An Efficient Method to Simulate Wildfire Propagation Using Irregular Grids

Conor Hackett[1], Rafael de Andrade Moral[2], Gourav Misra[3], Tim McCarthy[3], Charles Markham[4]

[1]Hamilton Institute, National University of Ireland Maynooth, Kildare, W23 A3HY, Ireland.

[2]Department of Mathematics and Statistics, National University of Ireland Maynooth, Kildare, Ireland.

[3]National Centre for Geocomputation, National University of Ireland Maynooth, Kildare, W23 NPY6, Ireland.

[4]Computer Science Department, National University of Ireland Maynooth, Kildare, W23 A3HY, Ireland.

Correspondence to: Conor Hackett (conor.hackett.2018@mumail.ie)

Abstract. Climate change and land-use changes are projected to make wildfires more frequent and intense, with a global increase of extreme fires of up to 14 % by 2030, 27 % by the end of 2050 and 57 % by the end of the century (United Nations Environment Programme & GRID-Arendal, 2022). These values were forecasted by a Representative Concentration Pathway with a radiative forcing value of six (W/m^2) which represents the energy imbalance in the Earth's energy system caused by greenhouse gases and other factors, in the year 2100. This latest information has increased interest of how the large scale, often catastrophic, events can be reduced and more effectively managed. One critical area revolves around real-time fire line prediction and how resources can be better deployed to reduce the propagation of wildfires. In this paper a novel software platform called the Irregular Grid Software (IGS) was developed which allows the simulation of wildfires on a configurable grid using mathematical models for fire propagation. The aim of IGS was to explore computational differences between different grid types and a comparison to preexisting software while producing outputs of an acceptable similarity to that preexisting software. The configurable grid was built using a Voronoi diagram where a fire can spread between polygons, propagating throughout the grid. The configurable grid allows cross comparison of both regular grids such as square, hexagonal, triangular, and irregular grids such as a randomly seeded Voronoi diagram and a newly developed flammable resolution grid (FRG). The FRG was adapted to focus attention on areas of higher importance which provides greater precision at the cost of extra computing time. The irregular grid approach and ForeFire, an existing industry standard program were compared. The comparison included simulations of wildfires located in the Wicklow Mountains, in Ireland, a region used by fire services for exercises. The performance of the grid-based techniques was examined using a set of experiments to characterise the model's response to key factors such as wind, elevation, and fuel type. The objective of this paper was to compare the various grid types on the metrics of similarity with ForeFire and computational time, while also comparing the FRG to ForeFire on the same metrics with multiple sample wildfires. From the sample wildfires tested in this paper the results show that the IGS runs on average 34 times quicker than ForeFire on the same computing platform while retaining an average result similarity of 80% with ForeFire.

Keywords

Wildfire, Forest Fire, Bushfire, Simulation, Modelling, Rothermel Model, Irregular Grid, Voronoi Diagrams, Cellular Automata, ForeFire, DecaMap

1 Introduction

All fires, including wildfires, require four components to support combustion (Helene et al., 2019). These are fuel (vegetation), an oxidising agent (oxygen in the air), heat (initiates the process) and a chemical chain reaction (to sustain combustion). Combustion is a chemical process in which fuel and oxidiser combine to form heat greater than the fuel and oxygen mixture's flash point, if there is sufficient heat generated to overcome heat loss, combustion is sustained in a chain reaction. Together they make up the "fire tetrahedron"; a fire will cease to exist if any component is diminished within the tetrahedron (Figure 1).

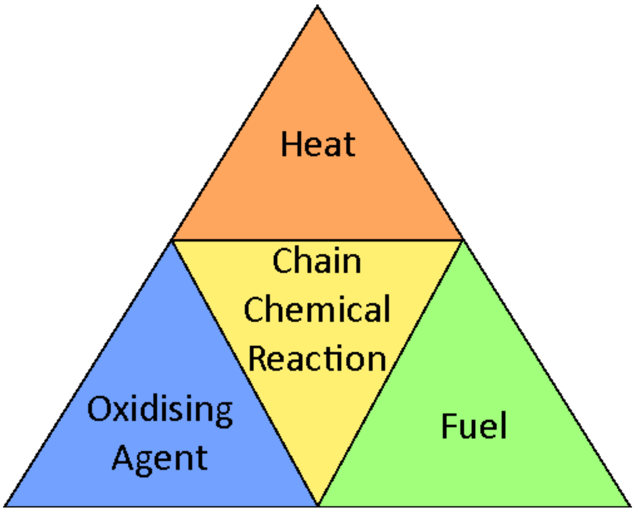

Figure 1: A 2D net of the "fire tetrahedron", which shows all four components required to support combustion.

A wildfire is a potentially destructive fire that burns throughout an area, this includes forest fires and bushfires (Haghani et al., 2022). Usually, they begin in remote areas where there is a higher density of combustible vegetation such as grasslands and peatlands. They can sometimes threaten urban areas (Park et al., 2023). Wildfires can be caused by both natural events such as lightning, and through human-caused actions such as arson and farming techniques like slash-and-burn (Jiao et al., 2023; dos Reis et al., 2021). The frequency of wildfires has increased within forested extratropical and boreal regions in recent years, this is most likely due to climate change creating drier terrain, allowing fires to burn more easily (Janssen et al., 2023; Abatzoglou et al., 2018; Halofsky et al., 2020). This however varies regionally where factors such as wildland-urban interface, land use changes and fire exclusion policies can have an impact on the number of wildfires (Beltrán-Marcos et al., 2023; Piñol et al., 2005).

The devastation of wildfires can be measured in different ways such as fatalities, ecological and economic damage. In California, USA, and Australia a small subset of wildfires that burn in terrain satisfying the correct conditions can potentially grow to be very large fast burning fires which can present an increased risk to life (Keeley and Syphard, 2021; Blanchi et al., 2014). Wildfires impact the ecosystem by destroying some of the flora and displacing or killing the fauna (Kala, 2023).When wildfires burn through terrain, they release Carbon dioxide, further contributing to an increase in greenhouse gas emissions (Xue et al., 2024; Jones et al., 2019).

The economic impact of wildfires includes damage to the agricultural and forestry sectors (Meier et al., 2023). When a wildfire spreads to an urban area it can also cause considerable damage to town infrastructure (Park et al., 2023).

Countries that normally have a wet climate such as Ireland, may also start to have an increased number of wildfires due to warmer temperatures, likely caused by climate change (McElwain and Sweeney, 2003; Boegelsack et al., 2018; Benyon et al., 2023). Wildfires have had an ecological impact in Ireland where they tend to be smaller, slower burning fires occurring mainly in bogs (peatlands), which are home to many rare species of both flora and fauna (Prat-Guitart et al., 2019). The high abundance of gorse (*Ulex europaeus*) in bogs makes them a common place for wildfires to occur as the plant is easily ignitable.

Even though wildfires that burn in Ireland, Australia and the USA tend to have different burning behaviours, it is still possible to model them all as the underlying physics involved remain consistent. However, the parameters characterising fuel properties, terrain and weather conditions need to be changed to represent the surrounding environment.

2 Overview of Wildfire Propagation Models

With the increasing severity and frequency of wildfires in forested extratropical and boreal regions, the ability to model and predict wildfire propagation has become an invaluable asset to planners and firefighters (Penney et al., 2019). Planners can use this information to forecast and prevent fires, while firefighters can also use this information to find the optimal locations to apply interventions. These interventions include construction of firebreaks, controlled burning, or application of water using fire engines or Helicopters equipped with a Bambi-Bucket (SFI-Defence Organisation Innovation Challenge, Challenge 1, 2024).

There are three main branches of wildfire modelling, which include statistical, empirical, and physical models (Weber, 1991). Statistical models are built on a statistical description of wildfires found by observing sample fires and are less focused in the thermodynamics of fire. The McArthur Forest Fire Danger Index is a statistical wildfire model (McArthur, 1966; Noble et al., 1980). The statical model was developed by only sampling fires burning dry grassland and forest litter which means it needs to be used with caution on other fuel types. The model takes as input, variables such temperature of curing for the fuel, air temperature, relative humidity, and wind velocity to produce a value representing the fire danger index. This value can then be converted into an estimated rate of spread for a wildfire.

Empirical models for wildfire spread are built upon the principles of conservation of energy but do not differentiate between the various modes of head transfer. An empirical model was used to calculate fire spread through porous fuels in a fuel bed (Frandsen, 1971). This model calculates the rate of spread by finding the heat generated from combustion of existing fuel and spreading this heat using the principles of the conservation of energy into the surrounding fuel, until that fuel reaches its flashpoint and ignites.

Physical models do differentiate between the various modes of heat transfer and are deep-rooted in fundamental physics and mathematics of combustion. A physical model was developed to find the rate of spread between two particles in a fuel bed (Fons, 1946). To calculate the rate of spread for the fire, this model found the time it took for fire to spread the distance from one particle to another. This method also used the principles of the conservation of energy but this time it incorporated all three forms of heat transfer: conduction, convection, and approximating radiation.

One of the most cited wildfire models is the Rothermel Model (Rothermel, 1972; Andrews, 2018). It was developed by Richard C. Rothermel in 1972 to help forest managers predict the behaviour of wildfires. The Rothermel model represents one approach to wildfire modelling as a physics-based, semi-empirical model formed using the conservation of energy that underpins many operational fire modelling tools. The Rothermel model was based on the laws of heat transfer but was developed using experimental data from constructed fuel beds.

The Rothermel model takes as input multiple environmental, fuel and fuel bed factors to produce an estimated rate of spread as an output Eq. (1). In the Rothermel model the numerator measures total heat transferred to neighbouring fuel while the denominator measures energy required to ignite neighbouring fuel and R is the spread rate of the fire. It does this on a 1-dimensional line, but this can be expanded to more dimensions by measuring R on separate lines along each direction of interest. An easy way to interpret this is with a section of burning terrain acting as a heat source, while the fuel and fuel bed of the neighbouring terrain acts as a heat sink. If there is a surplus amount of heat produced to ignite the neighbouring terrain the fire will spread (Figure 2). The Rothermel model is normally presented using the United States customary units. In this paper these values were converted into SI units within the fire simulations developed, this is the same as in ForeFire to simplify comparison in this paper. Another approach that could be considered would be to use the Rothermel equation reformulated in SI units (Wilson, 1980). The original Rothermel equation that was used in this paper is written as:

$$R = \frac{I_R \xi (1 + \Phi_w + \Phi_s)}{\rho_b \varepsilon Q_{ig}} \tag{1}$$

where $R \geq 0$ is the rate of spread in m/s (ft/min), $I_R \geq 0$ is the reaction intensity in J/m^2/s (Btu/ft^2/min), $0 \leq \xi \leq 1$ is the propagating flux ratio, $\Phi_w \geq 0$ is the wind factor, $\Phi_s \geq 0$ is the slope factor, $\rho_b > 0$ is the bulk density in kg/m^3 (lb/ft^3), $\varepsilon > 0$ is the effective heating number and $Q_{ig} > 0$ is the heat of preignition in J/kg (Btu/lb).

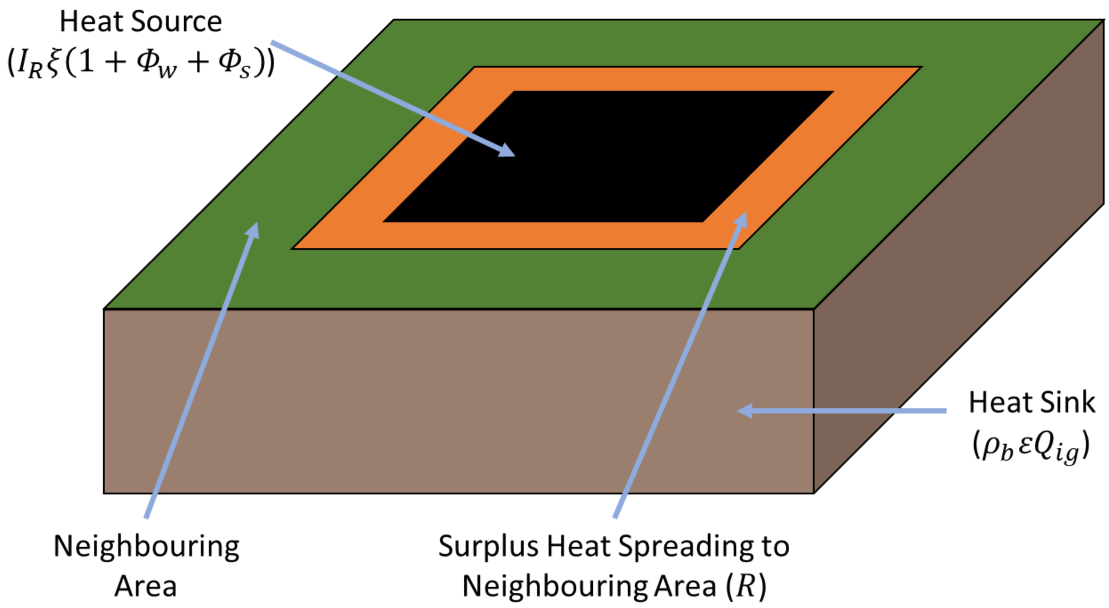
Heat Source
$(I_R\xi(1+\Phi_w+\Phi_s))$
Heat Sink
$(\rho_b\varepsilon Q_{ig})$
Neighbouring Area
Surplus Heat Spreading to Neighbouring Area (R)

Figure 2: Example of how the Rothermel model functions, where the heat source is the location of the fire (where heat is being generated from combustion), the heat sink absorbs heat from the heat source slowing down propagation, and the surplus heat is used to spread fire to the neighbouring area.

The Rothermel model is used as an optional fire propagation model in platforms such as Cell2Fire and ForeFire. There are three types of propagation models: cellular automata, cellular and continuous. Cellular automata models are restricted to a grid and a strict set of rules, where each cell is in a finite number of possible states and the state of one cell can influence the state of its neighbouring cells. Cellular models are also restricted to a grid, but the state of a cell is continuous and not confined to only influencing its neighbours (San Martin and Torres, 2023). Continuous models propagate over continuous space and therefore are not restricted to a grid or finite states.

