UNIVERSIDADE D
COIMBRA

Jorge Fernando Alvarinhas Pereira

Wildfire Spread Prediction Model Calibration Using Metaheuristic Algorithms

Thesis submitted to the University of Coimbra in fulfillment of the requirements for the Master's Degree in Engineering Physics under the scientific supervision of Ph.D. Jérôme Amaro Pires Mendes and Ph.D. Cristiano Premebida.

September of 2022

UNIVERSITY OF COIMBRA

MASTER IN ENGINEERING PHYSICS

Wildfire Spread Prediction Model Calibration Using Metaheuristic Algorithms

Jorge Fernando Alvarinhas Pereira

Supervisors:
Prof. Doutor Jérôme Amaro Pires Mendes
Prof. Doutor Cristiano Premebida

Coimbra, 2022

Esta cópia da tese é fornecida na condição de que quem a consulta reconhece que os direitos de autor são pertença do autor da tese e que nenhuma citação ou informação obtida a partir dela pode ser publicada sem a referência apropriada.

This copy of the thesis has been supplied on condition that anyone who consults it is understood to recognize that its copyright rests with its author and that no quotation from the thesis and no information derived from it may be published without proper acknowledgement.

Agradecimentos

Em primeiro lugar quero agradecer ao meu orientador, Doutor Jérôme Mendes, pelo acompanhamento que me deu no decorrer do projeto. A sua exigência manteve-me no caminho certo e, nos momentos de incerteza, a sua amizade e incentivo mostraram-me a direção correta. Por tudo, obrigado. Agradeço também ao Jorge Sampaio pela ajuda e exemplo que me deu durante o projeto, e aos restantes colegas do laboratório pelas conversas, partilha, e discussões enriquecedoras. Quero ainda agradecer a todos os intervenientes no projeto "IMFire" da ADAI (Associação para o Desenvolvimento da Aerodinâmica Industrial), ADDF (Associação para o Desenvolvimento do Departamento de Física da Universidade de Coimbra) e ISR pois as suas colaborações foram muito importantes na realização do meu trabalho.

Aos meus pais, Alberto Jorge e Maria Teresa, e ao meu irmão André agradeço toda a confiança, apoio e princípios que me transmitiram desde sempre pois são a base dos meus sucessos. Para o meu avô Zé, que perdi durante este percurso, um agradecimento especial por, desde cedo, ter incentivado a minha curiosidade bem como pela educação e princípios que me incutiu.

Agradeço à Mariana por todo o apoio e compreensão que sempre me deu e pela lucidez que me traz nos momentos de maior dificuldade e incerteza.

Agradeço aos meus amigos da faculdade que me aturaram, ajudaram e de quem tantos bons momentos ficam guardados - Rita, Diogo, Rodrigo e ao primo Zé Pedro. Ao Gonçalo, em especial, pela profunda amizade e companheirismo constante ao longo destes anos.

Este trabalho for executado sob o projeto "IMFire - Intelligent Management of Wildfires", ref. PCIF/SSI/0151/2018, totalmente financiado por fundos nacionais através do Ministério da Ciência, Tecnologia e Ensino Superior, Portugal.

Abstract

Every year, wildfires cause significant losses and destruction around the world. In order to aid in their management and mitigate their impact, efforts have been directed towards developing decision support systems that can predict wildfire propagation. In a real wildfire event, these systems provide the authorities with information about the fire propagation in the near future, thus allowing them to make better decisions. Wildfire spread prediction systems are based on fire propagation models, from which the most used and accepted model is the Rothermel model. However, given the complexity of the wildfire phenomena and the uncertainty in the definition of some of its input parameter values, the Rothermel model can produce misleading results of fire propagation.

In this work, three metaheuristic algorithms, genetic algorithm (GA), differential evolution (DE) and simulated annealing (SA), have been implemented for calibration of the Rothermel model's input parameters. First, the one-dimensional Rothermel model was calibrated using the three metaheuristics on 37 datasets containing data from controlled experimental fires. The calibration results were compared against the predictions provided by the non-calibrated Rothermel model and the three metaheuristics were compared in terms of their calibration and time performances. Moreover, a two-stage methodology based on the calibration of the fire spread model and the use of the calibrated parameters for obtaining improved predictions was tested. For this, a two-dimensional fire propagation model based on the Rothermel model was calibrated using the three metaheuristic algorithms. Afterward, the calibration results were used for predicting the fire propagation for a future time instant. Both the calibration and the prediction stages used data from a real controlled prescribed fire and the methodology was compared against the use of the fire propagation model without any calibration.

