01109	CANAL	0.0001	0.0002	-0.0097
01243	CENTRAL	0.0009	0.0018	0.0017
01282	IKES	0.0013	0.0025	0.0024
00821	CRONAN	0.0016	0.0032	0.0030
01224	IZAVIKNEK RIVER	0.0022	0.0044	0.0041
01247	BIG HOLLOW	0.0027	0.0053	0.0050
01218	TITNUK CREEK	0.0037	0.0074	0.0072
01132	CALWOOD	0.0039	0.0077	0.0070
00721	CUERVITO	0.0054	0.0108	-0.0916

Figure 4.4: The top and bottom 10 performing simulations for each model. See the full report for analytical results.

HISI. The two methods offer advantages and disadvantages in their application and accuracy; their main difference is that Lawson produces a strong diurnal cooling effect whereas Van Wagner has a weak nighttime cooling effect. In the Dogrib simulation of ELMFIRE, the Lawson model is used, but ELMFIRE can only implement the Van Wagner model. As such, the simulation that is used here to represent the results of Prometheus were rerun with the Van Wagner interpolation model, to make the comparison more straightforward.

The results of the Prometheus and ELMFIRE simulation are presented in Fig. 4.6 and show that ELMFIRE predicts significantly smaller fire size than Prometheus, and both models underpredict the extent of the fire. Focusing on the discrepancy between ELMFIRE and Prometheus, the difference between the two predictions can be condensed to the way the two models propagate the fire in 2D.

Take the example given in table 3 or the 2009 FBP Update document [32]. For a fire with head fire rate of spread of 17.46 m/min, and a Length-to-breadth ratio of 1.98 (referring to the length to breadth ratio of the elliptical fire spread from a point source. The original document predicts a backwards rate of spread of 2.16 m/min, using the FBP methodology. ELMFIRE uses the updated methododology employed in Farsite to calculate the rate of spread in any direction. It specifically calculates the backwards rate of spread through:

$$ROS_b = ROS\frac{LOW - \sqrt{LOW^2 - 1}}{LOW + \sqrt{LOW^2 - 1}}$$

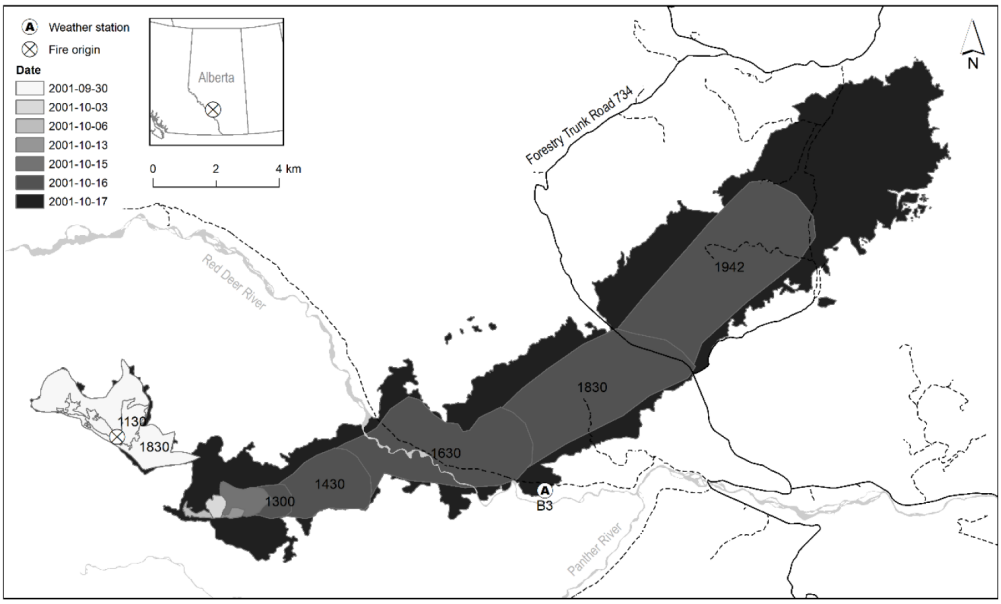


Figure 4.5: The extent and propagation of the 2001 Dogrib fire. Figure taken from the Prometheus validation supporting document "Case study information pertaining to the Dogrib sample dataset for Prometheus" by Neal McLoughlin, last revised February 3rd 2019

Plugging in the values from the update document yields a value of 1.29 m/min. This behavior can be considered a universal property of ELMFIRE and a key driver of the difference between it and Prometheus. In the dogrib simulations, a critical component of the fire spread is the transverse travel of the fire over and across the landscape ridges. Before the fire crosses the river, in particular, it has to spread on a 60 degree downhill, shown in Fig. 4.7, which would produce drastically different rates of spread. A comparison of multiple models in the literature [14] shows that different models treat point sources with different elliptical shapes and dimensions, and the ellipses of ELMFIRE and Prometheus are particularly different in their flank and backfires.

To further investigate this, the Dogrib simulations were repeated, with terrain effects turned off, meaning the elevation was changed to a constant value, and the slope and aspect were both set to zero. The results depicted in Fig. 4.8 show that, with only the wind pushing the fire, and thus the burn extent primarily depending on the head fire spread rate, the results are near identical (barring differences in flank fire).

With the above results, it is clear that ELMFIRE and Prometheus may produce different results, because of how they treat topography and wind interactions, with regards to spread rates and directional spread rate interpolation. Despite this result, the above simulations show that ELMFIRE is capable of producing similar outputs to Prometheus, closely approximating the behavior of real fires.