Cell2Fire is a program that utilises cellular automata to simulate wildfire spread (Pais et al., 2021). Traditionally cellular automata have rules that express how the state of one cell interacts with its neighbouring cells. Cell2Fire can be given a fire spread model, a starting fire and information about the environment to begin simulating fire spread. As Cell2Fire is a program built on cellular automata it restricts the spread of fire to a regularly shaped square grid using the Moore neighbourhood where each cell has 8 neighbours (corner neighbouring cells inclusive) (Małecki, 2017). Cell2Fire generates the common ellipse shape of a wildfire for each cell based on the rate of spread. Once any point on the ellipse touches the centroid of a neighbouring cell that cell will also begin generating its own ellipse to calculate rate of spread. The uniform grid presents trade-off between resolution and computation time. Low resolution grids are fast but have the potential to be less precise. High resolution grids are computationally intensive. This will be discussed in further detail later in the paper.

ForeFire is another program that simulates wildfire spread but does this over a continuous space instead of the discrete space used in cellular automata models (Filippi et al., 2014). ForeFire uses a fire model, such as the Rothermel model to compute propagation. It also takes spatial environmental data over a selected region of terrain as an input. The environmental data contains wind data in the form of zonal (west to east) and meridian (south to north) wind speeds, elevation data regarding the terrain and a numerical index representing different land cover types for the terrain. All data inputted into ForeFire is in the form of metric units, if the fire spread model required imperial or United States customary units (such as the Rothermel model), they need to be converted before calculating fire spread and converted back to metric units once fire spread has been calculated. ForeFire is seeded with information about the fire's starting location, the spatial resolution, and the duration of the simulation. During the simulation ForeFire calculates fire spread using the current fire location and environmental inputs. The increased fire spread perimeter is comprised of nodes called markers that ForeFire uses to continue spreading for the next iteration. Each marker has a propagation vector representing its movement direction and speed of propagation. The speed at which the marker moves at is determined by the fire spread model. If the distance between markers is greater than a selected spatial resolution, more markers will be created and redistributed along the fire front. If the distance between markers is less than quarter of the spatial resolution, they will be merged into one marker (Figure 3). For this paper the spatial resolution of ForeFire was set to the same as its default example's resolution of 30 m^2. On completion the software returns, the final set of coordinates (known as markers) describing the location of the fire front. The use of markers instead of a grid allows for increased resolution as their position is on a continuous plane, limited by the resolution of the input data and their set spatial resolution. However, the continual placing, removing, and moving of markers add extra computational burden on the simulation.

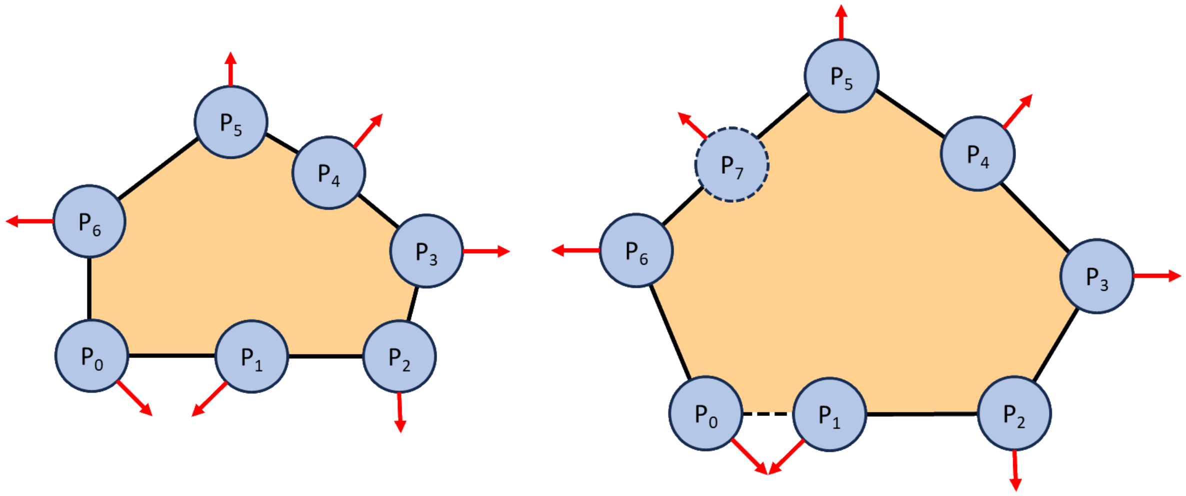

Figure 3: Example of ForeFire markers at timestep 0 (left) and timestep 1 (right). In the example the fire line (black line), markers (blue circles), direction of spread for markers (red arrows) and where the fire has been (orange) are all shown. In this example markers P_0 and P_1 will combine into one marker as the distance between them is quarter the spatial resolution. Marker P_7 will be created as the distance between markers P_5 and P_6 is greater than the spatial resolution.

The technique of simulating wildfires using the Rothermel model in this paper follows a similar methodology as was used to simulate wildfires using cellular automata (Zhang et al., 2021). In that paper wildfires progressed from a cell's centroid to neighbouring cell centroids at rate determined by the Rothermel model. Once a wildfire had reached a neighbouring cell's centroid that cell would also being spreading the wildfire to its neighbouring cells. Similar techniques will be used in the methods section of this paper.

3 Methods

This paper's novel approach, named the Irregular Grid Software (IGS), was purposely designed to use the same input parametrisation as ForeFire to allow for a fair comparison between both platforms.

3.1 Resources

A satellite image of the region around Lough Dan, County Wicklow, Ireland, was used as input data for both ForeFire and the IGS. A random forest machine learning model was applied to the satellite data to produce a land cover map. Each pixel on the landcover map referenced an associated land cover type including pastures, sparsely vegetated areas, mixed forests, moors and heathland, urban fabric, water bodies, clouds, and cloud shadows (Appendix A).

Each type of land cover was mapped to a set of physical attributes that were contained in a lookup table used by both programs. ForeFire contains one of these files called a fuel file, by default which was also used by the IGS (Fuels.ff Fuel Attribute Table, 2024). ForeFire's fuel file was indexed based on Corine Land Cover classes (Home :: Corine Land Cover classes, 2024). The fuels file contained values for fuel particle density (kg/m^3), fuel particle moisture content, fuel particle surface area to volume ratio (m^{-1}), fuel height (m), the oven-dry fuel load (kg/m^2) and fuel particle low heat content (J/kg) for the different land cover types. For both the IGS and ForeFire the Rothermel model had no mineral damping, and the moisture of extinction was set to 0.3 for all fuels. Each of the listed fuel properties were required to run the Rothermel model. The land cover map had two additional fuel types for when it was not possible to identify the land type due to clouds or shadows. These fuel types were not present in the simulation area, as it was cloud free, so they did not affect either program.

The Sentinel-2 satellite data and a geographic information software, Snap Desktop's elevation band generator was used to produce an elevation map of the area (Copernicus Data Space Ecosystem, 2024; Download - STEP, 2024). The elevation map gives the height in meters of elevation above sea level for each pixel in the satellite data.

For experimental simulation, a set of different wind speeds and directions were selected to allow a comparison between ForeFire and the IGS under the same conditions. However, an API developed also allows software such as Windy to provide live, forecasted, and historical wind data for simulations (Windy, 2024). Wind data was then split into zonal (west to east) and meridian (south to north) wind speeds which are mapped at each pixel across the satellite data.

ForeFire contained a tool that allowed the land cover, elevation, wind data and metadata to be combined into a single NetCDF file (Tools, 2024; Network Common Data Form (NetCDF), 2024). This same file was then used by both ForeFire and the IGS.

3.2 Simulating Wildfire Propagation Using the IGS

The IGS was a Python program with the aim of generating a grid-based fire spread model that could be compared to ForeFire. The use of a grid with static points allows a model to compute fire spread without having to continually move and add markers during the simulation as found in ForeFire.

A Voronoi diagram was used to create an irregular grid. They have been used to simulate the geographic spread of disease in the past (Hackett et al., 2021). A Voronoi diagram takes in a set of points called sites. From these sites it generates edges (lines) located equidistant between itself and other sites, perpendicular to the direction between both sites. For the length of a particular edge the two sites that are on either side of it are the two sites that are closest to it, therefore once the distance between the edge is closer to a third different site, that edge stops. This creates contiguous polygons where every point within a polygon's perimeter is closer to that polygon's site than any other polygon's site (Figure 4). This means that each edge separating polygons is equidistant to both polygons' sites. The Voronoi diagram was generated using the efficient Fortune's algorithm implemented in the Foronoi (not spelled Voronoi) Python library (foronoi, 2024; Fortune, 1987). One of the biggest advantages of using a Voronoi diagram is the ability to create irregular simple contiguous shapes which allows for efficient computation of fire spread between polygons.

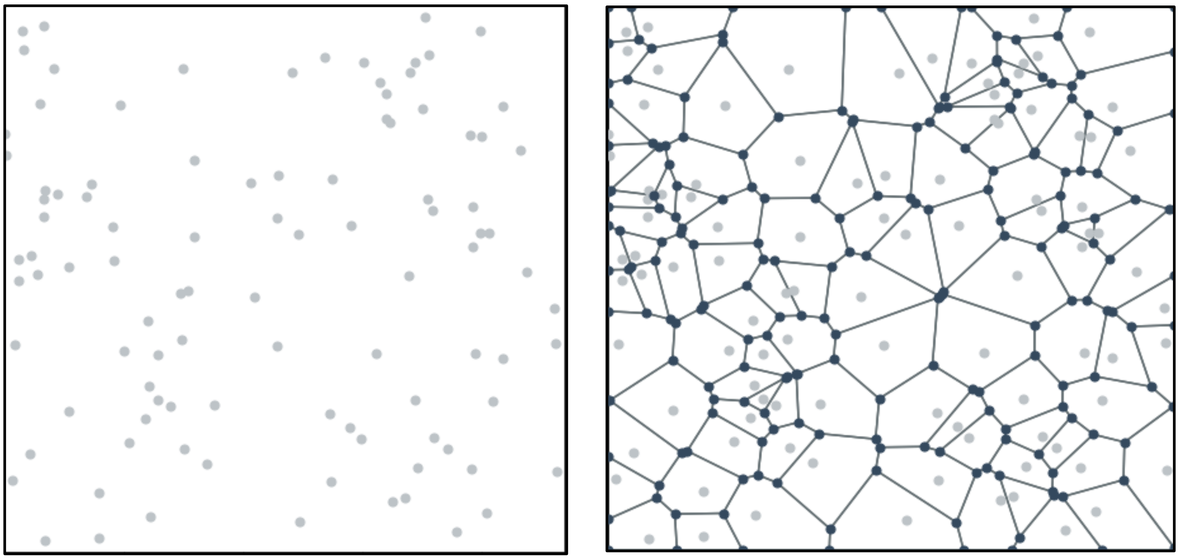

Figure 4: Positions of randomly placed sites (left) and the resulting generated Voronoi diagram of polygons (right).

The NetCDF file containing input data, created using ForeFire, is imported into the IGS using the Snappy Python library (How to use the SNAP API from Python, 2024). Snappy allowed the NetCDF data to be read into Python arrays used for computation. The generated Voronoi diagram was overlayed on top of the NetCDF data. The elevation, wind, and fuel values (per fuel index) were recorded for every pixel in all polygons of the Voronoi diagram. This was done using the scan-line polygon fill algorithm to extract environmental data from the pixels contained within the polygon (Al-Rawi, 2014). The mean elevation, wind and fuel values of each polygon were then saved.

The IGS starts with a fire located at the site of a Voronoi polygon. The propagation of this fire towards the site of each bounding polygon is modelled using the Rothermel model. When the fire reaches the site in the bounding polygon, the fire then progresses towards the new neighbouring polygons from that bounding polygon. Fires may propagate towards each other simultaneously. Fire progress is recorded as a ratio of how far it has propagated towards neighbouring polygon sites and the total distance between the two polygon sites. This is known as the propagation ratio. When the Rothermel model is run, the propagation ratio from the ignited polygon to its neighbours will increase if the rate of spread is positive (Figure 5) Eq. (2). When the propagation ratio reaches 1, the neighbouring polygon will become ignited and will begin also spreading fire as this represents the fire spreading the entire distance between both polygon sites. The propagation ratio is calculated as follows:

$$\left(P_{ij}\right)_{t+dT} = \frac{\left(\left(P_{ij}\right)_t d_{ij}\right) + \left(R_{ij} dT\right)}{d_{ij}} \tag{2}$$

where i is an ignited polygon, j is a polygon that neighbours the ignited polygon, t is the current time (s), $dT > 0$ is the timestep (s), $0 \leq (P_{ij})_{t+dT} \leq 1$ is the propagation ratio between polygons i and j at time $t + dT$, $0 \leq \left(P_{ij}\right)_t \leq 1$ is the propagation ratio between polygons i and j at time t, $R_{ij} \geq 0$ is the rate of spread between polygons i and j calculated by the Rothermel model (m/s) and $d_{ij} > 0$ is the distance between the sites of polygons i and j (m).

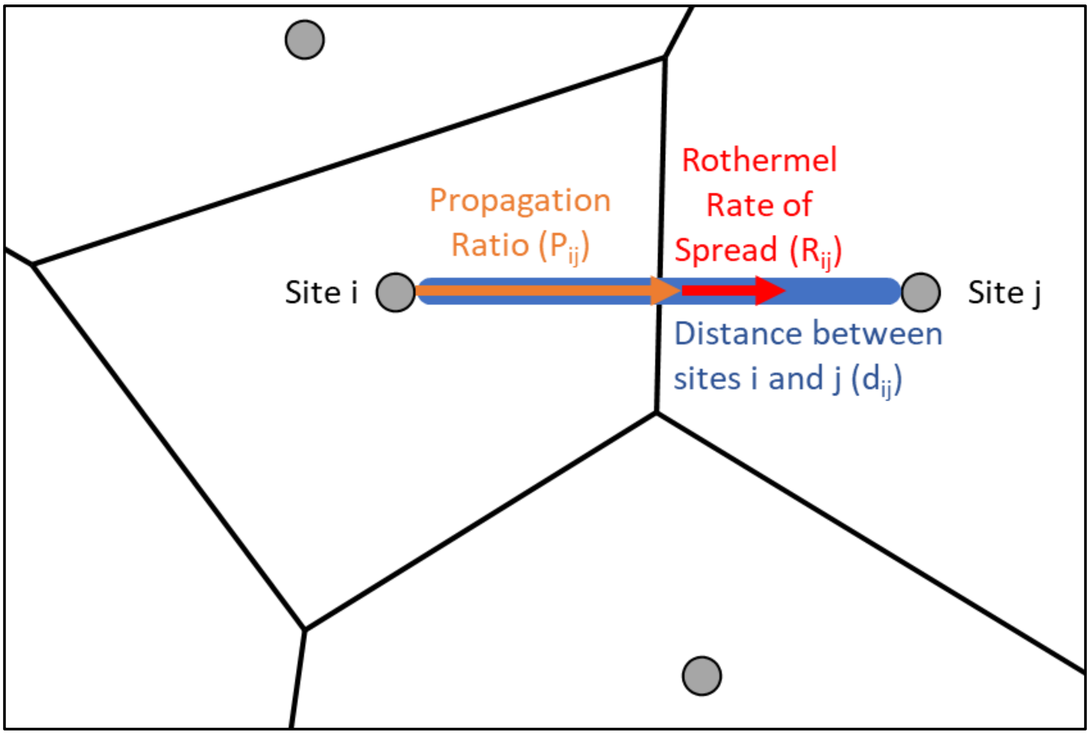

Figure 5: Example of the fire spreading from polygon i to j. Where the blue line represents the total distance (d_{ij}) between sites i and j (m), the orange arrow represents the propagation ratio (P_{ij}) of how far fire has spread from site i to j and the red arrow represents the fire's rate of spread determined by the Rothermel model (R_{ij}) from site i to j (m/s).