The results of the calibration of both the one-dimensional Rothermel model and the two-dimensional Rothermel-based fire propagation model showed that differential evolution is a very suitable algorithm to be used in the wildfire spread prediction area, which is predominantly dominated by genetic algorithms. Additionally, the fire spread predictions were significantly improved by the calibration, with reductions in prediction error of more than 80%, in relation the fire spread predictions performed without any previous calibration.

The work developed in this thesis confirmed the quality of genetic algorithms as a calibration algorithm for the Rothermel model and showed the potential of the differential evolution as a very suitable alternative as a calibration algorithm. Moreover, the importance of the two-stage methodology was proven and the fire spread predictions significantly improved.

Keywords: wildfire spread prediction, genetic algorithm, differential evolution, simulated annealing, model calibration.

Resumo

Todos os anos, os incêndios rurais causam inúmeras perdas e destruição em todo o mundo. De forma a auxiliar na sua gestão e a mitigar o seu impacto, têm sido direcionados recursos para o desenvolvimento de sistemas de apoio à decisão que tenham a capacidade de prever a propagação dos incêndios. Durante uma ocorrência real, estes sistemas fornecem informações acerca da propagação do incêndio rural, num horizonte temporal próximo, permitindo assim que as autoridades responsáveis tomem melhores decisões no combate. Os sistemas de predição da propagação de incêndios são baseados em modelos de propagação de fogo, dos quais o mais utilizado e reconhecido é o modelo de *Rothermel*. No entanto, dada a complexidade dos incêndios rurais e a incerteza associada com a definição de alguns dos seus parâmetros de entrada, o modelo de *Rothermel* pode gerar resultados de propagação de fogo pouco exatos.

O trabalho desenvolvido consistiu na implementação de três metaheurísticas - algoritmo genético (GA), evolução diferencial (DE) e recozimento simulado (SA) - para efetuar a calibração de parâmetros de entrada do modelo de *Rothermel*. Inicialmente, o modelo de *Rothermel* foi calibrado pelas três metaheurísticas utilizando 37 datasets de fogos controlados. Os resultados desta calibração foram comparados com as predições resultantes do modelo de *Rothermel* não calibrado e os tempos de calibração necessários foram obtidos, o que permitiu comparar o desempenho das três metaheurísticas. Posteriormente, foi testada uma metodologia baseada na calibração do modelo de propagação de fogo e consequente utilização dos parâmetros calibrados para obter predições futuras. Foi utilizado um modelo de propagação a duas dimensões baseado no modelo de *Rothermel*, tendo este sido calibrado pelas três metaheurísticas. De seguida, os resultados da calibração foram utilizados para obter predições da propagação do fogo para instantes futuros. As fases de calibração e predição foram realizadas utilizando dados de um incêndio controlado e os resultados obtidos foram comparados com as predições provenientes do modelo não calibrado.

Os resultados das calibrações realizadas demonstraram que a evolução diferencial é um algoritmo bastante adequado para a área da predição da propagação de incêndios, área onde predominam os algoritmos genéticos. Adicionalmente, as predições da propagação do fogo melhoraram significativamente quando precedidas de calibração, tendo-se verificado reduções no erro de predição de mais de 80% em relação às predições obtidas sem ser realizada calibração.

O trabalho desenvolvido nesta tese comprovou a competência do algoritmo genético como algoritmo de calibração do modelo de *Rothermel* e demonstrou o potencial do algoritmo evolução diferencial como um algoritmo de calibração alternativo. Além disto, a importância da metodologia baseada na calibração e predição ficou também patente pelas melhorias significativas verificadas nas consequentes predições da propagação de fogo.

Palavras-chave: predição de propagação de incêndios, algoritmo genético, evolução

diferencial, recozimento simulado, calibração de modelos.