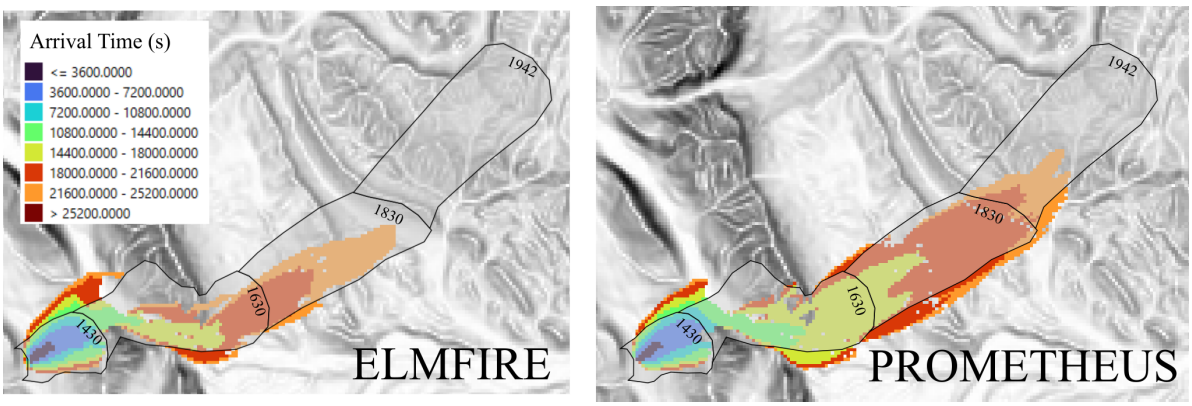


Figure 4.6: The burn scars of ELMFIRE and Prometheus for the Dogrib fire with identical / parsed input parameters.

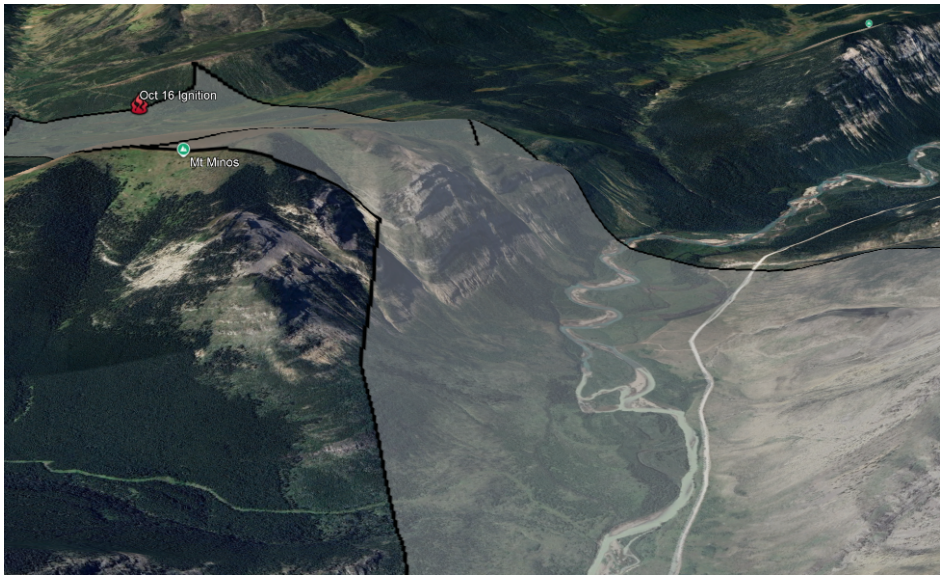


Figure 4.7: The 60 degree downhill part of fire spread shown in Google Earth. It is highly likely that the fire spread of the real fire was assisted and accelerated by firebrands travelling from the top of the hill to the valley below and beyond the river. However, firebrand-assisted propagation is already reflected in the CFFDRS spread rate regressions, and as such should not be activated in ELMFIRE.

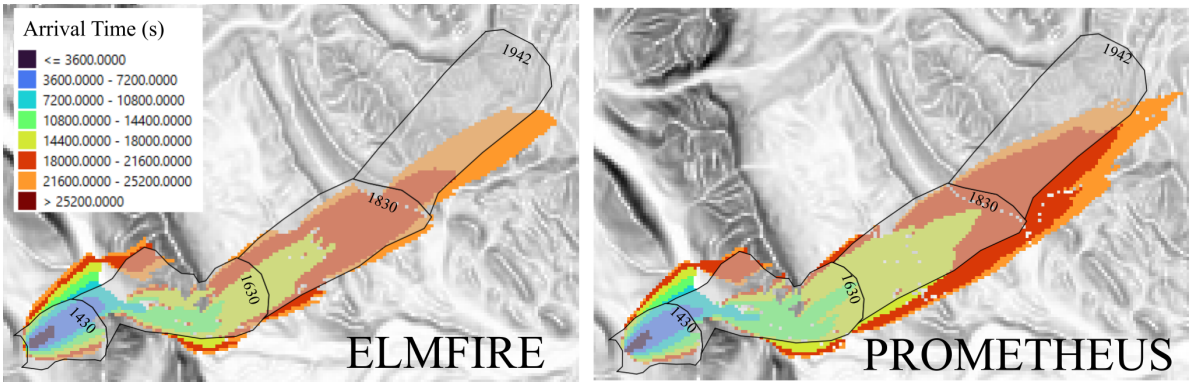


Figure 4.8: The Dogrib simulation case with flat terrain. Only the wind affects the fire spread direction and magnitude.

Chapter 5

Input Parameters for ELMFIRE

Table 5.1: All the input options and namelists in ELMFIRE.