The Rothermel model requires data describing the fuel, the terrain (slope) and wind. Each polygon has fuel data associated with it, during fire spread the fuel data of the polygon that the fire is spreading to is used. If fuel data from the polygon the fire is spreading from is used, then fire would spread from that polygon to all its neighbours at an equal rate in conditions where there is no slope or wind. A weighted fraction between the two polygons also was not used as it would create additional computational time for simulating a wildfire. Therefore, the fuel properties of the polygon the fire is spreading to was selected. The slope is found by getting the difference in elevation between the neighbouring polygon and ignited polygon which is then divided by the distance between the polygon sites (Figure 6):

$$S_{ij} = \frac{H_j - H_i}{d_{ij}} \tag{3}$$

where i is the ignited polygon, j is the neighbouring polygon, S_{ij} is the slope from i to j, H_j is the mean elevation of polygon j (m), H_i is the mean elevation of polygon i (m) and $d_{ij} > 0$ is the flat distance between the polygon sites i and j (m).

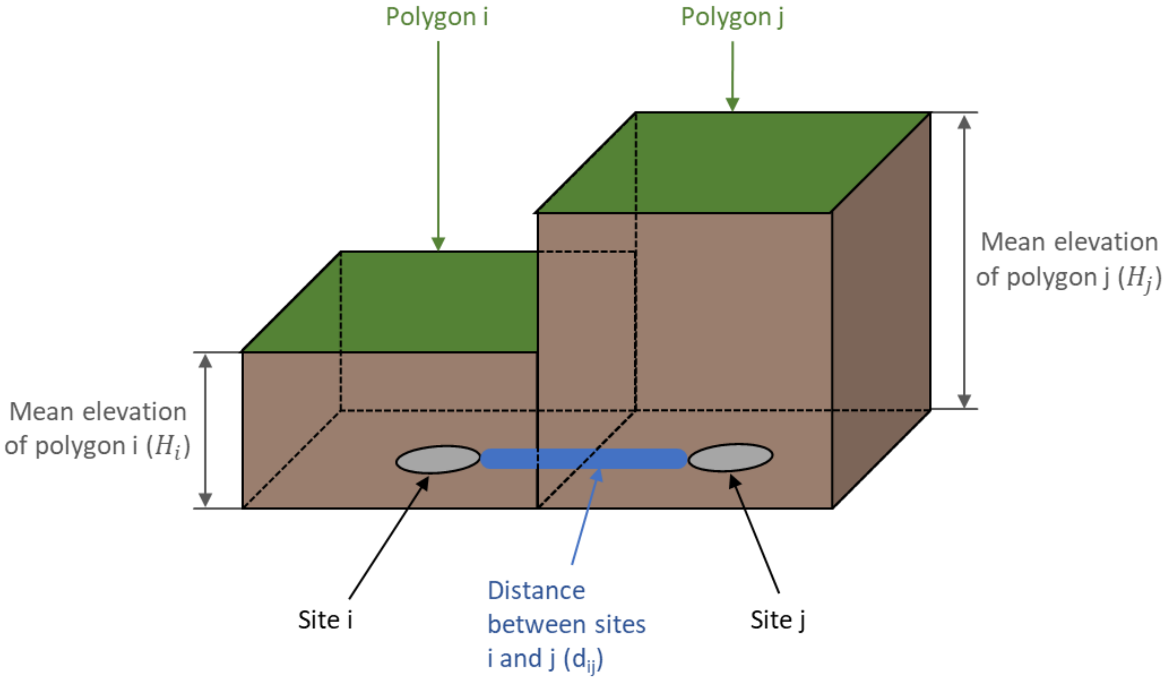
Polygon i
Polygon j
Mean elevation of polygon j (H_j)
Mean elevation of polygon i (H_i)
Site i
Distance between sites i and j (d_{ij})
Site j

Figure 6: Variables used in Eq. (3) to determine the slope from polygon i to polygon j.

The normal wind value is used in the Rothermel model, it is found by getting the dot product between the wind vector and the unit vector describing the ignited polygon and neighbouring polygon sites' coordinates (Figure 7):

$$\left(W_{ij}\right)_n = \begin{bmatrix} \left(\boldsymbol{W_j}\right)_{\boldsymbol{z}} \\ \left(\boldsymbol{W_j}\right)_{\boldsymbol{m}} \end{bmatrix} \cdot \frac{\vec{ij}}{\|\vec{ij}\|} \tag{4}$$

where i is the ignited polygon's site, j is the neighbouring polygon's site, $\left(W_{ij}\right)_n$ is the normal wind from i to j (m/s), $\left(W_j\right)_z$ is the mean zonal wind speed of j (m/s), $\left(W_j\right)_m$ is the mean meridian wind speed of j (m/s) and $\overrightarrow{ij}$ is the vector from polygon site i to polygon site j.

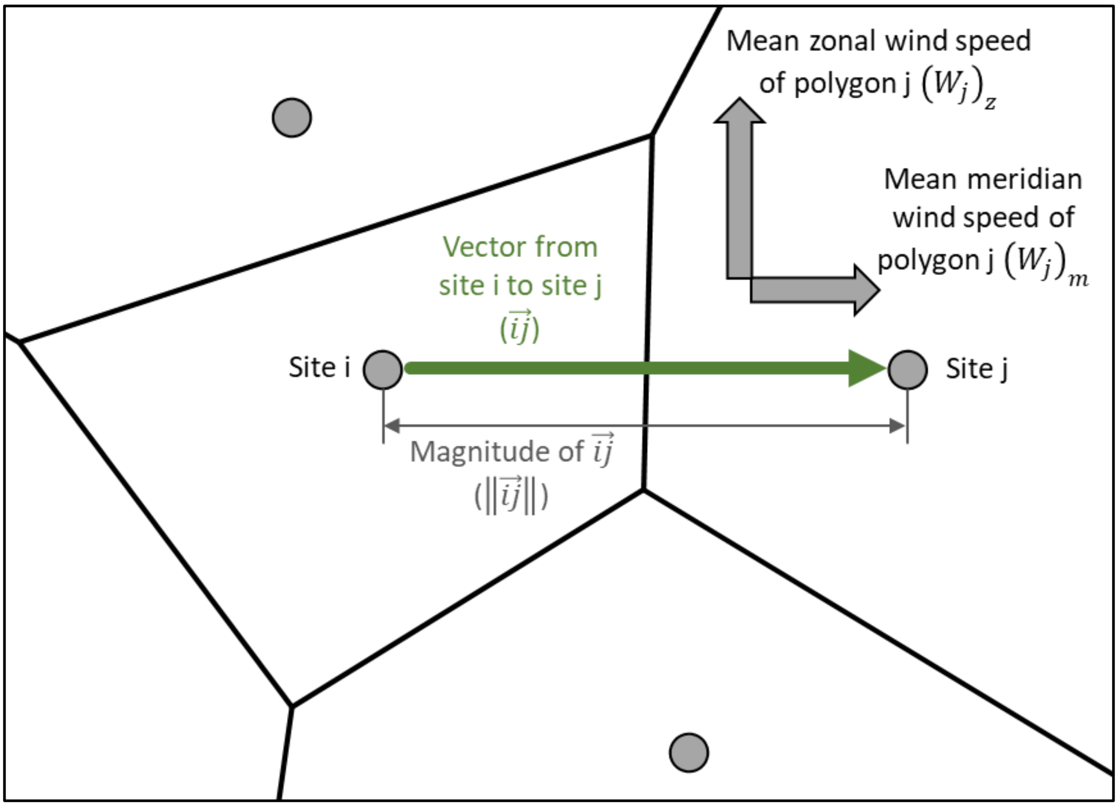
Mean zonal wind speed of polygon j $\left(W_j\right)_z$
Mean meridian wind speed of polygon j $\left(W_j\right)_m$
Vector from site i to site j ($\overrightarrow{ij}$)
Site i
Site j
Magnitude of $\overrightarrow{ij}$ ($\|\overrightarrow{ij}\|$)

Figure 7: Finding the normal wind for fire spreading from site i to j by getting the dot product of both the mean zonal and meridian wind speed properties of polygon j and the unit vector of spread from site i to j.

A visualisation was created to show the spatial distribution of the polygons and evolution of the fire line. The polygons were rendered in distinct colours to represent their states. Polygons rendered red represented polygons that were currently on fire, black represented completely burnt, clear without colour represent polygons that had not caught fire yet and green represented the wildfire source polygon(s) where the fire started. The colour of a polygon was determined from a combination of the masses in the states mentioned. A 50% transparency was used to allow the underlying map data to be seen (Figure 8).

The produced Voronoi Diagram was then scaled and overlayed on top of the satellite image so each pixel coordinate in the Voronoi Diagram would correspond to a pixel on the satellite image. The area of the polygon was then found in pixel coordinates and converted to an area in m^2. For fire line calculations, the oven dry fuel load per m^2 was sourced from the fuel file and used to estimate the total fuel per polygon (kg). The total fuel in the polygon could then be split into flammable vegetation, currently ignited vegetation and vegetation that had been completely burnt, written as:

$$V_i = N_i - (I_i + B_i) \tag{5}$$

where i is a polygon, $N_i \geq 0$ is the total amount of fuel in polygon i (kg), $V_i \geq 0$ is the amount of remaining flammable vegetation in polygon i (kg), $I_i \geq 0$ is the amount of fuel in polygon i currently ignited (kg) and $B_i \geq 0$ is the amount of fuel in polygon i that has burnt (kg). To estimate the mass of a polygon that was currently on fire, the propagation ratios from the ignited polygon and its neighbours were found. The total propagation ratios of fire that had spread within the ignited polygon only, and the total half distances between the ignited and neighbouring polygons were calculated. This value was then multiplied by the total amount of fuel in the ignited polygon, the amount of completely burnt fuel was then subtracted to find the amount of fuel that was currently on fire in that ignited polygon, as written below:

$$q_{ij} = \left(P_{ij} d_{ij}\right) + \left(d_{ij}\left(P_{ji} - 0.5\right)\right) \tag{6}$$

$$I_i = N_i \frac{\sum_{j=1}^{n_i} \min\left(q_{ij}, \frac{d_{ij}}{2}\right)}{\sum_{j=1}^{n_i} \frac{d_{ij}}{2}} - B_i \tag{7}$$

where i is a polygon , j is a polygon neighbouring i, $q_{ij} \geq 0$ is the total distance fire has spread within polygon i between it and polygon j (m), $N_i \geq 0$ is the total amount of fuel in polygon i (kg), $I_i \geq 0$ is the amount of fuel in polygon i currently ignited (kg), $B_i \geq 0$ is the amount of fuel in polygon i that has burnt (kg), $0 \leq P_{ij} \leq 1$ is the propagation ratio between polygons i and j, $n_i \geq 0$ is the number of neighbours polygon i has and $d_{ij} > 0$ is the distance between polygons i and j (m). This was to ensure only fire that had spread within the polygon was measured as any other spread in neighbouring polygons was used to find the mass of their polygon that was on fire instead.

The reaction intensity parameter from the Rothermel model found the energy produced per unit area per time step. Using the equation for the reaction intensity, the energy produced per unit mass per unit time step $\left(E_{ij}\right)$ can be evaluated. This is done by removing the net fuel load $(w_n)_j$ from the equation Eq. (8) to give Eq. (9) which as previously stated uses United States customary units. $E_{ij}$was found using the fuel properties from neighbouring polygons and was converted to metric units. We have:

$$(I_R)_{ij} = \Gamma'_j \ (w_n)_j h_j \eta_{M_j} \eta_{S_j} \tag{8}$$

$$E_{ij} = \Gamma'_j \ h_j \eta_{M_j} \eta_{S_j} \tag{9}$$

where i is a polygon, j is a polygon neighbouring i, $(I_R)_{ij} \geq 0$ is the reaction intensity of polygon j in J/m^2/s (Btu/ft^2/min), $(w_n)_j \geq 0$ is the net fuel load of polygon j in kg/m^2 (lb/ft^2), $E_{ij} \geq 0$ is the energy produced spreading from polygon i to polygon j per mass per time increment in J/kg/s (Btu/lb/min), $\Gamma'_j \geq 0$ is the optimum reaction intensity of polygon j in 1/s (1/min), $h_j \geq 0$ is the low heat content of polygon j in J/kg (Btu/lb), $0 \leq \eta_{M_j} \leq 1$ is the moisture damping coefficient of polygon j and $0 \leq \eta_{S_j} \leq 1$ is the mineral damping coefficient of polygon j. Just like the base Rothermel model, Eq. (9) may have poorer performance in some fuel beds at lower fuel loadings, especially those with high packing ratios. Minor modifications were made to the Rothermel model that helped reduce the over sensitivity to fuel depth (Albini, 1976). This updated form of the Rothermel model was not used by the IGS as ForeFire used the base Rothermel model which would make any comparison unfair.