Contents

List of Acronyms **ix**

List of Figures **xi**

List of Tables **xiii**

1 Introduction **1**
1.1 Context and Motivation . . . 1
1.2 Problem formulation . . . 1
1.3 Objectives and main contributions . . . 2
1.4 Requirements analysis . . . 3
1.5 Structure of the thesis project . . . 3

2 Wildfire spread prediction and literature review of calibration of prediction models **5**
2.1 The Rothermel model . . . 5
2.2 The need for calibration of fire spread models . . . 6
2.3 Wildfire spread prediction calibration literature overview . . . 8
2.3.1 Wildfire spread calibration literature using genetic algorithms . . . 8
2.3.2 Wildfire spread calibration implementing parallel computing . . . 12
2.4 Literature review summary . . . 15

3 Overview of the implemented metaheuristic algorithms for wildfire spread model calibration **19**
3.1 Notation and definitions . . . 19
3.2 Genetic algorithms . . . 20
3.3 Differential evolution . . . 21
3.4 Simulated Annealing . . . 22

4 Calibration of the Rothermel model **25**
4.1 Calibration methodology . . . 25
4.1.1 Solution structure: input parameters to be calibrated . . . 25
4.1.2 Fitness function . . . 26
4.1.3 Calibration algorithm . . . 26
4.2 Results . . . 27
4.2.1 Datasets . . . 27
4.2.2 Results analysis . . . 28
4.3 Conclusions . . . 33

5 Calibration of fire spread prediction model **35**
5.1 Fire spread simulator 35
5.2 Methodology 36
5.2.1 Solution structure 37
5.2.2 Fitness function 37
5.2.3 Calibration and prediction methodology 38
5.3 Results 39
5.4 Conclusions 46

6 Conclusions **47**
6.1 Conclusions 47
6.2 Future work 48

Appendix A Paper published on Mathematics journal **55**

Appendix B Paper accepted on the IECON2022 conference **75**

List of Acronyms

S²F²M Statistical System for Forest Fire Management.

ADAI Associação para o Desenvolvimento da Aerodinâmica Industrial.

ADDF Associação para o Desenvolvimento do Departamento de Física.

DDDGA Dynamic Data-Driven Genetic Algorithm.

DE Differential Evolution.

DEM Digital Elevation Map.

FCCS Fuel Characteristics Classification System.

GA Genetic Algorithm.

ISR Institute of Systems and Robotics.

NFFL Northern Forest Fire Laboratory.

RMSE Root Mean Square Error.

SA Simulated Annealing.

SAPIFE Sistema Adaptativo para la Prediccin de Incendios Forestales basados en Estratgias Estadstico-Evolutivas.

WFA WildFire Analyst.

List of Figures

2.1 Illustration of fire spread prediction using only one set of non-calibrated input parameters. Adapted from [24]. 7
2.2 Two-stage framework for fire spread prediction, adapted from [24]. 9
2.3 Genetic algorithm using the Master/Worker paradigm, adapted from [38]. 13

4.1 Representation of a candidate solution, corresponding to four input parameters. 26
4.2 Calibration results from Algorithm 4: comparison of the GA-calibrated (Algorithm 1), DE-calibrated (Algorithm 2), and SA-calibrated (Algorithm 3) models against the non-calibrated Rothermel model, for every dataset. 29
4.3 Iteration of first occurrence of the best fitness value, for each algorithm and dataset. 30

5.1 Illustration of how various layers which describe the location where the fire occurs and serve as input for the fire spread simulator. 36
5.2 Illustrations of the symmetric difference between two sets A and B. . . . 37
5.3 Images regarding the prescribed fire used as test case for this work. . . . 38
5.4 Scenario 1: fire spread predictions for $t_2 = 8\,min$, with calibration performed using the real fire area from $t_1 = 4\,min$. 43
5.5 Scenario 2: fire spread predictions for $t_2 = 12\,min$, with calibration performed using the real fire area from $t_1 = 4\,min$. 44
5.6 Scenario 3: fire spread predictions for $t_2 = 12\,min$, with calibration performed using the real fire area from $t_1 = 8\,min$. 45