Name	Default value	Variable Type	Units	Explanation
	MISCELLANEOUS			
BUILDING_FUEL_MODEL_FILE	'building_fuel_models.csv'	string	file	1.7
FUEL_MODEL_FILE	'null'	string	file	1.7
MISCELLANEOUS_INPUTS_DIRECTORY	'null'	string	path	1.7
PATH_TO_GDAL	'/usr/bin'	string	path	1.7
SCRATCH	'null'	string	path	1.7
	SMOKE			
DT_SMOKE_OUTPUTS	3600	float	seconds	1.18
ENABLE_SMOKE_OUTPUTS	.FALSE.	bool	-	1.18
PM_EMISSION_FACTOR_FLAMING	17.4	float	g/kg	1.18
PM_EMISSION_FACTOR_SMOLDERING	49.8	float	g/kg	1.18
DRY_WOOD_CALORIFIC_VALUE	19	float	MJ/kg	1.18
FLAMING_TIME	180	float	s	1.18
SMOLDERING_TIME	3600	float	s	1.18

INPUTS				
ADJ_FILENAME	-	string	file	1.2
ASP_FILENAME	-	string	file	1.2
USE_BARRIERS	.FALSE.	bool	-	1.10
BARRIER_FILENAME	-	string	file	1.10
BLDG_AREA_FILENAME	-	string	file	1.13
BLDG_FOOTPRINT_FRAC_FILENAME	-	string	file	1.13
BLDG_FUEL_MODEL_FILENAME	-	string	file	1.13
BLDG_NONBURNABLE_FRAC_FILENAME	-	string	file	1.13
BLDG_SEPARATION_DIST_FILENAME	-	string	file	1.13
CBD_FILENAME	-	string	file	1.2
CBD_TIMES_100	.TRUE.	bool	-	1.2
CBH_FILENAME	-	string	file	1.2
CBH_TIMES_10	.TRUE.	bool	-	1.2
CC_FILENAME	-	string	file	1.2
CC_IN_PERCENT	.TRUE.	bool	-	1.2
CH_FILENAME	-	string	file	1.2
CH_TIMES_10	.TRUE.	bool	-	1.2
DEM_FILENAME	-	string	file	1.2
DT_METEOROLOGY	-9999.9	float	seconds	1.2
FBFM_FILENAME	-	string	file	1.2
FMC_FILENAME	-	string	file	1.2
FOLIAR_MOISTURE_CONTENT	90	float	%	1.2
FUELS_AND_TOPOGRAPHY_DIRECTORY	-	string	file	1.2
GRID_DECLINATION	0	float	degrees	1.2.2
IGNITION_MASK_FILENAME	-	string	file	1.12.1
LANDSCAPE_FILENAME	-	string	file	1.2
LAND_VALUE_FILENAME	-	string	file	1.16
LH_MOISTURE_CONTENT	60	float	%	1.2
LW_MOISTURE_CONTENT	60	float	%	1.2
DEAD_MC_IN_PERCENT	.TRUE.	bool	-	1.2
LIVE_MC_IN_PERCENT	.TRUE.	bool	-	1.2

PHI_FILENAME	-	string	file	1.2
POPULATION_DENSITY_FILENAME	-	string	file	1.16
REAL_ESTATE_VALUE_FILENAME	-	string	file	1.16
SDI_FILENAME	-	string	file	1.14
PCL_FILENAME	-	string	file	1.14
SLP_FILENAME	-	string	file	1.2
ERC_FILENAME	-	string	file	(to be removed)
FMC_FILENAME	-	string	file	1.2
IGNITIONS_CSV_FILENAME	-	string	file	1.12.1
M100_FILENAME	-	string	file	1.2
M10_FILENAME	-	string	file	1.2
M1_FILENAME	-	string	file	1.2
MLH_FILENAME	-	string	file	1.2
MLW_FILENAME	-	string	file	1.2
PYROMES_FILENAME	-	string	file	1.17
ROTATE_ASP	.FALSE.	bool	-	1.2.2
ROTATE_WD	.FALSE.	bool	-	1.2.2
TIMED_LOCATIONS_CSV	'null'	string	file	1.6
WD_FILENAME	-	string	file	1.2
WS_FILENAME	-	string	file	1.2
USE_BSQ_XML_HEADER	.TRUE.	bool	-	1.7
USE_CONSTANT_FMC	.TRUE.	bool	-	1.2
USE_CONSTANT_LH	.TRUE.	bool	-	1.2
USE_CONSTANT_LW	.TRUE.	bool	-	1.2
USE_EXISTING_BSQS	.FALSE.	bool	-	1.7
USE_LAND_VALUE	.FALSE.	bool	-	1.16
USE_POPULATION_DENSITY	.FALSE.	bool	-	1.16
USE_REAL_ESTATE_VALUE	.FALSE.	bool	-	1.16
USE_TILED_IO	.FALSE.	bool	-	1.7
VRT_INSTEAD_OF_TIF	.FALSE.	bool	-	1.2
WEATHER_DIRECTORY	-	string	path	1.2
WS_AT_10M	.FALSE.	bool	-	1.2