The mean energy for fuel burnt while the fire spread to its neighbouring polygons was calculated by multiplying the mean E_{ij} from all neighbouring polygons by the mass of ignited fuel in the polygon and the time step. The energy was then divided by the low heat content of the polygon to get the amount of ignited mass lost to burning:

$$dB_i = \frac{\frac{\sum_{j=1}^{n_i} E_{ij}}{n_i} I_i dT}{h_i} \tag{10}$$

where i is a polygon, j is a polygon neighbouring i, $E_{ij} \geq 0$ is the energy produced spreading from polygon i to polygon j per mass per time increment in J/kg/s (Btu/lb/min), $dB_i \geq 0$ is the rate at which ignited fuel in polygon i becomes burnt fuel in kg/dT, $n_i > 0$ is the number of polygons that neighbour i, $I_i \geq 0$ is the amount of fuel that is currently ignited in polygon i in kg, $dT > 0$ is the timestep in s and $h_i \geq 0$ is the low heat content of polygon i in J/kg (Btu/lb). This value was then added to the total amount of fuel that has been burnt in that polygon:

$$(B_i)_{t+dT} = (B_i)_t + dB_i \tag{11}$$

where i is a polygon, $dT > 0$ is the timestep (s), $dB_i \geq 0$ is the rate at which ignited fuel in polygon i becomes burnt fuel (kg/dT), t is time in the simulation (s), and $(B_i)_t \geq 0$ is the amount of fuel that is burnt in polygon i at t time (kg). Tracking both ignited and burnt fuel amounts paints a clear image of what areas are still on fire and which have ceased burning (Figure 8).

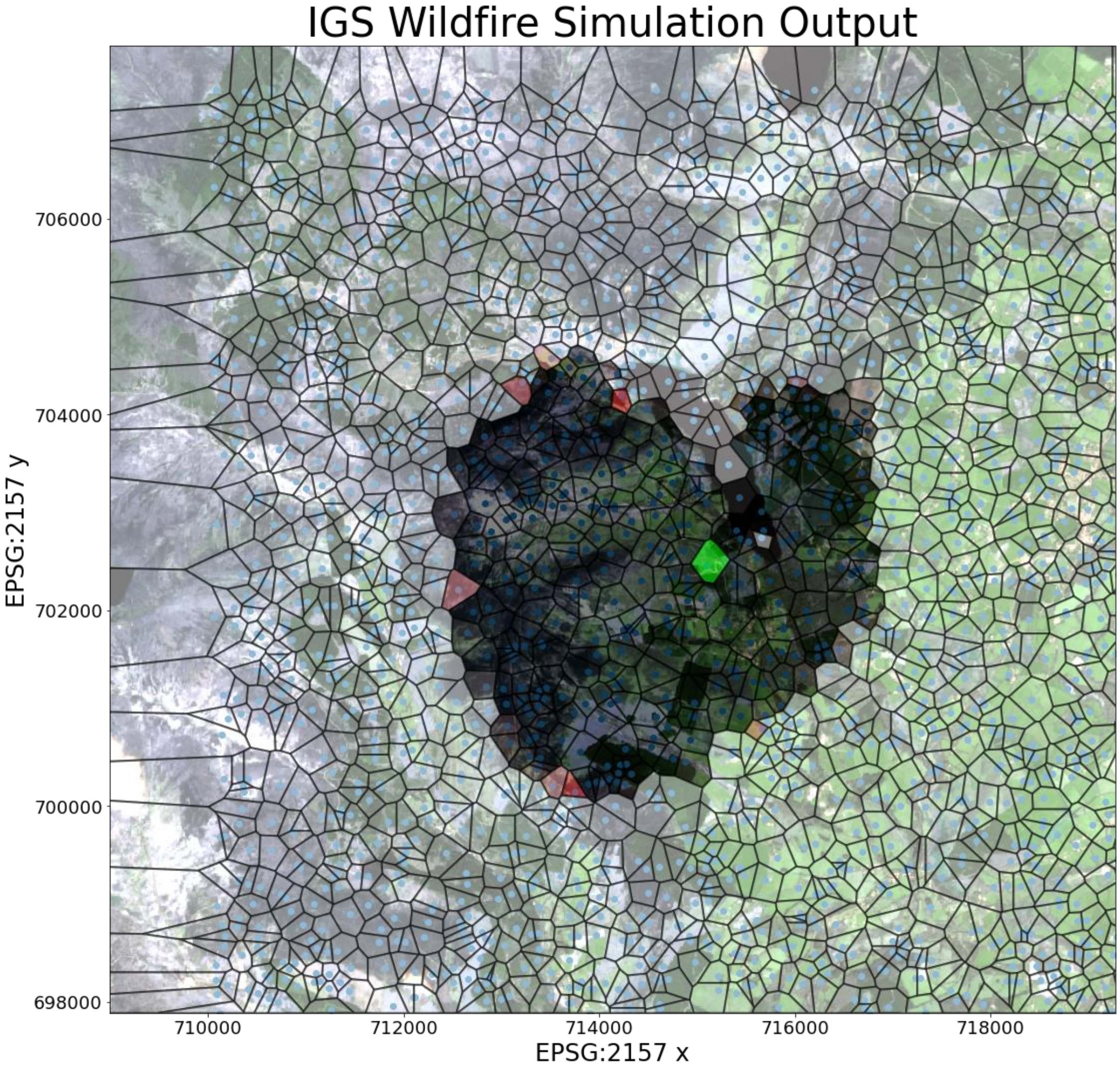

Figure 8: Sample simulation output after 100,000 seconds showing not burnt (transparent), wildfire source (green), burning (red) and burnt (black) polygons based on the mass of the polygon in that state, with coloured satellite image of the terrain in the background (Copernicus 2022).

The boundary of ignited and burnt polygons can then be converted into a fire line, representing the outer perimeter of the fire. This allows the IGS to be compared to other continuous programs such as ForeFire. The boundary polygons need to be found to produce a fire line. A straightforward algorithm was used to find the boundary polygons where each polygon was checked to see if it can spread the fire (a polygon is only able to spread fire if the fire spread model has reached that polygon's site). Once a set of polygons capable of spreading fire is found, it can be shortened by checking if these polygons have one or more neighbours that cannot spread fire. This gives an updated set of boundary polygons that are on the fire line perimeter of the IGS. This was then converted to a list containing boundary edges by checking the edges of each boundary polygon and recording any edges between a boundary polygon and a polygon that cannot spread fire. Getting the boundary edges of the boundary polygons helps reduce underpredicting of how far the fire has spread within a given boundary polygon. Polygons that can spread fire while also touching the outer perimeter of the NetCDF file are checked and any edges of that polygon on the perimeter were added to the list of boundary edges. Edges each consist of two vertices. To begin a random edge is selected from the list of boundary edges and a recursive algorithm finds the next edge to share vertices with the randomly selected edge, removing it from the list and storing it in a stack. This process is repeated until no more shared vertices can be found and the stack then becomes an ordered boundary. The process of selecting a random edge continues until all edges have been removed from the list (Figure 9). Sometimes the vertices of edges do not line up perfectly, so a 3 m (approximately 0.01 % of the region width) margin of error between vertices of different edges was allowed to ensure a completed shape. All edges shorter than the margin of error were removed to prevent other problems generating the completed boundary.

The ordered boundary was then smoothened to provide the final representation of the fire line, using a cubic spline. A Python library SciPy has a subpackage called interpolation (Interpolation (scipy.interpolate), 2024). This subpackage can be used to generate cubic splines given separate lists of x and y coordinates. Traditionally cubic splines are performed where the values of x coordinates are ordered in increasing value. This will not work for the IGS's output as reordering the x coordinates would make the boundary unordered and ruin the boundary shape. To circumvent this, a parameter (c) was defined as the cumulative distance between all previous points in the ordered boundary. For example, the first point would have c equal to 0, while the second point a would be equal to the distance between the first and second point. The third point would have c equal to the distance between the first and second point summed with the distance between the second and third point. Due to c being a list of increasing values it was possible to perform a cubic spline on both the x and y coordinates separately using c (Cubic spline for non-monotonic data (not a 1d function), 2024). The x and y coordinates can them be re-combined to produce a cubic spline. With the cubic spline, the fire line becomes a smooth curve, more closely resembling how a real fire would appear (Figure 9).

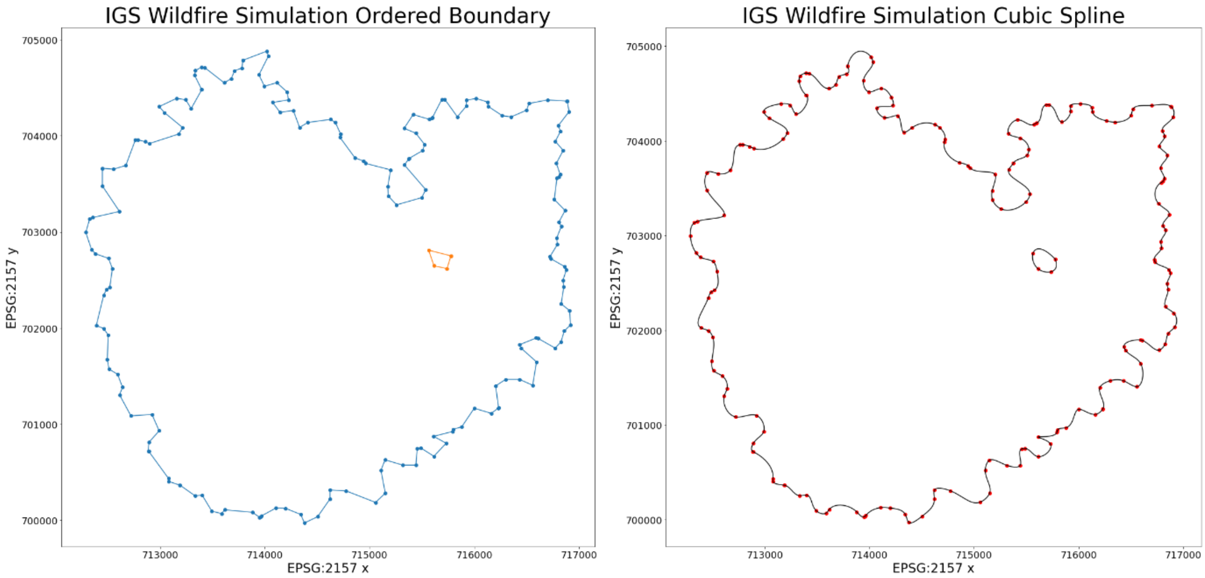

Figure 9: Left: Ordered boundary consisting of polygon edges from the IGS output that create an outer perimeter of the simulated wildfire; right: the ordered boundary smoothened by a cubic spline.

3.3 Methodology to Compare Two Wildfire Outputs

A new method was developed to compare the similarity of outputted wildfires. This method took inspiration from both set theory and receiver operating characteristics (ROC). When comparing outputted wildfires, one of the wildfires was treated as the ground truth. For this paper ForeFire was selected as the ground truth due to its increased resolution, therefore the IGS was the predicted wildfire. The area of predicted fire that overlapped with the ground truth was treated as true positives. Areas where IGS predicted fire and the ForeFire did not were false positives. Areas where ForeFire predicted fire and IGS did not were false negatives and areas where both IGS and ForeFire predicted there would not be fire were true negatives (Figure 10). Unfortunately, the area of true negatives could be considered infinite, dependant on the size of the plane where the fire is being simulated. Therefore, true negatives were not counted and any confusion matrix calculations involving them could not be found.

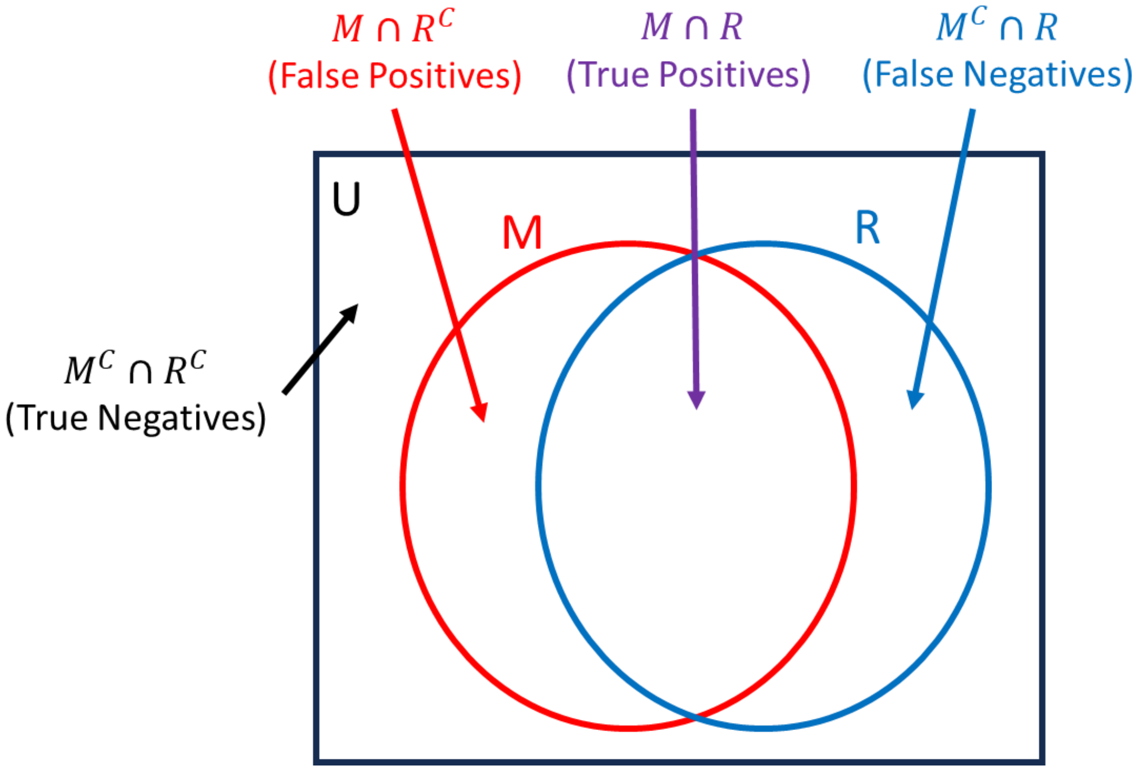

Figure 10: Calculating similarities of IGS (M) and ForeFire (R) wildfire shape outputs. Where the area of $M \cap R$ are the true positives, $M \cap R^C$ are the false positives, $M^C \cap R$ are the false negatives and $M^C \cap R^C$ are the true negatives.