List of Tables

2.1 Identification of the parameters in Rothermel model's equations. 7
2.2 Review of the literature on wildfire spread prediction calibration using genetic algorithms. 17

4.1 Parameter settings for the calibration methodology, Algorithm 4, using GA (Algorithm 1), DE (Algorithm 2), and SA (Algorithm 3). 28
4.2 Results of the proposed calibration algorithm (Algorithm 4) using: GA (Algorithm 1), DE (Algorithm 2), and SA (Algorithm 3). 30
4.3 Calibration results of σ and δ, for the three metaheuristics and for each dataset. 31
4.4 Comparison of the calibration results of M_f and U, for the three metaheuristics. 32

5.1 Parameter settings for the calibration methodology, Algorithm 5, using GA (Algorithm 1), DE (Algorithm 2), and SA (Algorithm 3). 40
5.2 Calibration and prediction results from the three algorithms when performing calibration using fire data of $t_1 = 4min$. 41
5.3 Calibration and prediction results from the three algorithms when performing calibration using fire data of $t_1 = 8\,min$.. 41
5.4 Fire spread prediction results using the default (non-calibrated) values of σ and δ (σ' and δ'). 42

Chapter 1

Introduction

1.1 Context and Motivation

This thesis was carried out in the scope of a research fellowship in the Institute of Systems and Robotics (ISR) related to the project "IMFire - Intelligent Management for Wildfires". The goal of the "IMFire" project is to develop a web-based platform dedicated to civil protection for the management of wildfires. The "IMFire" platform covers three important targets: prevention of wildfire occurrences, planning based on prediction of wildfire propagation, and the improvement of the effectiveness of firefighting strategies. The work developed for this thesis fits in the second target, the prediction of wildfire propagation.

Every year, wildfires provoke significant losses and devastation throughout the world. According to the latest European Commission's annual report on wildfires, which is for the year 2020, fires over 30 ha were observed in 39 countries from Europe, the Middle East, and North Africa [1, 2, 3]. These wildfires have resulted in a total burnt area of $1,075,145$ ha for 2020, which is approximately 35% larger than the area registered in 2019. Furthermore, in a 2022 report published by the United Nations Environment Programme, it is estimated that the probability of catastrophic wildfire events will increase by a factor of 1.57, by the end of the century [4]. Furthermore, wildfires affect ecosystems by destroying natural habitats, resources, and wildlife. They are also responsible for fatalities, injuries, health problems, and destruction of human infrastructures. Consequently, these problems result in a huge economic impact [5].

It is therefore crucial to assign resources to wildfire investigation and management. As stated before, this work will focus on the prediction of wildfire propagation. Wildfire spread prediction has tremendous importance because it allows authorities to identify the areas affected by the wildfire in advance, and take appropriate measures based on that information.

1.2 Problem formulation

Several mathematical models have been developed for understanding fire behavior [6]. Generally, these models consist of sets of equations which result in numerical solutions of certain variables (e.g., linear velocity of the fire, height of the flame) that provide insight on the evolution of the fire in space and time. According to [6], the most used and

accepted model of fire[1] spread is the Rothermel model [7], especially in Mediterranean European countries [8]. Additionally, the Rothermel model is at the core of some of the most cited fire spread simulators[2] such as FARSITE [9], FIRESTATION [10], and fireLib [11]. It was an early-defined project constraint that the work developed should be focused only on the Rothermel fire spread model.

The Rothermel model consists of a set of equations which leads to a final equation that calculates the rate of spread R of a fire front - the rate of spread R is equivalent to the value of the linear velocity of the fire front. For calculating the rate of spread, the model uses a set of input parameters related to the fuel, terrain and atmospheric conditions from the location of the fire. Additionally, fire simulators such as FARSITE [9] and FIRESTATION [10] implement the Rothermel model in a more advanced and also practical way, by using a raster approach. Simulators such as these are very important because of their graphical components, which allow for a visual representation of the fire propagation. However, despite being the "most widespread and practical mathematical model to date" [6], the Rothermel model can produce inexact results. According to [12], the three main causes for divergence between real fire propagation and fire model propagation predictions are the following: the model's lack of applicability to the scenario at hand, the model's intrinsic lack of prediction quality, and the inaccuracy in the estimation of the input parameter's values. Moreover, according to [13], even if a perfect mathematical model for fire propagation prediction existed, the uncertainties associated with spatial and temporal variations of fuels, topography and weather conditions - which generate poor estimations of the input parameter's values - would result in erroneous fire behavior predictions.