WS_IN_KPH	.FALSE.	bool	-	1.2
ONLY_READ_NEEDED_WX_BANDS	.FALSE.	bool	-	1.4
SURFACE_SPREAD_MODEL	'ROTHERMEL'	string	switch	1.2.1
START_DC	400	float	-	1.2.1
START_DMC	80	float	-	1.2.1
DAILY_WEATHER_FILENAME	-	string	file	1.2.1
	OUTPUTS			
ACCUMULATE_EMBER_FLUX	.FALSE.	bool	-	
BINARY_OUTPUTS_DUMP_FRACTION	1.00	float	fraction	1.6
CALCULATE_FLAME_LENGTH_STATS	.FALSE.	bool	-	1.6
CALCULATE_TIMES_BURNED	.FALSE.	bool	-	1.12.5
CONVERT_TO_GEOTIFF	.TRUE.	bool	-	1.6
DTDUMP	3600	float	seconds	1.6
DUMP_AFFECTED_LAND_VALUE	.FALSE.	bool	-	1.6
DUMP_AFFECTED_POPULATION	.FALSE.	bool	-	1.6
DUMP_AFFECTED_REAL_ESTATE_VALUE	.FALSE.	bool	-	1.6
DUMP_BINARY_OUTPUTS	.FALSE.	bool	-	1.6
DUMP_CFFDRS_DEBUG	.FALSE.	bool	-	
DUMP_CROWN_FIRE	.FALSE.	bool	-	1.6
DUMP_CROWN_FIRE_AREA	.FALSE.	bool	-	1.6
DUMP_CRITICAL_FLIN	.FALSE.	bool	-	1.6
DUMP_EMBER_FLUX	.FALSE.	bool	-	1.11.8
DUMP_EMBER_FLUX_TRANSIENT	.FALSE.	bool	-	1.11.8
DUMP_EMBER_IGNITION	.FALSE.	bool	-	1.11.8
DUMP_EMITIMES	.FALSE.	bool	-	1.18
DUMP_EVERY_STEP	.FALSE.	bool	-	
DUMP_FIRE_SIZE_STATS	.TRUE.	bool	-	1.6
DUMP_FIRE_VOLUME	.FALSE.	bool	-	1.6
DUMP_FLAME_LENGTH	.FALSE.	bool	-	1.6
DUMP_FLIN	.FALSE.	bool	-	1.6
DUMP_FUEL_CONSUMPTION	.FALSE.	bool	-	
DUMP_HOURLY_RASTERS	.FALSE.	bool	-	1.6

DUMP_HPUA	.FALSE.	bool	-	1.6
DUMP_HRR_TRANSIENT	.FALSE.	bool	-	1.6
DUMP_PHI	.FALSE.	bool	-	1.6
DUMP_REACTION_INTENSITY	.FALSE.	bool	-	1.6
DUMP_SPOTTING_OUTPUTS	.FALSE.	bool	-	1.6, 1.11.8
DUMP_SPREAD_DIRECTION	.FALSE.	bool	-	1.6
DUMP_SPREAD_RATE	.FALSE.	bool	-	1.6
DUMP_SURFACE_FIRE	.FALSE.	bool	-	1.6
DUMP_SURFACE_FIRE_AREA	.FALSE.	bool	-	1.6
DUMP_TAGGED	.FALSE.	bool	-	1.6
DUMP_TIME_OF_ARRIVAL	.FALSE.	bool	-	1.6
DUMP_TIMINGS	.FALSE.	bool	-	1.6
DUMP_TOTAL_DFC_RECEIVED	.FALSE.	bool	-	1.6
DUMP_TOTAL_RAD_RECEIVED	.FALSE.	bool	-	1.6
DUMP_TRANSIENT_ACREAGE	.FALSE.	bool	-	1.6
DUMP_TRANSIENT_DFC	.FALSE.	bool	-	
DUMP_TRANSIENT_RAD	.FALSE.	bool	-	
DUMP_VELOCITY	.FALSE.	bool	-	1.6
DUMP_WD20	.FALSE.	bool	-	1.6
DUMP_WS20	.FALSE.	bool	-	1.6
EMBER_COUNT_BIN_HI(:)	0	integer*2	ember count	1.6, 1.11.8
EMBER_COUNT_BIN_LO(:)	0	integer*2	ember count	1.6, 1.11.8
FLAME_LENGTH_BIN_HI(:)	0	float	feet	1.6
FLAME_LENGTH_BIN_LO(:)	0	float	feet	1.6
FULL_BINARY_OUTPUTS	.TRUE.	bool	-	1.6
MINIMUM_AREA_FOR_BINARY_OUTPUTS	0	float	acres	1.6
NUM_EMBER_COUNT_BINS	0	integer	bin count	1.6, 1.11.8
NUM_FLAME_LENGTH_BINS	0	integer	bin count	1.6
NUM_VIRTUAL_STATIONS	0	integer	integer	1.6.1
OUTPUTS_DIRECTORY	-	string	path	1.6
RUN_ID	-	string	string	1.6
SPREAD_RATE_IN_M	.FALSE.	bool	-	1.6