The threat score was chosen as the metric of accuracy due to it measuring true positives, false positives, and false negatives simultaneously without requiring a measurement of true negatives. The default threat score equation was used but two weighting factors were added to allow the scaling of how important false positives and false negatives are relative to the true positives. It is given by:

$$TS = \frac{TP}{TP + w_1 FN + w_2 FP} \tag{12}$$

where TS is the threat score, TP is the area of true positives, FN is the area of false negatives, FP is the area of false positives, $w_1 \geq 0$ is the weighting coefficient for FN and $w_2 \geq 0$ is the weighting coefficient for FP.

If TS is 1 both the IGS an ForeFire polygons line up perfectly, while if TS is 0 then there is no overlap between the IGS and ForeFire polygons. Increasing w_1 or w_2 will make FN and FP respectively, have a greater impact on lowering TS while decreasing w_1 or w_2 will have the opposite effect. For the comparison of results in this paper $w_1 = w_2 = 1$.

4 Comparison of Grid Types

Five different grid types were compared to find which grid type may be the optimal grid for comparison against ForeFire. All five grids were built using Voronoi diagrams. The five grid types consisted of: randomly plotted sites, triangular tessellating grid, square tessellating grid, hexagonal tessellating grid and a grid where more sites would be plotted closer to the wildfire source and on landcover types defined by the fuel properties table, which are important to monitor; this grid is referred to as the flammable resolution grid (FRG). Each grid uses the Von Neuman neighbourhood for determining which cells neighbour other cells (corner neighbouring cells exclusive) (Małecki, 2017).

The grid with randomly plotted sites grid takes as an input the number of sites it must plot and a bounding box to plot the sites inside. It randomly plots that number of sites within the bounding box. If a site is plotted on top of an existing site, then that site will be re-plotted. Each polygon in the randomly plotted grid will have at least one neighbour with no maximum number of neighbours (Figure 12).

The triangular grid consists of tessellating equilateral triangles of equal size. It uses a user inputted number of sites it must plot and a bounding box to plot the sites inside. It is one of the more complicated grid types to plot using a Voronoi diagram. The program iterated through possible different numbers of sites to place on each row until it finds the optimal number. Using these values, it was possible to find the horizontal distance between sites and therefore the required vertical height of triangles on each column using the properties of equilateral and right-angled triangles (Figure 12), through the following equation:

$$y = \sqrt{3x^2} = Y_l + Y_s \tag{13}$$

where $y > 0$ is the height of the equilateral triangles, $x > 0$ is the horizontal distance between sites, $Y_s > 0$ is the short vertical offset and $Y_l > 0$ is the long vertical offset. The program continues iterating until it finds the smallest number of sites to place in a row, where once the entire grid is filled vertically there will still be approximately the same number of sites as originally inputted. A short vertical offset is found to generate the triangular shape. The short vertical offset is calculated as the site for each equilateral triangle should be equidistant from all corners of the triangle. Therefore, using Pythagoras's theorem, the horizontal distance between sites and vertical height of the triangles, the location of these sites can be found by:

$$Y_l = x^2 + Y_s^2 \tag{14}$$

$$Y_s = -\frac{x^2 - y^2}{2y} \tag{15}$$

where $Y_s > 0$ is the short vertical offset, $x > 0$ is the horizontal distance between sites, $y > 0$ is the height of the equilateral triangles and $Y_l > 0$ is the long vertical offset. In triangular tessellation on a singular row the triangles appear upright (generated from the short vertical offset) followed by an upside-down triangle in a repeating pattern. The upside-down triangles require the long vertical offset to be placed correctly. To find the long vertical offset the short vertical offset is taken away from the vertical height of the triangles (Figure 11) Eq. (13). Where both rows and columns are indexed even or odd, the short vertical offset is applied. While on odd indexed rows and even indexed columns, or even indexed rows and odd indexed columns, the long vertical offset is applied. This placement of sites creates a triangular grid. Due to rounding errors in Python and the Foronoi Python library, additional small edges were sometimes created between sites. To fix this, if the length of an edge was under a certain threshold it was removed to prevent the grid from assuming some sites had additional neighbours (Figure 12).

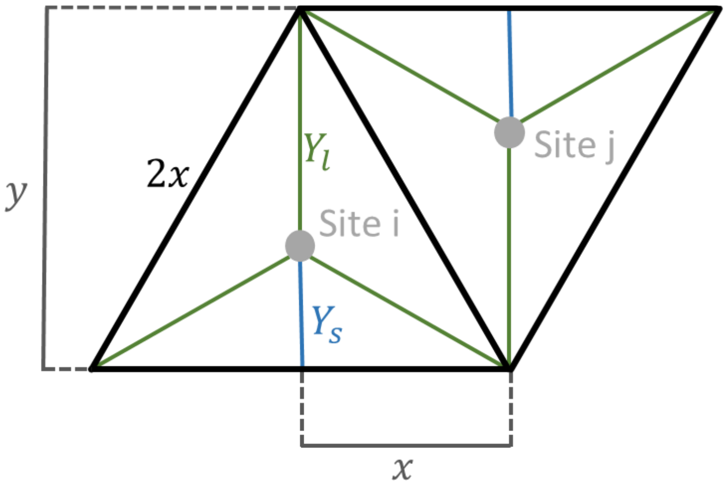

Figure 11: Calculation of Voronoi diagram site placement for equilateral triangular tessellation. Given the distances of x and y, Y_s and Y_l can be derived using the Pythagorean theorem.

The square grid consists of tessellating squares in a chess board pattern where the squares are all equal size and aligned both horizontally and vertically. It uses the user inputted number of sites and a bounding box to plot the sites inside. The floored square root of the number of sites to plot is found and is used to equally space sites within the bounding box into equidistant columns and rows. Each polygon in the square grid (not on the edge) will have four neighbours (Figure 12).

The hexagonal grid consists of tessellating regular hexagons of equal size defined by the user inputted number of sites and a bounding box. Like the triangular grid, the program iterated through possible different numbers of sites to place on each column until it finds the optimal number. The optimal number of points to place was found by getting a number as close to what was originally stated within the bounding box, as the horizontal distance between sites could be found given the vertical distance, i.e.:

$$x = \sqrt{y^2 - \left(\frac{y}{2}\right)^2} \tag{16}$$

where $x > 0$ is the horizontal distance between sites and $y > 0$ is the vertical distance between sites. Every even column is then offset vertically by half the vertical distance between sites. Each polygon in the hexagonal grid (not on the edge) will have six neighbours (Figure 12).

A random grid with increased focus on a region of interest was created called the FRG. It was designed to have a greater density of sites in regions most likely to be affected by wildfire and regions that are important to monitor. The regions that are most likely to be affected by wildfires tend to be situated closer to the ignition point of the fire or contain landcover types that can sustain large fires (e.g. forested areas). Areas of preselected importance tend to be areas with a higher density of people (e.g., urban areas). To do this the program runs through each pixel of the landcover map within the bounding box where it has a base probability to select the pixel it is currently on (this ensures not too many sites are generated). Once a pixel has been selected a second probability is generated to determine if a site should be placed at this position based on the distance from the ignition point and the landcover type of that pixel:

$$G = F(1 + m)^{-k} \tag{17}$$

where $0 \leq G \leq 1$ is the probability of a site being placed, $0 \leq F \leq 1$ is the coefficient affecting the probability of site placement based on landcover type, $m \geq 0$ is the distance between the current pixel and the mean ignition point, divided by the distance between the ignition point and furthest point within the bounding box and $k > 0$ is a coefficient that weighs site placement probability based on $(1 + m)$. This allows most of the computing to be performed where it is most likely to be needed instead of wasting resources in areas the fire is unlikely to reach or to spread fast in. Each polygon in the FRG will have at least one neighbour with no maximum number of neighbours (Figure 13).

A fire was set at the coordinates EPSG:2157 (715122, 702388) at 0 seconds in the simulation. The environment had no wind, and the fire was simulated for 100,000 seconds with a timestep of 1,000 seconds. The fire was simulated 10 times for each grid type where the time it took for the IGS to simulate the fire, and the threat score were both recorded (Table 1). Values for the equation of site placement of the FRG were selected so the FRG would have approximately 2,000 sites each time.

	Triangular Grid	**Square Grid**	**Hexagonal Grid**	**Random Grid**		**Fire Resolution Grid**		
Number of Sites	1952	1936	1974	2000		-		
Threat Score	0.84	0.84	0.82	-		-		
	Simulation Time (s)	**Simulation Time (s)**	**Simulation Time (s)**	**Simulation Time (s)**	**Threat Score**	**Number of Sites**	**Simulation Time (s)**	**Threat Score**
Fire 1	32.8	31.65	45	46.71	0.84	1995	52.46	0.87
Fire 2	33.27	31.56	43.43	49.99	0.79	1975	52.32	0.86

Fire 3	32.05	31.82	43.76	44.35	0.78	1980	52.37	0.85
Fire 4	31.86	32.14	44.03	48.83	0.82	1946	50.26	0.91
Fire 5	32.26	32.08	44.41	43.85	0.8	2032	54.88	0.86
Fire 6	32.19	32.14	43.4	44.7	0.72	2034	53.04	0.89
Fire 7	32.02	31.81	44.3	45.52	0.79	2044	53.14	0.89
Fire 8	33.86	32.63	44.1	42.06	0.83	1996	52.46	0.83
Fire 9	31.89	31.41	44.13	40.8	0.76	2002	52.28	0.89
Fire 10	32.02	31.93	44.29	40.9	0.79	1973	50.96	0.9
Mean	32.42	31.92	44.08	44.77	0.79	1997.7	52.42	0.87
Standard Deviation	0.67	0.35	0.47	3.11	0.03	31.23	1.24	0.03
Standard Error	0.21	0.11	0.15	0.98	0.01	9.88	0.39	0.01

Table 1: Number of sites, threat scores and simulation times for the different grid types. All grid types excluding the FRG have a constant number of sites. All grid types except the Random Grid and FRG have a constant threat score.

The regular grids and the random grid had a set number of sites while the FRG had a varying level of sites due to the nature of its site generation. As the regular grids were deterministic each simulation produced the same wildfire and therefore their threat score was constant throughout, unlike the random grid and FRG. The regular grids tend to have quicker execution times than the random grid and FRG, where the FRG has the longest simulation time. The random grid and FRG also have significantly higher standard deviations and standard errors in their simulation time, this is due to both grids generating a completely new grid layouts each time the simulation is run while the regular grids always have the exact same grid layout. The random grid had the lowest threat scored while the FRG had the highest threat score.

The triangular, square, and hexagonal grids can estimate the shape of the fire line well as shown by their threat scores, but they are confined to their regular shape and may struggle to represent some of the more irregular shapes of actual fire lines. As the Von Neumann neighbourhood was used to define neighbours. The regular grids all share the property of having an equal number of neighbours (not on the bounding box edges) that are all equidistant from each other which gives all polygons an equal spreading opportunity (Figure 12). The regular grids had an equal resolution across the entire grid. This leaves room for further optimisation where areas less likely to burn according to the model could have a lower resolution.

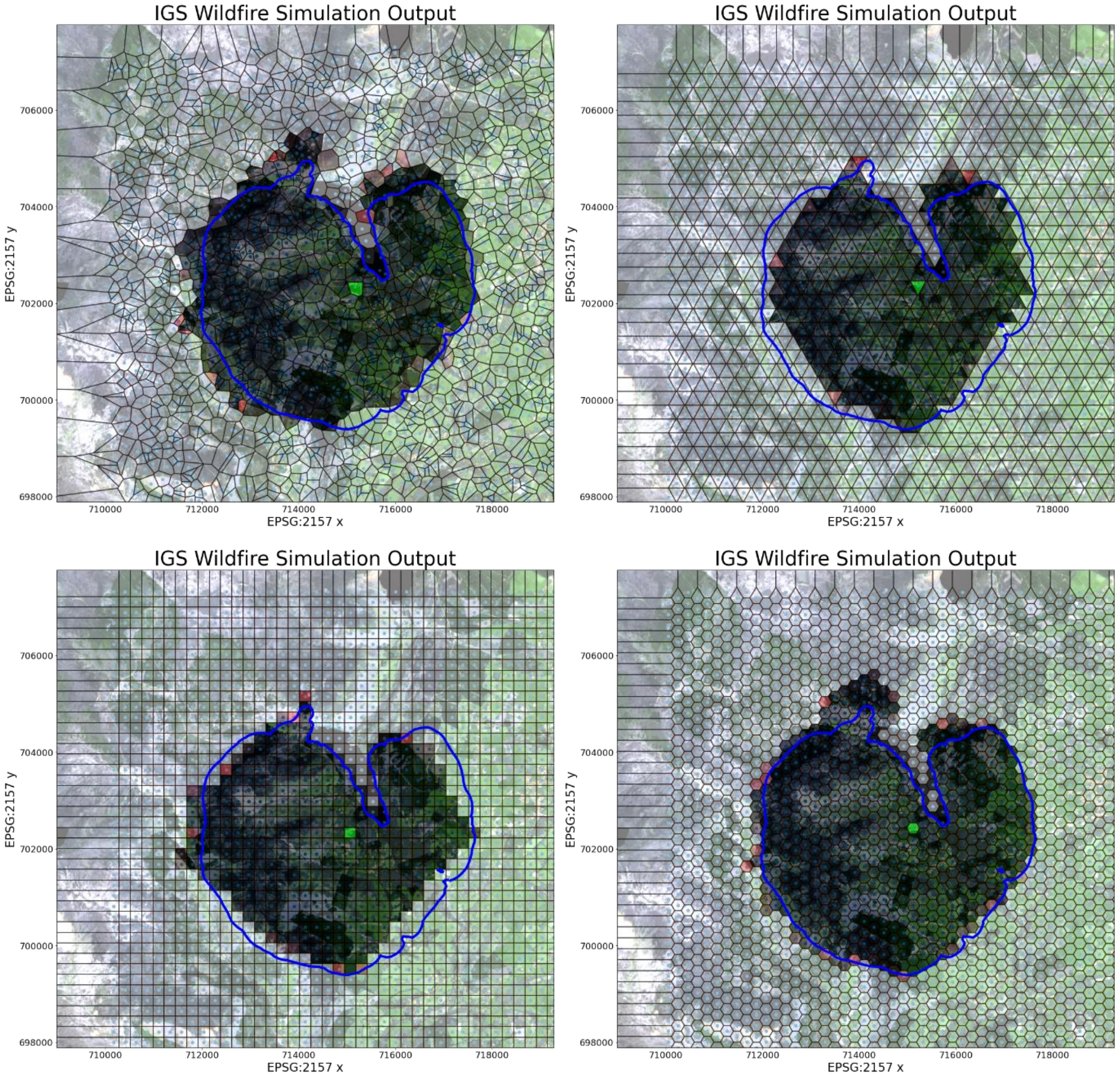
IGS Wildfire Simulation Output
IGS Wildfire Simulation Output
IGS Wildfire Simulation Output
IGS Wildfire Simulation Output
EPSG:2157 y
706000
704000
702000
700000
698000
710000
712000
714000
716000
718000
EPSG:2157 x

Figure 12: Outputs of different IGS grid types (coloured polygons) compared to ForeFire's output (blue). Top left: randomly plotted sites, top right: triangular grid, bottom left: square grid, bottom right: hexagonal grid (Copernicus 2022).