Considering the the above information and the facts described in Section 1.1, it becomes well established that the inaccuracy associated with the Rothermel model's input parameters is an important problem. Moreover, metaheuristic algorithms such as Genetic Algorithms (GA), Ant Colony Optimization (ACO), Simulated Annealing (SA) and Particle Swarm Optimization (PSO) have proven their effectiveness for many different optimization/calibration problems [14, 15, 16, 17]. In this way, to improve the quality of wildfire spread predictions, methodologies based on metaheuristic algorithms should be studied [18, 19, 20].

1.3 Objectives and main contributions

The objective of this work is to improve accuracy of the fire propagation prediction by calibrating the Rothermel model's input parameters.

The main contributions of this thesis were:

- A literature review of wildfire spread prediction calibration based on genetic algorithms (Chapter 2). The vast majority of the works found for wildfire spread prediction calibration use genetic algorithms for calibrating the Rothermel model. Hence, the search process and literature review are focused on GA-based works.

[1] The term "fire" will be used throughout this thesis in certain parts, as opposed to "wildfire". "Fire" generally refers to a planned and controlled combustion, which is used frequently in laboratories or open fields, for controlled fire experiments. The term "wildfire" refers to an uncontrolled combustion, which is dangerous and has potential consequences such as the ones described in Section 1.1.

[2] As stated, a considerable number of fire spread simulators implement the Rothermel model as the main fire spread model. In this sense, during this thesis, when a reference is made to the Rothermel model (e.g., the Rothermel model's input parameters), the cited fire spread simulators should be considered as well.

The review resulted in the article "A Review of Genetic Algorithm Approaches for Wildfire Spread Prediction Calibration" [21], published in January of 2022.

- Implementation and comparison of three metaheuristic algorithms, genetic algorithms, differential evolution and simulated annealing, for the calibration of the Rothermel model's input parameters (Chapter 4). Datasets from real prescribed fires were used. This work resulted in the accepted conference paper "Wildfire Spread Prediction Model Calibration Using Metaheuristic Algorithms" [22] on the 48th Annual Conference of the IEEE Industrial Electronics Society (IECON 2022).

- Adaptation of the three developed algorithms (GA, DE, and SA) for input parameter calibration of a two-dimensional (2D) Rothermel-based fire spread model with consequent predictions using the calibration results. Data from a prescribed fire was used.

1.4 Requirements analysis

The functional requirements of the developed algorithms are to obtain calibrated sets of input parameters of the Rothermel model. Additionally, the algorithms' code must allow the users to choose the model's input parameters to be calibrated and define the algorithms' parameters.

The nonfunctional requirements are the following:

- The algorithms have to be implemented using parallel computing, when possible;

- The calibrated model must improve the average prediction error by at least 40 %;

- The average model calibration time must be inferior to 30 minutes.

1.5 Structure of the thesis project

This thesis is organized into five more chapters.

Chapter 2 - Wildfire spread prediction and literature review of calibration of prediction models presents the literature review on wildfire spread prediction and calibration of prediction models.

Chapter 3 - Overview of the implemented metaheuristic algorithms for wildfire spread model calibration is dedicated to the description of the notation and of the three metaheuristic algorithms implemented in this work.

In **Chapter 4 - Calibration of the Rothermel model** the calibration methodology and the calibration results of the Rothermel model are shown.

Chapter 5 - Calibration of fire spread prediction model presents the application of the metaheuristic algorithms for calibration of a wildfire spread model, and the calibration and fire prediction results.

In **Chapter 6 - Conclusions**, the final conclusions are drawn and the future work is discussed.

Appendix A and **Appendix B** contain the published journal paper and the accepted conference paper, respectively, which were concluded during the development of this thesis.