TIME_AT_BURNED_ACRES(:)	-9.00E+09	float	acres	1.6
USE_EMBER_COUNT_BINS	.FALSE.	bool	-	1.6, 1.11.8
USE_FLAME_LENGTH_BINS	.FALSE.	bool	-	1.6
USE_FOUR_DIGITS_IN_IWX_BAND	.FALSE.	bool	-	1.12.2
VIRTUAL_STATION_X(:)	0	integer	coordinate	1.6.1
VIRTUAL_STATION_Y(:)	0	integer	coordinate	1.6.1
	TIME_CONTROL			
BAND_ONE_HOUR_OF_YEAR	0	integer	hour	1.4
BURN_PERIOD_CENTER_FRAC	0.667	float	float	1.4.1
BURN_PERIOD_LENGTH	10	float	hours	1.4.1
CURRENT_YEAR	-1	integer	year	1.4.1
DT_INTERPOLATE_M1	300	float	seconds	1.4
DT_INTERPOLATE_M10	3000	float	seconds	1.4
DT_INTERPOLATE_M100	30000	float	seconds	1.4
DT_INTERPOLATE_MLH	9.00E+08	float	seconds	1.4
DT_INTERPOLATE_MLW	9.00E+08	float	seconds	1.4
DT_INTERPOLATE_FMC	9.00E+08	float	seconds	1.4
DT_INTERPOLATE_WIND	300	float	seconds	1.4
FORECAST_START_HOUR	20	float	hour	1.4.1
HOUR_OF_YEAR	-1	integer	hour	1.4.1
OVERNIGHT_ADJUSTMENT_FACTOR	0.1	float	float	1.4.1
RANDOMIZE_SIMULATION_TSTOP	.FALSE.	bool	-	1.4
SIMULATION_DT	5	float	seconds	1.4
SIMULATION_DTMAX	600	float	seconds	1.4
SIMULATION_TSTART	0	float	seconds	1.4
SIMULATION_TSTOP	3600	float	seconds	1.4
SUNRISE_HOUR	-1	float	hour	1.4.1
SUNSET_HOUR	-1	float	hour	1.4.1
TARGET_CFL	0.4	float	float	1.4
USE_DIURNAL_ADJUSTMENT_FACTOR	.FALSE.	bool	-	1.4.1
	MONTE_CARLO			
ADD_TO_IGNITION_MASK	-9.00E+09	float	ignition mask value	1.12.1

ALLOW_MULTIPLE_IGNITIONS_AT_A_PIXEL	.FALSE.	bool	-	1.12.1
CSV_FIXED_IGNITION_LOCATIONS	.FALSE.	bool	-	1.12.1
EDGEBUFFER	3000	float	meters	1.12.1
ERC_IS_PLIGNRATE	.FALSE.	bool	-	(to be removed)
IGNITION_MASK_SCALE_FACTOR	1	float	float	1.12.1
METEOROLOGY_BAND_START	-1	integer	weather band	1.12.2
METEOROLOGY_BAND_STOP	-1	integer	weather band	1.12.2
METEOROLOGY_BAND_SKIP_INTERVAL	-1	integer	weather band	1.12.2
NUM_ENSEMBLE_MEMBERS	1	integer	integer	1.12.1
NUM_RASTERS_TO_PERTURB	0	integer	integer	1.12.3
NUM_METEOROLOGY_TIMES	-1	integer	integer	1.12.2
PDF_TYPE(:)	'null'	character	distribution type	1.12.3
PDF_LOWER_LIMIT(:)	0	float	value dependent	1.12.3
PDF_UPPER_LIMIT(:)	0	float	value dependent	1.12.3
PDF_MEAN	0	float	value dependent	1.12.3
PDF_SIGMA	0	float	value dependent	1.12.3
PERCENT_OF_PIXELS_TO_IGNITE	5	float	%	1.12.1
RANDOM_IGNITIONS	.FALSE.	bool	-	1.12.1
RANDOM_IGNITIONS_TYPE	1	integer	random ignition mode	1.12.1
RASTER_TO_PERTURB(:)	'null'	string	layer name	1.12.3
SEED	2024	integer	integer	1.9.5
SPATIAL_PERTURBATION(:)	'null'	string	perturbation type	1.12.3
TEMPORAL_PERTURBATION(:)	'null'	string	perturbation type	1.12.3
USE_ERC	.FALSE.	bool	-	(to be removed)
USE_IGNITION_MASK	.FALSE.	bool	-	1.12.1
WIND_DIRECTION_FLUCTUATION_INTENSITY_MAX	-1	float	degrees	1.12.4
WIND_DIRECTION_FLUCTUATION_INTENSITY_MIN	-1	float	degrees	1.12.4
WIND_SPEED_FLUCTUATION_INTENSITY_MAX	-1	float	mph	1.12.4
WIND_SPEED_FLUCTUATION_INTENSITY_MIN	-1	float	mph	1.12.4
PERTURB_WIND_DIRECTION_FLUCTUATION_INTENSITY	.FALSE.	bool	-	1.12.4
PERTURB_WIND_SPEED_FLUCTUATION_INTENSITY	.FALSE.	bool	-	1.12.4
POINT_WIND_TO_CENTER	.FALSE.	bool	-	1.12.1

ALLOW_NONBURNABLE_PIXEL_IGNITION	.TRUE.	bool	-	1.12.1
BANDTHICKNESS	2	integer	pixels	1.9.5
CRITICAL_CANOPY_COVER	0.39	float	%	1.9.3
CROWN_FIRE_ADJ	1	float	scale factor	1.9.3
CROWN_FIRE_MODEL	1	integer	categorical	1.9.3
CROWN_FIRE_SPREAD_RATE_LIMIT	250	float	ft/min	1.9.3
CROWN_RATIO	1	float	ratio	1.2
FEEDBACK_LEVEL	0	integer	output level	1.9.1
T_LINE_IGN(:)	-1	float	seconds	1.5.2
X_LINE_IGN_START(:)	-1	float	coordinate	1.5.2
X_LINE_IGN_END(:)	-1	float	coordinate	1.5.2
Y_LINE_IGN_START(:)	-1	float	coordinate	1.5.2
Y_LINE_IGN_END(:)	-1	float	coordinate	1.5.2
DT_WIND_FLUCTUATIONS	15	float	seconds	1.9.4
ESTIMATE_URBAN_LOSSES	.FALSE.	bool	-	1.13
MAX_LOW	8	float	ratio	1.9.2
MAX_RUNTIME	999999	float	seconts	1.9.5
MODE	1	integer	categorical	1.15
MULTIPLE_HOSTS	.FALSE.	bool	-	1.8
NUM_IGNITIONS	0	integer	ignitions	1.5.1
PHIS_ADJ	1.00E+00	float	scale factor	1.9.2
PHIW_ADJ	1.00E+00	float	scale factor	1.9.2
PLIGNRATE_MIN	0.00E+00	float	percentage	(to be removed)
RANDOMIZE_RANDOM_SEED	.FALSE.	bool	-	1.9.5
SURFACE_ACCELERATION_TIME_CONSTANT	1	float	multiplier	(to be removed)
T_IGN(:)	0	float	seconds	1.5.1
UNTAG_CELLS_TIMESTEP_INTERVAL	10	integer	timesteps	1.9.6
UNTAG_TYPE_2	.FALSE.	bool	-	1.9.6
UNTAG_TYPE_3	.FALSE.	bool	-	1.9.6
WIND_DIRECTION_FLUCTUATION_INTENSITY	0	float	degrees	1.9.4
WIND_FLUCTUATIONS	.FALSE.	bool	-	1.9.4