The randomised Voronoi grid can produce some of the more irregular shapes which can allow it to follow the irregular pattern of real terrain. Unlike the regular grids each polygon does not have an equal spreading opportunity due to the distance between neighbouring sites varying. The randomised Voronoi grid has a similar resolution throughout the grid which was not an optimal use of computation power. The FRG can follow the shape of the fire line with greater precision due to its high resolution in areas of importance. It also decreases computational waste due to it rendering smaller numbers of polygons in areas of little importance. It does not have an equal spreading opportunity for each polygon and takes longer than all other types of grids to simulate the spread of fire (Figure 13).

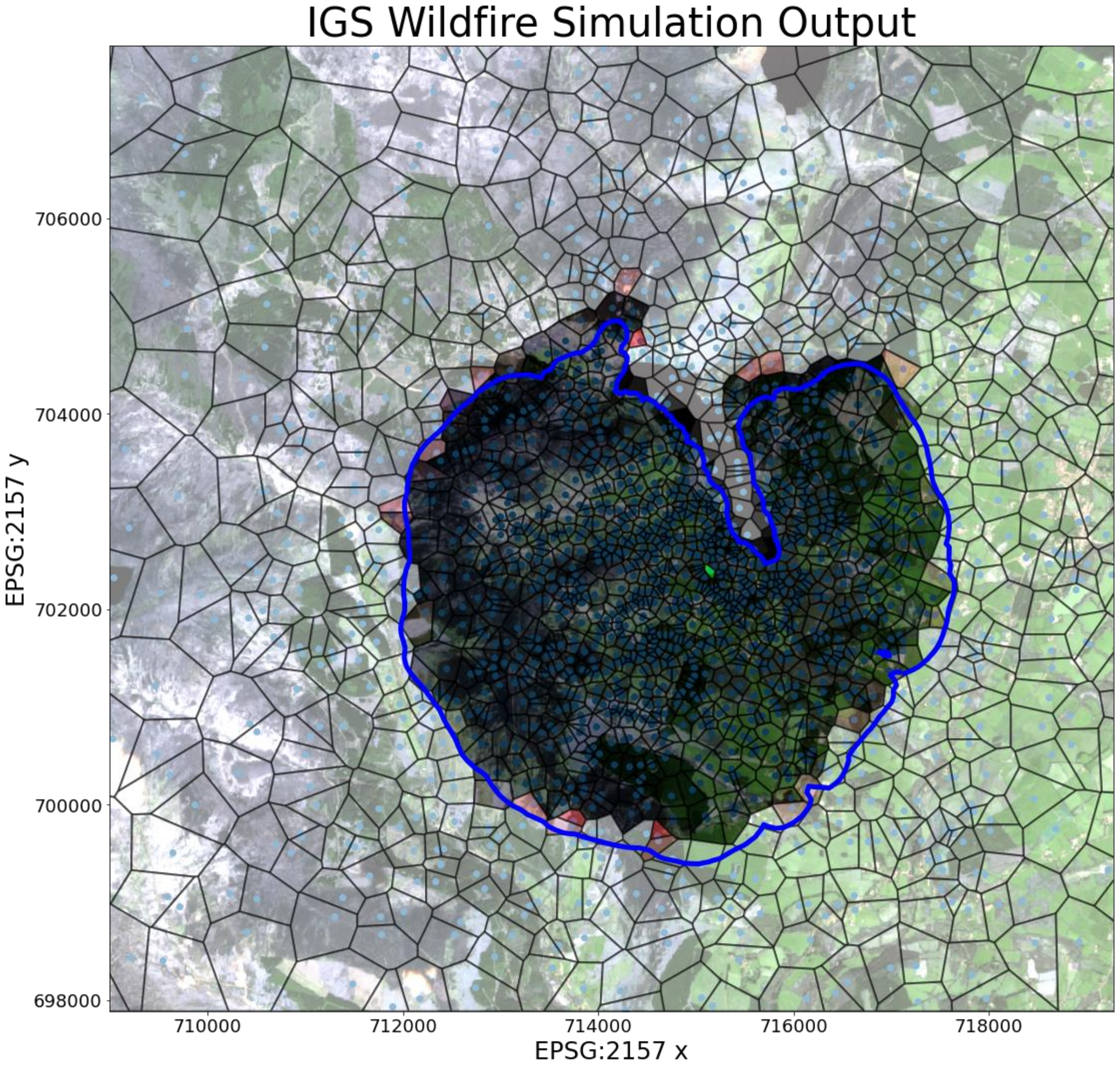

Figure 13: Output of the flammable resolution grid (coloured polygons) compared to ForeFire's output (blue) (Copernicus 2022).

Due to the greater resolution of the FRG, it will be used in the remaining comparisons in this paper. To find how effective IGS was at simulating wildfires it was then compared to the industry standard wildfire modelling program ForeFire.

5 Summary of Results

To compare IGS and ForeFire, 20 wildfires were simulated. The wildfires were randomly placed in various locations within the Wicklow Mountains, Ireland with varying wind speeds and for different durations. This was done to test the versatility of the IGS. Some simulations included multiple separate wildfires. Outputs of ForeFire and the cubic spline of the IGS using the FRG were graphed and then their similarity was found (Figure 14). While simulating these wildfires the time it took to run the simulation of the IGS usually took longer than ForeFire even though it required less computing. This was due to ForeFire being built in the more computationally efficient language C++ (Zehra et al., 2020). To make the comparison between the IGS and ForeFire fair, the main simulation loop of the IGS was ported to C++ from Python to provide an equal comparison. The remaining code for IGS was still in Python. The comparison does not include the setup times for ForeFire or the IGS.

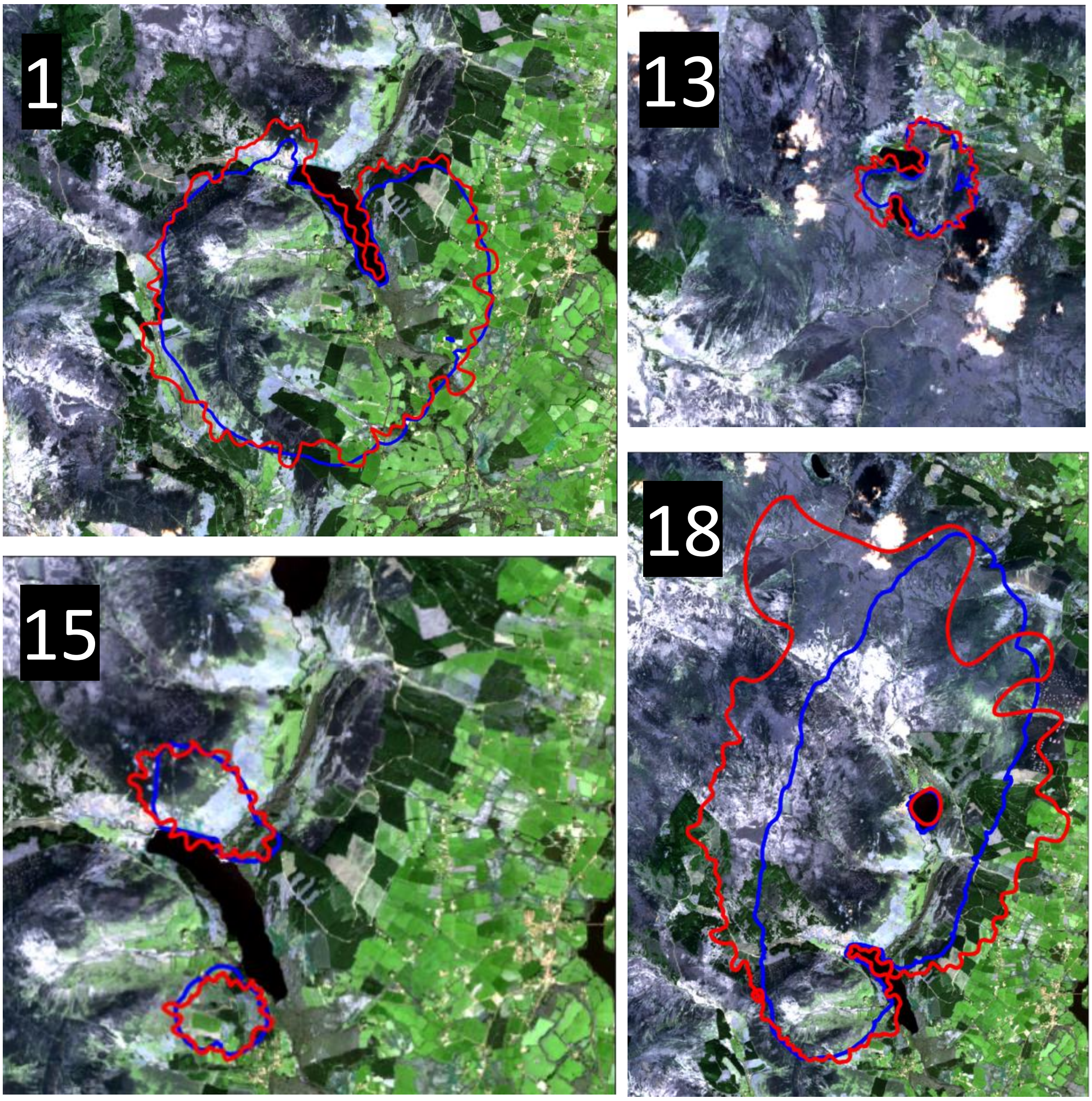

Figure 14: IGS outputs (Red) compared to ForeFire output (Blue). (Top left: Wildfire 1, Top Right: Wildfire 13, Bottom left: Wildfire 15 and Bottom Right: Wildfire 18 (Table 2)). (Copernicus 2022)

The time it took to run the main simulation loops of ForeFire and IGS in both Python and C++ were recorded and compared. The area of true positives, false positives, and false negatives for the IGS when compared to ForeFire were recorded for each wildfire and the corresponding threat score was calculated. This data is presented in (Table 2).

From the sample wildfires simulated, the IGS runs on average 34 times quicker than ForeFire with a mean fire line similarity of 0.8 when compared to ForeFire. The use of IGS instead of ForeFire allows multiple wildfires to be simulated (restricted to one area) in the same amount of time it would take ForeFire to run a simulation of one wildfire. From (Table 2) it is evident that the fire lines produced by the IGS when compared to their ForeFire counterpart tended to contain mostly areas of true positives followed by false positives and finally false negatives. There is also a trend where IGS simulated wildfires that are run for longer durations tend to not perform as well as those run for shorter durations relative to ForeFire. Adding wind or additional wildfire sources in the comparison does not impact the relative computing time or similarity to ForeFire. There are some limitations with the IGS, however. The FRG can have problems with larger wildfires as the frequency of polygons decreases the further a wildfire spreads from its ignition point. This is evident in Wildfire 18 (Figure 14) where there is a large drop in site resolution at distances far away from the wildfire-starting location. The large distance between the IGS and ForeFire's northern and western fire lines highlights this issue. The cubic spline used, forces the curve to intersect the points on the edges of polygons. Due to the resolution of sites within the IGS, this makes the cubic spline produce small modulations on the fire line while ForeFire has straighter edges.

Wildfire	Start Points and Time (x, y, t) (EPSG:2157, s)	Wind (Zonal, Meridian) (m/s)	Simulation Run Time (s)	ForeFire Time (s)	IGS Python Time (s)	IGS C++ Time (s)	ForeFire - IGS C++ Time (s)	ForeFire / IGS C++ Time (s)	True Positives (m^2)	False Positives (m^2)	False Negatives (m^2)	Threat Score
1	(715122, 702388, 0)	(0, 0)	100,000	5.3	46	0.193	5.105	27	21,162,353	2,597,119	594,021	0.87
2	(713350, 704946, 0)	(0, 0)	100,000	7.1	58	0.321	6.773	22	31,641,462	6,709,658	393,613	0.82
3	(715651, 706392, 0)	(0, 0)	100,000	6.9	68	0.267	6.613	26	29,336,486	6,015,675	477,550	0.82
4	(710773, 707941, 0)	(0, 0)	100,000	5.9	72	0.294	5.584	20	28,808,953	4,668,653	492,470	0.85
5	(713513, 708268, 0)	(0, 0)	100,000	6.6	61	0.256	6.342	26	28,944,866	6,508,166	543,992	0.80
6	(712496, 708990, 0)	(0, 0)	100,000	5.9	65	0.314	5.561	19	28,969,212	4,046,505	273,223	0.87
7	(715823, 708190, 0)	(0, 0)	100,000	7.1	61	0.289	6.791	24	31,247,060	5,138,147	315,410	0.85
8	(713467, 711230, 0)	(0, 0)	100,000	6.3	63	0.291	6.018	22	29,105,672	3,749,638	351,839	0.88
9	(714378, 701907, 0)	(0, 0)	100,000	5.5	55	0.238	5.263	23	25,224,112	2,781,007	1,005,543	0.87
10	(704110, 703419, 0)	(0, 0)	100,000	7.2	55	0.257	6.896	28	30,443,340	3,596,271	892,268	0.87
11	(709231, 705966, 0)	(0, 0)	50,000	1.8	27	0.072	1.717	25	5,684,170	418,904	771,298	0.83
12	(714122, 706378, 0)	(0, 0)	50,000	2.5	26	0.081	2.451	31	6,941,426	482,771	772,200	0.85
13	(714027, 716143, 0)	(0, 0)	30,000	1.4	15	0.028	1.394	51	1,701,109	497,978	84,778	0.74
14	(713369, 707072, 0)	(0, 0)	120,000	9.9	75	0.371	9.509	27	43,051,154	8,672,817	792,791	0.82
15	(715122, 702388, 0)	(0, 0)	20,000	1.7	11	0.043	1.621	39	1,833,936	293,936	237,246	0.78
16	(714868, 704531, 2000) (715148, 702437, 0) (715930, 703049, 1200)	(0, 0)	30,000	2.2	14	0.034	2.142	64	3,856,323	398,370	671,296	0.78
17	(715804, 702092, 2700) (714633, 704695, 0) (715297, 707714, 0)	(0, 0)	50,000	3.7	27	0.106	3.609	35	13,747,087	3,459,420	326,943	0.78
18	(713945, 703907, 0)	(1, 3)	50,000	10.6	31	0.158	10.404	67	42,742,800	25,938,603	3,724,274	0.59
19	(712649, 715686, 0)	(1, -3)	20,000	2.1	11	0.040	2.068	53	4,779,324	1,406,943	1,845,529	0.60
20	(713925, 707395, 0)	(-0.5, -0.4)	30,000	2.0	17	0.043	1.997	47	4,085,839	283,454	714,927	0.80
Mean				5.1	43	0.185	4.893	34	20,665,334	4,383,202	764,061	0.80
Standard Deviation				2.8	23	0.118	2.664	15	13,902,996	5,658,324	792,456	0.08
Standard Error				0.6	5.1	0.03	0.6	3.3	3,108,804	1,265,240	177,199	0.02
Sum									413,306,686	87,664,034	15,281,210	0.80

Table 2: Comparison of 20 simulated wildfires in ForeFire and IGS under the metrics of computing time (ForeFire/IGS C++ Time) and similarity of resulting fire line (Threat Score, where $w_1, w_2 = 1$).