Chapter 2

Wildfire spread prediction and literature review of calibration of prediction models

This chapter presents the theoretical basis for the work developed and the literature review. Section 2.1 describes the Rothermel model, which is the fire spread model to be calibrated. Section 2.2 explains the reasons behind the need for calibration of fire spread models. Finally, Section 2.3 presents a detailed state-of-the-art review. The literature review presented in this chapter resulted in the published article [21], which consequently, makes this chapter considerably coincident with the article's content.

2.1 The Rothermel model

The Rothermel model, proposed in [7], estimates a Rate Of Spread R of a fire front, given by

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

which is measured in units of distance per unit of time ([m/s] or [ft/min]), and it represents the linear velocity of a fire in the main direction of propagation and in a given set of conditions. The equations of the associated factors in (2.1) $I_R(\rho_p, \sigma, \delta, w_0, S_T, h, M_x, M_f, S_e)$, $\xi(\sigma, \rho_p, w_0, \delta)$, $\phi_w(\rho_p, w_0, \delta, \sigma, U)$, $\phi_s(\rho_p, w_0, \delta, tan\phi)$, $\rho_b(w_0, \delta)$, $\varepsilon(\sigma)$, and $Q_{ig}(M_f)$ depend on several input parameters and are given by (2.2) to (2.20).

The input parameters of the Rothermel model (2.1) are identified in Table 2.1 and can be separated into three categories: fuel properties, topography and wind properties. The fuel properties are heat content (h), mineral content (S_T (total) and S_e (effective)), oven-dry particle density (ρ_p), oven-dry fuel load (w_0), surface-area-to-volume ratio (σ), fuel bed depth (δ), dead fuel moisture of extinction (M_x) and fuel moisture (M_f). Topography is represented by slope steepness (tanϕ), and wind properties correspond to the midflame wind speed (U). A deeper insight into the Rothermel model can be gained in [7, 23].

$$
\begin{aligned}
I_R &= \Gamma' w_n h \eta_M \eta_S && (2.2)\\
\Gamma' &= \Gamma'_{max} \left(\frac{\beta}{\beta_{op}}\right)^A exp\left[A\left(1-\frac{\beta}{\beta_{op}}\right)\right] && (2.3)\\
A &= 133\sigma^{-0.7913} && (2.4)\\
\beta &= \frac{\rho_b}{\rho_p} && (2.5)\\
\rho_b &= \frac{w_0}{\delta} && (2.6)\\
\Gamma'_{max} &= \frac{\sigma^{1.5}}{(495+0.0594\sigma^{1.5})} && (2.7)\\
\beta_{op} &= 3.348\sigma^{-0.8189} && (2.8)\\
w_n &= w_0(1-S_T) && (2.9)\\
\eta_M &= 1-2.59r_M+5.11(r_M)^2-3.52(r_M)^3 && (2.10)\\
r_M &= \frac{M_f}{M_x}\,(max=1.0) && (2.11)\\
\eta_S &= 0.174S_e^{-0.19}\,(max=1.0) && (2.12)\\
\xi &= \frac{exp[(0.792+0.681\sigma^{0.5})(\beta+0.1)]}{(192+0.2595\sigma)} && (2.13)\\
\phi_w &= CU^B\left(\frac{\beta}{\beta_{op}}\right)^{-E} && (2.14)\\
C &= 7.47exp(-0.133\sigma^{0.55}) && (2.15)\\
B &= 0.02526\sigma^{0.54} && (2.16)\\
E &= 0.715exp(-3.59\times10^{-4}\sigma) && (2.17)\\
\phi_S &= 5.275\beta^{-0.3}(tan\phi)^2 && (2.18)\\
\varepsilon &= exp\left(\frac{-138}{\sigma}\right) && (2.19)\\
Q_{ig} &= 250+1116M_f && (2.20)
\end{aligned}
$$

2.2 The need for calibration of fire spread models

Figure 2.1 presents a general illustration for wildfire spread prediction, which consists in feeding a fire simulator with a set of input parameters that aim to represent the initial real fire conditions, at t_0. The result of the fire simulator, i.e. the simulated wildfire perimeter, at t_1, should match the propagation of the real wildfire, i.e. the real wildfire perimeter [25]. However, the input parameters are related to the environmental conditions, e.g. fuel, weather, and terrain characteristics as described in Section 2.1, and obtaining them becomes a difficult task in order to provide an accurate prediction.