WIND_SPEED_FLUCTUATION_INTENSITY	0	float	mph	1.9.4
X_IGN(:)	0	float	coordinate	1.5.1
Y_IGN(:)	0	float	coordinate	1.5.1
USE_PYROMES	.FALSE.	bool	-	1.17
WSMFEFF_LOW_MULT	5.08E-03	float	conversion factor	1.9.2
WX_BILINEAR_INTERPOLATION	.FALSE.	bool	-	1.2
WX_BANDS_KEPT_IN_MEM	30	integer	bands	1.19
CLEAN_SCRATCH	.FALSE.	bool	-	1.7
HRR_ELLIPSE_ADJ	0.5	float	adjustment factor	1.13
	WUI			
BLDG_AREA_CONSTANT	20	float	meters squared	1.13
BLDG_SEPARATION_DIST_CONSTANT	10	float	meters	1.13
BLDG_NONBURNABLE_FRAC_CONSTANT	0	float	percent	1.13
BLDG_FOOTPRINT_FRAC_CONSTANT	1	float	percent	1.13
BLDG_FUEL_MODEL_CONSTANT	1	integer	fuel model	1.13
BLDG_SPREAD_MODEL_TYPE	1	integer	switch	1.13
USE_BLDG_SPREAD_MODEL	.FALSE.	bool	-	1.13
USE_CONSTANT_BLDG_SPREAD_MODEL_PARAMS	.TRUE.	bool	-	1.13
GLOBAL_HARDENING_FACTOR	1	float	float	
INTERFACE_MODEL_TYPE	1	integer	categorical	1.13
BANDTHICKNESS_WUI	5	integer	pixels	1.13
CRITICAL_HF_WUI	0	float	-	1.13
	SPOTTING			
GENERATION_MODEL	'RANDOM'	string	Categorical	1.11.1
SPOTTING_DISTANCE_MODEL	'UNIFORM'	string	Categorical	1.11.2
ACCUMULATION_MODEL	'LAGRANGIAN'	string	Categorical	1.11.3
IGNITION_MODEL	'DIRECT'	string	Categorical	1.11.4
CRITICAL_SPOTTING_FIRELINE_INTENSITY(:)	0	float	kW/m	1.11.1
CROWN_FIRE_SPOTTING_PERCENT	100	float	%	1.11.1
CROWN_FIRE_SPOTTING_PERCENT_MAX	100	float	%	1.11.7
CROWN_FIRE_SPOTTING_PERCENT_MIN	100	float	%	1.11.7
EMBER_GR	1.00E-03	float	$embers/m^2$	1.11.1

EMBER_SAMPLING_FACTOR	1	float	scale factor	1.11.1
ENABLE_SPOTTING	.FALSE.	bool	-	1.11
ENABLE_SURFACE_FIRE_SPOTTING	.FALSE.	bool	-	1.11.1
GLOBAL_SURFACE_FIRE_SPOTTING_PERCENT	0	float	%	1.11.1
GLOBAL_SURFACE_FIRE_SPOTTING_PERCENT_MAX	0	float	%	1.11.7
GLOBAL_SURFACE_FIRE_SPOTTING_PERCENT_MIN	0	float	%	1.11.7
MAX_SPOTTING_DISTANCE	0	float	meters	1.11.2
MEAN_SPOTTING_DIST	0	float	meters	1.11.2
MEAN_SPOTTING_DIST_MAX	0	float	meters	1.11.7
MEAN_SPOTTING_DIST_MIN	0	float	meters	1.11.7
MIN_SPOTTING_DISTANCE	0	float	meters	1.11.2
NEMBERS_MAX	1	integer	firebrands	1.11.1
NEMBERS_MAX_HI	1	integer	firebrands	1.11.7
NEMBERS_MAX_LO	1	integer	firebrands	1.11.7
NEMBERS_MIN	30	integer	firebrands	1.11.1
NEMBERS_MIN_HI	0	integer	firebrands	1.11.7
NEMBERS_MIN_LO	0	integer	firebrands	1.11.7
NORMALIZED_SPOTTING_DIST_VARIANCE	0	float	meters	1.11.2
NORMALIZED_SPOTTING_DIST_VARIANCE_MAX	0	float	meters	1.11.7
NORMALIZED_SPOTTING_DIST_VARIANCE_MIN	0	float	meters	1.11.7
PIGN	1	float	%	1.11.4
PIGN_MAX	1	float	%	1.11.7
PIGN_MIN	1	float	%	1.11.7
LOCAL_IGNITION_TIME	30	float	seconds	1.11.4
CELL_IGNITION_DELAY	100	float	seconds	1.11.4
SOURCE_FUEL_IGN_MULT(:)	1	float	scale factor	1.11.4
SPOTTING_DISTRIBUTION_TYPE	'LOGNORMAL'	string	Categorical	1.11.2
SPOT_FLIN_EXP	0.5	float	exponent	1.11.2
SPOT_FLIN_EXP_HI	0.5	float	exponent	1.11.7
SPOT_FLIN_EXP_LO	0.5	float	exponent	1.11.7
SPOT_WS_EXP	0.9	float	exponent	1.11.2
SPOT_WS_EXP_HI	0.9	float	exponent	1.11.7