6 Discussion

Comparing the five different grid types produced a table of computational times and the similarity of their outputs relative to ForeFire (Table 1). Overall, the regular grids produced the same output every time the program ran making them deterministic. They also took approximately the same time to run the simulation as shown by the small standard deviation in their computational times. The small standard deviation can be accounted for with other uncontrollable minor factors effecting the computer such as background tasks. Both irregular grids have a much higher standard deviation in their computational times. This is because the process of seeding these grids is non-deterministic, so a different grid is produced each time a simulation is run. Some of the irregular grids have overall regions of higher resolution than others, which can lead to varying computational times. The number of neighbours each grid has is also correlated to the computational time, as the program must calculate the propagation ratio between each ignited polygon that can spread fire to its neighbour. The square and triangular grids have the fastest computational times while having four and three neighbours respectively. The hexagonal grid has the next fastest computational time with six neighbours. The random grid has a computational time very similar to that of the hexagonal grid even though there is no set number of neighbours per polygon. This is most likely due to each polygon having approximately six neighbours. The FRG has the slowest computational time even though most polygons also have approximately six neighbours. This is due to the increased density of polygons close to the wildfire source, as propagation ratios need to be calculated for all these ignited polygons and their neighbours that fire can spread to. In terms of similarity of output to ForeFire, the FRG performs the best followed by the square, triangular, hexagonal and random grids. As previously mentioned, the FRG has a high density of polygons near the wildfire source, allowing the simulation to nearly follow the continuous movement of the wildfire simulated in ForeFire. For the regular grids to be comprised of regular shapes (and equilateral triangles in the case of the triangular grid) it was only possible to have a set number of polygons in the simulation area. In the table, the number of sites for the regular grids was set as close to 2,000 as possible while still retaining a regular grid. The disparity in the number of polygons between the grids can explain some of the differences in similarities with ForeFire. The irregular random grid tends to produce outputs with the lowest similarity to ForeFire's than any other grid. This may be due to the minor variations in cell sizes within the random grid. The regular grids have a set number of directions fire can spread in to reach neighbouring polygons; they also struggle with following the irregular patterns in the Earth's terrain as seen by the lake (Figure 12). Overall, there is no definitive best grid type. It really depends on what the user is trying to simulate and how detailed they are willing to have the simulation in exchange for more computational time. In this paper the FRG was chosen as it produced the results most like ForeFire's.

Comparing the IGS and ForeFire produced a table examining the computational efficiency and similarity of their outputs (Table 2). ForeFire is deterministic and therefore produces the same results every time it is run. In this comparison the IGS was using the FRG grid which is non-deterministic. The IGS wildfires that ran for shorter durations tended to be computationally faster than the wildfires that ran for longer durations, relative to ForeFire. This is caused by the increasing size of the simulated wildfire as it burns, resulting in more ignited polygons that can spread fire to their neighbours therefore creating additional computational burden per iteration. The IGS produces a larger area of false positives than that of false negatives when compared to

ForeFire, this is due to the fire line selection process. As the IGS selects the fire line from the outer edges of the polygon if fire has reached that polygon's site, regardless of how far the fire has spread within that polygon to its surrounding edges; it sometimes overestimates the distance the fire has spread. This is the reason for larger area of false positives than false negatives when compared to ForeFire. The threat score for similarity between the IGS and ForeFire increases with length of simulation. This is due to the overall area of the wildfire which tends to be mostly true positives, growing at a faster rate than the boundary of the wildfire where false positives and false negatives tend to occur.

7 Conclusion

A novel software platform that allows the change of grid type using simple parameterisation for wildfire modelling was developed. This allowed grids to be compared within the same framework. The software included irregular grids constructed from Voronoi diagrams. The approach of using irregular grids to predict the spread of wildfires allows for efficient computing while retaining a reasonable level of similarity to existing methods in results defined by the threat score. Existing software such as ForeFire can produce more precise simulations as it models fires in continuous space by moving markers representing the fire line, instead of simulating the spread of fire on a fixed grid determined before running the simulation. The use of continually moving markers requires additional computational time. The spatial resolutions of both programs must be noted however as ForeFire has a spatial resolution of 30 m^2 throughout, while the FRG has a varying spatial resolution with a maximum resolution of 10 m^2. The IGS has been incorporated into DecaMap's Common Operational Picture Software Platform, an application that has potential to assist with wildfire management (DecaMap, 2024).

The software has a few issues that could be improved in future work. A better form of site placement than the FRG could be developed to prevent the resolution of site placement decreasing at such a high rate at distances further from the wildfire source. A higher resolution could also be used to reduce the modulated boundary pattern produced by the cubic splines when converting the IGS output to a line. Porting the rest of the IGS from Python to a more computationally efficient programming language such as C++ would speed up the setup process for the IGS making the overall software more efficient.

For the results ForeFire was treated as the ground truth as its continuous nature can be made to produce results of a higher resolution than that set in the IGS experiments. Both approaches use the same fire progression model. ForeFire could be considered an industry standard as it is commonly referenced in relevant literature (Kaur et al., 2016; Farguell et al., 2019; Trucchia et al., 2019).

This paper compared different grid types. It was shown that the FRG is the best option for greater resolution near the wildfire source. The regular grids are a better choice for more efficient computation. The decision on which grid type to choose should be selected on a case-by-case basis by the user. The paper also showed that the IGS runs approximately 34 times quicker than ForeFire with a mean similarity of 0.8. While ForeFire will always produce wildfire simulations at a greater spatial resolution than the IGS, if computational efficiency is a concern where multiple wildfires may need to be simulated in the one location, then the IGS may end up being an alternative option.

8 Appendix A

A near cloud free satellite image (product) taken by the Sentinel-2 satellite in April 2022 was downloaded from the European Space Agency (ESA) sentinel-hub portal, a satellite image hosting platform (EO Browser, 2024). Sentinel-2 products contain multiple bands which capture electromagnetic radiation (light) reflected off the Earth's surface. A Level 1-C product (top of atmosphere reflectance values) was used in this study. The data was then atmospherically corrected to bottom of atmosphere values using the Semi-Automatic Classification tool in the geographic information software QGIS which follows the Dark Object Subtraction (DOS) algorithm (Chavez, 1988; QGIS, 2024; Gilmore et al., 2015). This is due to Level 1-C products originally containing information regarding both the Earth's surface and atmosphere which may not accurately depict the actual conditions on the surface of Earth. Three additional indices for highlighting vegetation, urban areas and soil (i.e. normalised difference vegetation index (NDVI), normalised difference built up index (NDBI) and normalised difference tillage index (NDTI) respectively) were generated from the original Sentinel-2 bands. These indices are simple ratios calculated from the original multispectral bands and are known to help distinguish various land cover classes, i.e. NDVI for vegetation, NDBI for urban areas and NDTI for bare soil. NDVI is a ratio between red and the near infra-red bands whereas NDBI is calculated from the short wave infra-red (SWIR2) and near infra-red bands (Sun et al., 2019; Kebede et al., 2022). The tillage index (NDTI) is calculated from both the available SWIR bands (SWIR1 and SWIR2) (Sun et al., 2019). Areas of known landcover in the Wicklow Mountains, Ireland were digitized to generate seventy polygons in total, which were further equally split into two sets. The first set was used to train a model, and the other was used as an independent validation of the predicted land cover class map, The training and validation datasets had 35 polygons each which totalled to an area of 1.45 km^2 and 1.4 km^2 respectively. The polygons were selected in a manner to ensure a coverage of the entire study area, separation between the polygons (avoiding extremely close areas for both training and validation) and intra-class variability for avoiding bias in the model. A supervised random forest algorithm was applied on the stack of seven original bands (red, green, blue, near infra-red (NIR), NIR narrow, short wave infra-red (SWIR1 and SWIR2) captured by the Sentinel-2 satellite and the three additional indices to predict the land cover classes. A supervised random forest algorithm was then applied on the stack of seven original bands (red, green, blue, near infra-red (NIR), NIR narrow, short wave infra-red (SWIR1 and SWIR2) captured by the Sentinel-2 satellite and the three additional indices to predict the land cover classes. The random forest classification algorithm is non-parametric in nature and known to improve estimates by averaging outputs from multiple decision trees randomly subset from the data. It has been successfully used for land cover mapping in previous studies and the algorithm was run using the RStoolbox library in the R programming environment (Sibanda and Ahmed, 2021; Abdi, 2020; Pringle et al., 2018). The cloud covered pixels in the area were classified based on a simple thresholding of the blue band (Baetens et al., 2019). The predicted land cover map was compared against the validation dataset which provided an overall accuracy of 97%. The predicted land cover map for the area under study is shown in (Figure A1).

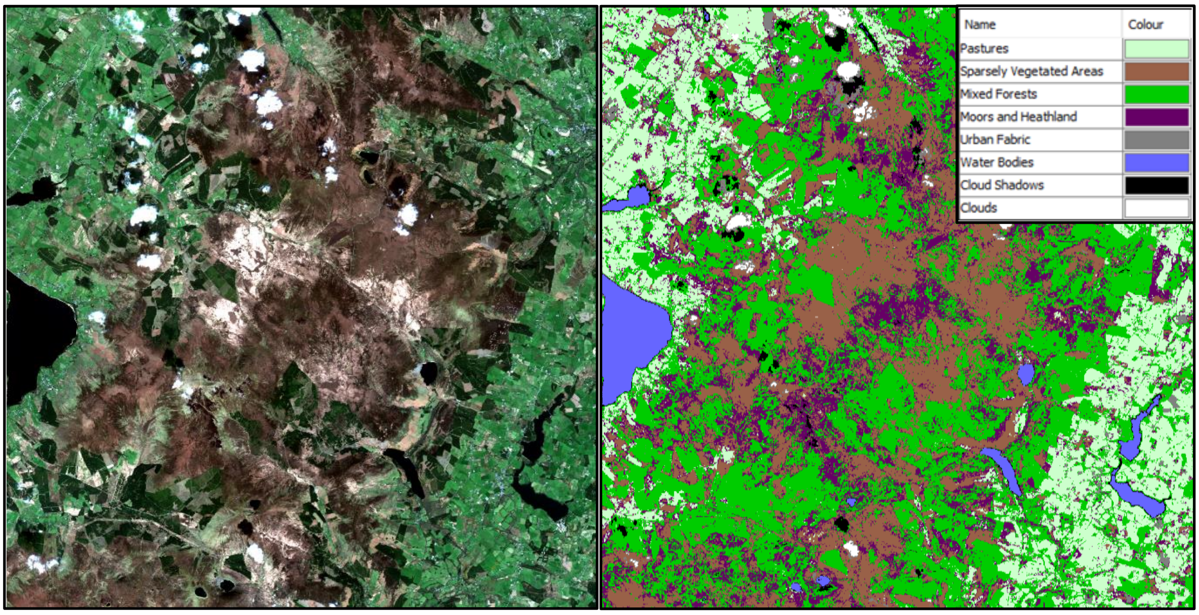

Figure A1: Left: Sentinel-2 based true colour image of the study area; and right: predicted land cover map for April 2022 (Copernicus 2022).

Code availability: Code is available upon request but may be subject to a standard licence.

Data availability: Data used in this study are freely available online from many sources which have been referenced within the text.

Author contributions: Conor Hackett, Charles Markham and Rafael de Andrade Moral conceived and implemented the research. Gourav Misra and Tim McCarty provided the landcover map used by the software. All authors contributed to the editing and have read and agreed to the published version of the manuscript.

Competing interests: The authors declare no competing interests.

Acknowledgements

This publication has emanated from research conducted with the financial support of Taighde Éireann – Research Ireland under Grant number 18/CRT/6049 and SFI/21/FIP/DO/9945P. The opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Research Ireland. We would like to thank the community and the referees for their valuable comments which helped improve the manuscript.

References

Abatzoglou, J. T., Williams, A. P., Boschetti, L., Zubkova, M., and Kolden, C. A.: Global patterns of interannual climate–fire relationships, Glob Chang Biol, 24, 5164–5175, https://doi.org/10.1111/gcb.14405, 2018.

Abdi, A. M.: Land cover and land use classification performance of machine learning algorithms in a boreal landscape using Sentinel-2 data, GIsci Remote Sens, 57, 1–20, https://doi.org/10.1080/15481603.2019.1650447, 2020.

Albini, F. A.: Computer-based models of wildland fire behavior: a user’s manual, Ogden, UT: USDA Forest Service, Intermountain Forest and Range Experiment Station. 71 p., 1976.