SPOT_WS_EXP_LO	0.9	float	exponent	1.11.7
STOCHASTIC_SPOTTING	.FALSE.	bool	-	1.11.7
SURFACE_FIRE_SPOTTING_PERCENT(:)	100	float	%	1.11.1
SURFACE_FIRE_SPOTTING_PERCENT_MULT(:)	1	float	scale factor	1.11.4
TAU_EMBERGEN	6	float	seconds	1.11.1
P_EPS	0.01	float	PDF upper bound	1.11.2
USE_PHYSICAL_SPOTTING_DURATION	.FALSE.	bool	-	1.11.1
USE_CUSTOMIZED_PDF	.FALSE.	bool	-	1.11.2
USE_SUPERSEDED_SPOTTING	.TRUE.	bool	-	1.11
NO_SURFACE_FIRE	.FALSE.	bool	-	1.11
MU_CROSSWIND	0	float	meters	1.11.2
SIGMA_CROSSWIND	0	float	meters	1.11.2
MU_DOWNWIND	0	float	meters	1.11.2
SIGMA_DOWNWIND	0	float	meters	1.11.2
EMBER_GR_PER_MW_BLDG	10	float	float	1.11.1
EMBER_GR_PER_MW_VEGE	33.3	float	float	1.11.1
DIFF_WILDLAND_IGNITION	.FALSE.	bool	-	1.11.4
USE_EMBER_CONSUMPTION	.FALSE.	bool	-	1.11.5
USE_CROSSWIND_DISTRIBUTION	.FALSE.	bool	-	1.11.2
	SUPPRESSION			
B_SDI	1	float	scale factor	1.14
DT_EXTENDED_ATTACK	3600	float	seconds	1.14
ENABLE_EXTENDED_ATTACK	.FALSE.	bool	-	1.14
ENABLE_INITIAL_ATTACK	.FALSE.	bool	-	1.14
AREA_NO_CONTAINMENT_CHANGE	10000	float	acres per DT	1.14
INITIAL_ATTACK_TIME	1800	float	seconds	1.14
MAX_CONTAINMENT_PER_DAY	100	float	%	1.14
SDI_FACTOR	1	float	scale factor	1.14
USE_SDI	.FALSE.	bool	-	1.14
USE_SDI_LOG_FUNCTION	.FALSE.	bool	-	1.14
EXTENDED_ATTACK_MODEL	0	int	-	1.14
ENABLE_INDIRECT_ATTACK	.TRUE.	bool	-	1.14

EXTENDED_ATTACK_TIME	-1	float	seconds	1.14
INITIAL_CONTAINMENT_SHAPE_FACTOR	10	float	scale factor	1.14
AVAILABLE_SUPPRESSION_CAPACITY	2000	float	m/hr	1.14
FIRELINE_LENGTH_REF	15000	float	m	1.14
PCL_THRESHOLD	30	float	PCL units	1.14
FL_MAX_DIRECT_ATTACK	8	float	ft	1.14
SDI_MAX_DIRECT_ATTACK	100	float	SDI units	1.14
FIRE_LINE_THICKNESS	1	float	cells	1.14
DELTA_ROS	10	float	ft/min	1.14
DELTA_FL	2	float	ft	1.14
DELTA_SDI	10	float	SDI units	1.14
DELTA_PCL	10	float	PCL units	1.14
	CALIBRATION			
ADJUSTMENT_FACTORS_BY_PYROME	.FALSE.	bool	-	1.17
ADJUSTMENT_FACTORS_FILENAME	""	string	file	1.17
CALIBRATION_CONSTANTS_BY_PYROME	.FALSE.	bool		1.17
CALIBRATION_CONSTANTS_FILENAME	""	string	file	1.17
DURATION_MAX_DAYS	28	integer	days	1.17
DURATION_PDF_BY_PYROME	.FALSE.	bool	-	1.17
DURATION_PDF_FILENAME	""	string	file	1.17

Bibliography

[1] Development and structure of the Canadian Forest Fire Behavior Prediction System. Technical Report ST-X-3, Natural Resources Canada, Canadian Forestry Service, Ottawa, Canada, 1992.

[2] Frank A. Albini. Estimating wildfire behavior and effects. *Gen. Tech. Rep. INT-GTR-30. Ogden, UT: U.S. Department of Agriculture, Forest Service, Intermountain Forest and Range Experiment Station. 92 p.*, 30, 1976.

[3] Hal E. Anderson. *Predicting Wind-driven Wind Land Fire Size and Shape*. U.S. Department of Agriculture, Forest Service, Intermountain Forest and Range Experiment Station, 1983.

[4] Patricia L. Andrews. Modeling wind adjustment factor and midflame wind speed for Rothermel's surface fire spread model. Technical Report RMRS-GTR-266, U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station, Ft. Collins, CO, 2012.

[5] Patricia L. Andrews. The Rothermel surface fire spread model and associated developments: A comprehensive explanation. Technical Report RMRS-GTR-371, U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station, Ft. Collins, CO, 2018.

[6] Miguel G Cruz, Martin E Alexander, and Ronald H Wakimoto. Development and testing of models for predicting crown fire rate of spread in conifer forest stands. *Canadian Journal of Forest Research*, 35(7):1626–1639, July 2005.