Al-Rawi, I.: Implementation of an Efficient Scan-Line Polygon Fill Algorithm , Computer Engineering and Intelligent Systems, 5, 22–28, 2014.

Andrews, P. L.: The Rothermel surface fire spread model and associated developments: A comprehensive explanation, The Rothermel surface fire spread model and associated developments: A comprehensive explanation. Gen. Tech. Rep. RMRS-GTR-371. Fort Collins, CO: U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station. 121 p., 2018.

Baetens, L., Desjardins, C., and Hagolle, O.: Validation of Copernicus Sentinel-2 Cloud Masks Obtained from MAJA, Sen2Cor, and FMask Processors Using Reference Cloud Masks Generated with a Supervised Active Learning Procedure, Remote Sens (Basel), 11, 433, https://doi.org/10.3390/rs11040433, 2019.

Beltrán-Marcos, D., Calvo, L., Fernández-Guisuraga, J. M., Fernández-García, V., and Suárez-Seoane, S.: Wildland-urban interface typologies prone to high severity fires in Spain, Science of The Total Environment, 894, 165000, https://doi.org/10.1016/j.scitotenv.2023.165000, 2023.

Benyon, R. G., Inbar, A., Sheridan, G. J., and Lane, P. N. J.: Critical climate thresholds for fire in wet, temperate forests, For Ecol Manage, 537, 120911, https://doi.org/10.1016/j.foreco.2023.120911, 2023.

Blanchi, R., Leonard, J., Haynes, K., Opie, K., James, M., and Oliveira, F. D. de: Environmental circumstances surrounding bushfire fatalities in Australia 1901–2011, Environ Sci Policy, 37, 192–203, https://doi.org/10.1016/j.envsci.2013.09.013, 2014.

Boegelsack, N., Withey, J., O'Sullivan, G., and McMartin, D.: A Critical Examination of the Relationship between Wildfires and Climate Change with Consideration of the Human Impact, J Environ Prot (Irvine, Calif), 09, 461–467, https://doi.org/10.4236/jep.2018.95028, 2018.

Chavez, P. S.: An improved dark-object subtraction technique for atmospheric scattering correction of multispectral data, Remote Sens Environ, 24, 459–479, https://doi.org/10.1016/0034-4257(88)90019-3, 1988.

Copernicus Data Space Ecosystem: https://dataspace.copernicus.eu/, last access: 25 January 2024.

Home :: Corine Land Cover classes: https://land.copernicus.eu/content/corine-land-cover-nomenclature-guidelines/html/, last access: 25 January 2024.

DecaMap: https://decamap.com/, last access: 22 February 2024.

Download - STEP: https://step.esa.int/main/download/, last access: 25 January 2024.

Farguell, A., Cortés, A., Margalef, T., Miró, J. R., and Mercader, J.: Scalability of a multi-physics system for forest fire spread prediction in multi-core platforms, J Supercomput, 75, 1163–1174, https://doi.org/10.1007/s11227-018-2330-9, 2019.

Filippi, J.-B., Bosseur, F., and Grandi, D.: ForeFire: open-source code for wildland fire spread models, in: Advances in forest fire research, Imprensa da Universidade de Coimbra, 275–282, https://doi.org/10.14195/978-989-26-0884-6_29, 2014.

Fons, W. L.: Analysis of Fire Spread in Light Forest Fuels, J Agric Res, 72, 92–121, 1946.

Tools: https://github.com/forefireAPI/firefront/tree/master/tools, last access: 25 January 2024.

Fuels.ff Fuel Attribute Table: https://github.com/forefireAPI/firefront/blob/master/examples/aullene/fuels.ff, last access: 25 January 2024.

Fortune, S.: A sweepline algorithm for Voronoi diagrams, Algorithmica, 2, 153–174, https://doi.org/10.1007/BF01840357, 1987.

Frandsen, W. H.: Fire spread through porous fuels from the conservation of energy, Combust Flame, 16, 9–16, https://doi.org/10.1016/S0010-2180(71)80005-6, 1971.

Gilmore, S., Nalband, A., and Dewan, A.: Effectiveness of DOS (Dark-Object Subtraction) method and water index techniques to map wetlands in a rapidly urbanising megacity with Landsat 8 data, CEUR Workshop Proc, 1323, 100–108, 2015.

Cubic spline for non-monotonic data (not a 1d function): https://stackoverflow.com/questions/67460967/cubic-spline-for-non-monotonic-data-not-a-1d-function, last access: 25 January 2024.

Hackett, C., Moral, R. D. A., and Markham, C.: Simulating Disease in Periods of Low Mobility Using a Hybrid Diffusion and Compartmental Model Built on Geographic Data, in: 2021 32nd Irish Signals and Systems Conference, ISSC 2021, https://doi.org/10.1109/ISSC52156.2021.9467871, 2021.

Haghani, M., Kuligowski, E., Rajabifard, A., and Kolden, C. A.: The state of wildfire and bushfire science: Temporal trends, research divisions and knowledge gaps, Saf Sci, 153, 105797, https://doi.org/10.1016/j.ssci.2022.105797, 2022.

Halofsky, J. E., Peterson, D. L., and Harvey, B. J.: Changing wildfire, changing forests: the effects of climate change on fire regimes and vegetation in the Pacific Northwest, USA, Fire Ecology, 16, 4, https://doi.org/10.1186/s42408-019-0062-8, 2020.

Helene, P., Britez, C., and Carvalho, M.: Fire impacts on concrete structures. A brief review, Revista ALCONPAT, 10, 1–21, https://doi.org/10.21041/ra.v10i1.421, 2019.

Janssen, T. A. J., Jones, M. W., Finney, D., van der Werf, G. R., van Wees, D., Xu, W., and Veraverbeke, S.: Extratropical forests increasingly at risk due to lightning fires, Nat Geosci, 16, 1136–1144, https://doi.org/10.1038/s41561-023-01322-z, 2023.

Jiao, Q., Fan, M., Tao, J., Wang, W., Liu, D., and Wang, P.: Forest Fire Patterns and Lightning-Caused Forest Fire Detection in Heilongjiang Province of China Using Satellite Data, Fire, 6, 166, https://doi.org/10.3390/fire6040166, 2023.

Jones, M. W., Santín, C., van der Werf, G. R., and Doerr, S. H.: Global fire emissions buffered by the production of pyrogenic carbon, Nat Geosci, 12, 742–747, https://doi.org/10.1038/s41561-019-0403-x, 2019.

Kala, C. P.: Environmental and socioeconomic impacts of forest fires: A call for multilateral cooperation and management interventions, Natural Hazards Research, 3, 286–294, https://doi.org/10.1016/j.nhres.2023.04.003, 2023.

Kaur, I., Mentrelli, A., Bosseur, F., Filippi, J.-B., and Pagnini, G.: Turbulence and fire-spotting effects into wild-land fire simulators, Commun Nonlinear Sci Numer Simul, 39, 300–320, https://doi.org/10.1016/j.cnsns.2016.03.003, 2016.

Kebede, T. A., Hailu, B. T., and Suryabhagavan, K. V.: Evaluation of spectral built-up indices for impervious surface extraction using Sentinel-2A MSI imageries: A case of Addis Ababa city, Ethiopia, Environmental Challenges, 8, 100568, https://doi.org/10.1016/j.envc.2022.100568, 2022.

Keeley, J. E. and Syphard, A. D.: Large California wildfires: 2020 fires in historical context, Fire Ecology, 17, 22, https://doi.org/10.1186/s42408-021-00110-7, 2021.

Małecki, K.: Graph Cellular Automata with Relation-Based Neighbourhoods of Cells for Complex Systems Modelling: A Case of Traffic Simulation, Symmetry (Basel), 9, 322, https://doi.org/10.3390/sym9120322, 2017.

McArthur, A. G.: Weather and grassland fire behaviour, 1923-1978 & Australia. Forestry and Timber Bureau, 1966.

McElwain, L. and Sweeney, J.: Climate change in Ireland- recent trends in temperature and precipitation, Irish Geography, 36, 97–111, https://doi.org/10.1080/00750770309555815, 2003.

Meier, S., Elliott, R. J. R., and Strobl, E.: The regional economic impact of wildfires: Evidence from Southern Europe, J Environ Econ Manage, 118, 102787, https://doi.org/10.1016/j.jeem.2023.102787, 2023.

Noble, I. R., Gill, A. M., and Bary, G. A. V.: McArthur's fire-danger meters expressed as equations, Australian Journal of Ecology, 5, 201–203, https://doi.org/10.1111/j.1442-9993.1980.tb01243.x, 1980.

Pais, C., Carrasco, J., Martell, D. L., Weintraub, A., and Woodruff, D. L.: Cell2Fire: A Cell-Based Forest Fire Growth Model to Support Strategic Landscape Management Planning, Frontiers in Forests and Global Change, 4, https://doi.org/10.3389/ffgc.2021.692706, 2021.

Park, H., Nam, K., and Lim, H.: Is critical infrastructure safe from wildfires? A case study of wildland-industrial and -urban interface areas in South Korea, International Journal of Disaster Risk Reduction, 95, 103849, https://doi.org/10.1016/j.ijdrr.2023.103849, 2023.

Penney, G., Habibi, D., and Cattani, M.: Firefighter tenability and its influence on wildfire suppression, Fire Saf J, 106, 38–51, https://doi.org/10.1016/j.firesaf.2019.03.012, 2019.

Piñol, J., Beven, K., and Viegas, D. X.: Modelling the effect of fire-exclusion and prescribed fire on wildfire size in Mediterranean ecosystems, Ecol Modell, 183, 397–409, https://doi.org/10.1016/j.ecolmodel.2004.09.001, 2005.

Prat-Guitart, N., Nugent, C., Mullen, E., Mitchell, F. J. G., Hawthorne, D., Belcher, C. M., and Yearsley, J. M.: Peat Fires in Ireland, in: Coal and Peat Fires: A Global Perspective, Elsevier, 451–482, https://doi.org/10.1016/B978-0-12-849885-9.00020-2, 2019.

Pringle, M. J., Schmidt, M., and Tindall, D. R.: Multi-decade, multi-sensor time-series modelling—based on geostatistical concepts—to predict broad groups of crops, Remote Sens Environ, 216, 183–200, https://doi.org/10.1016/j.rse.2018.06.046, 2018.

QGIS: https://qgis.org/en/site/, last access: 25 January 2024.

dos Reis, M., Graça, P. M. L. de A., Yanai, A. M., Ramos, C. J. P., and Fearnside, P. M.: Forest fires and deforestation in the central Amazon: Effects of landscape and climate on spatial and temporal dynamics, J Environ Manage, 288, 112310, https://doi.org/10.1016/j.jenvman.2021.112310, 2021.

Rothermel, R. C.: A mathematical model for predicting fire spread in wildland fuels, Res. Pap. INT-115. Ogden, UT: U.S. Department of Agriculture, Intermountain Forest and Range Experiment Station. 40 p., 1972.

San Martin, D. and Torres, C.: 2D Simplified Wildfire Spreading Model in Python: From NumPy to CuPy, CLEI Electronic Journal, 26, https://doi.org/10.19153/cleiej.26.1.5, 2023.

Interpolation (scipy.interpolate): https://docs.scipy.org/doc/scipy/reference/interpolate.html, last access: 25 January 2024.

EO Browser: https://www.sentinel-hub.com/explore/eobrowser/, last access: 25 January 2024.

SFI-Defence Organisation Innovation Challenge, Challenge 1: https://www.sfi.ie/funding/funding-calls/future-innovator-defence/#row-collapse-item-1, last access: 25 January 2024.

Sibanda, S. and Ahmed, F.: Modelling historic and future land use/land cover changes and their impact on wetland area in Shashe sub-catchment, Zimbabwe, Model Earth Syst Environ, 7, 57–70, https://doi.org/10.1007/s40808-020-00963-y, 2021.

How to use the SNAP API from Python: https://senbox.atlassian.net/wiki/spaces/SNAP/pages/19300362/How+to+use+the+SNAP+API+from+Python, last access: 25 January 2024.

Sun, C., Bian, Y., Zhou, T., and Pan, J.: Using of Multi-Source and Multi-Temporal Remote Sensing Data Improves Crop-Type Mapping in the Subtropical Agriculture Region, Sensors, 19, 2401, https://doi.org/10.3390/s19102401, 2019.

Trucchia, A., Egorova, V., Butenko, A., Kaur, I., and Pagnini, G.: RandomFront 2.3: a physical parameterisation of fire spotting for operational fire spread models – implementation in WRF-SFIRE and response analysis with LSFire+, Geosci Model Dev, 12, 69–87, https://doi.org/10.5194/gmd-12-69-2019, 2019.

Network Common Data Form (NetCDF): https://www.unidata.ucar.edu/software/netcdf/, last access: 25 January 2024.

United Nations Environment Programme & GRID-Arendal: Spreading like Wildfire: The Rising Threat of Extraordinary Landscape Fires https://wedocs.unep.org/20.500.11822/38372, 2022.

Weber, R. O.: Modelling fire spread through fuel beds, Prog Energy Combust Sci, 17, 67–82, https://doi.org/10.1016/0360-1285(91)90003-6, 1991.

Wilson, R.: Reformulation of forest fire spread equations in SI units, Research Note INT-292. Ogden, UT: U.S. Department of Agriculture, Forest Service, Intermountain Range and Forest Experiment Station. 5 p., https://doi.org/10.2737/INT-RN-292, 1980.

Windy: https://windy.app/, last access: 25 January 2024.

Xue, C., Krysztofiak, G., Ren, Y., Cai, M., Mercier, P., Fur, F. Le, Robin, C., Grosselin, B., Daële, V., McGillen, M. R., Mu, Y., Catoire, V., and Mellouki, A.: A study on wildfire impacts on greenhouse gas emissions and regional air quality in South of Orléans, France, Journal of Environmental Sciences, 135, 521–533, https://doi.org/10.1016/j.jes.2022.08.032, 2024.

foronoi: https://github.com/Yatoom/foronoi, last access: 25 January 2024.

Zehra, F., Javed, M., Khan, D., and Pasha, M.: Comparative Analysis of C++ and Python in Terms of Memory and Time, Preprints (Basel), 2020.

Zhang, S., Liu, J., Gao, H., Chen, X., Li, X., and Hua, J.: Study on Forest Fire spread Model of Multi-dimensional Cellular Automata based on Rothermel Speed Formula, CERNE, 27, https://doi.org/10.1590/01047760202127012932, 2021.