[7] Jacques A. De Beer, Emily L. Dietz, Stanislav I. Stoliarov, and Michael J. Gollner. An empirical firebrand pile heat flux model. *Fire Safety Journal*, 141:104004, December 2023.

[8] Dougal Drysdale. *An Introduction to Fire Dynamics*. Wiley, August 2011.

[9] Mark Finney. FARSITE: a fire area simulator for fire managers. General Technical Report PSW-GTR-158, Forest Service, U.S. Department of Agriculture, Albany, CA: Pacific Southwest Research Station, February 1994.

[10] Mark A. Finney. FARSITE: Fire Area Simulator-model development and evaluation. Technical Report RMRS-RP-4, U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station, Ft. Collins, CO, 1998.

[11] M Hamada. On the rate of fire spread, disaster research. *Non-Life Insurance Rating Organisation Japan*, 1951.

[12] Keisuke Himoto and Takeyoshi Tanaka. Transport of disk-shaped firebrands in a turbulent boundary layer. In *Fire Safety Science Proceedings of the Eighth International Symposium*, volume 8, pages 433–444, Beijing, China, 2005. International Assosiation for Fire Safety Science.

[13] Kelvin G. Hirsch, Paul N. Corey, and David L. Martell. Using Expert Judgment to Model Initial Attack Fire Crew Effectiveness. *Forest Science*, 44(4):539–549, November 1998.

[14] Nikolaos Kalogeropoulos. *Wildfire Simulations to Protect Rural Communities and Avoid Dire Evacuations*. PhD Thesis, Imperial College London, London, 2025.

[15] Nikolaos Kalogeropoulos, Alexander Castagna, and Guillermo Rein. A Gaussian Dispersion Model for Haze Spread with Downwind Terrain Effects. *Submitted to the Journal of Loss Prevention in the Process Industry.*, 2025.

[16] Chris Lautenberger. Wildland fire modeling with an Eulerian level set method and automated calibration. *Fire Safety Journal*, 62:289–298, November 2013.

[17] Roger D. Ottmar, David V. Sandberg, Cynthia L. Riccardi, and Susan J. Prichard. An overview of the Fuel Characteristic Classification System — Quantifying, classifying, and creating fuelbeds for resource planningThis article is one of a selection of papers published in the Special Forum on the Fuel Characteristic Classification System. *Canadian Journal of Forest Research*, 37(12):2383–2393, December 2007.

[18] Dwi M.J. Purnomo, Yiren Qin, Maria Theodori, Maryam Zamanialaei, Chris Lautenberger, Arnaud Trouvé, and Michael Gollner. Reconstructing modes of destruction in wildland–urban interface fires using a semi-physical level-set model. *Proceedings of the Combustion Institute*, 40(1-4), January 2024.

[19] Yiren Qin. *A Physics-Based Eulerian Framework for Modeling Firebrand Showering in Regional-Scale Wildland and Wildland-Urban Interface Fire Simulations - ProQuest*. PhD thesis, November 2025.

[20] Ronald G Rehm and Randall J McDermott. Fire-front propagation using the level set method. Technical Report NBS TN 1611, National Bureau of Standards, Gaithersburg, MD, 2009. Edition: 0.

[21] Guillermo Rein. Smouldering Combustion Phenomena in Science and Technology. 1, 2009.

[22] F. Reisen, C. P. Meyer, C. J. Weston, and L. Volkova. Ground-Based Field Measurements of $PM_{2.5}$ Emission Factors From Flaming and Smoldering Combustion in Eucalypt Forests. *Journal of Geophysical Research: Atmospheres*, 123(15):8301–8314, August 2018.

[23] Richard C. Rothermel. A mathematical model for predicting fire spread in wildland fuels. Research Paper INT-115, US Forest Service, Odgen, Utah, 1972.

[24] N. Sardoy, J. L. Consalvi, A. Kaiss, A. C. Fernandez-Pello, and B. Porterie. Numerical study of ground-level distribution of firebrands generated by line fires. *Combustion and Flame*, 154(3):478–488, August 2008.

[25] Joe Scott. Comparison of Crown Fire Modeling Systems Used in Three Fire Management Applications. *USDA Forest Service - Research Paper RMRS-RP*, May 2006.

[26] Joe H. Scott. Nomographs for estimating surface fire behavior characteristics. *Gen. Tech. Rep. RMRS-GTR-192. Fort Collins, CO: U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station. 119 p.*, 192, 2007.

[27] J A Sethian. A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*, 93(4):1591–1595, February 1996.

[28] Cordy Tymstra, R.W. Bryce, B.M. Wotton, S.W. Taylor, and O.B. Armitage. Development and structure of Prometheus: the Canadian Wildland Fire Growth Simulation Model. Technical Report NOR-X-417, Northern Forestry Centre, Edmonton, 2010. OCLC: 499432152.

[29] Shawn Urbanski. Wildland fire emissions, carbon, and climate: Emission factors. *Forest Ecology and Management*, 317:51–60, April 2014.

[30] C. E. Van Wagner. *Development and structure of the Canadian Forest Fire Weather Index System.* Number 35 in Forestry technical report. Minister of Supply and Services Canada, Ottawa, 1987.

[31] C. E. Van Wagner. Conditions for the start and spread of crown fire. *Canadian Journal of Forest Research*, 7(1):23–34, March 1977.

[32] B.M. Wotton, M.E. Alexander, and S.W. Taylor. Updates and revisions to the 1992 Canadian Forest Fire Behavior prediCtion system. Technical Report GLC-X-10, Natural Resources Canada, Sault Ste. Marie, Ontario, Canada, 2009.