In more detail, some input parameters can be directly measured, such as terrain slope, which can also be obtained based on previous topographical information. But other parameters, such as fuel-specific parameters, require detailed knowledge about the local vegetation, which might not be available. Some input parameters, such as fuel moisture, are calculated using models based on meteorological data [26], while wind field maps are

Table 2.1: Identification of the parameters in Equations (2.2) to (2.20) [7, 23].

Parameter	Description
I_R	Reaction intensity (Btu/ft^2min)
Γ'	Optimum reaction velocity (min^{-1})
β	Packing ratio
ρ_b	Oven-dry bulk density (lb/ft^3)
Γ'_{max}	Maximum reaction velocity (min^{-1})
β_{op}	Optimum packing ratio
w_n	Net fuel load (lb/ft^2)
η_M	Moisture damping coefficient
η_S	Mineral damping coefficient
ξ	Propagating flux ratio
ϕ_w	Wind factor
ϕ_S	Slope factor
ε	Effective heating number
Q_{ig}	Heat of preignition (Btu/lb)

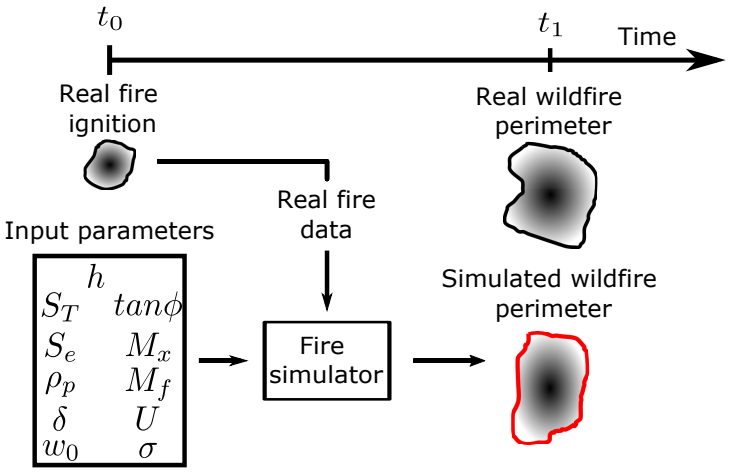


Figure 2.1: Illustration of fire spread prediction using only one set of non-calibrated input parameters. Adapted from [24].

estimated based on point observations from the available meteorological stations closer to the fire location. These estimations introduce a great amount of error in the prediction. In terms of behavior change, characteristics such as the terrain slope and the type of vegetation in a certain region are constant in time and space, while others, such as wind speed and direction, have very sudden variations during the wildfire [27]. Therefore, finding a set of input parameters that produces accurate results solely based on previous knowledge about the wildfire location and weather conditions is a very difficult task. Due to the uncertainty and the consequent inaccuracy in wildfire spread simulation, there is a need to calibrate the input parameters.

2.3 Wildfire spread prediction calibration literature overview

The search process for the presented literature review was performed by using the Science Direct[1] and IEEE Xplore[2] databases and defining the following search keywords: ("fire spread" OR "fire prediction" OR "fire rate of spread" OR "Rothermel model") AND ("genetic algorithm" OR "evolutionary algorithm" OR "calibration" OR "tuning"). The years considered for the search were from 2000 until 2021. Additionally, the references of the selected papers were also analyzed and served as source for finding new papers. The literature review rationale for article selection was based on the following criteria:

- Acceptance
    1. The article uses the Rothermel model or a Rothermel model based simulator for fire propagation prediction/simulation;
    2. The article uses evolutionary algorithms for Rothermel model calibration;
    3. The article focuses on improving the prediction results or its execution time.
- Rejection
    1. The article's method for fire propagation prediction is not based on the Rothermel model;
    2. The article implements calibration techniques other than evolutionary algorithms.

Based on the above search process, 15 papers were obtained.

In the following Sections 2.3.1 and 2.3.2, the main works dealing with wildfire spread prediction calibration are presented, providing a perspective of the philosophy currently being pursued in this research field.

2.3.1 Wildfire spread calibration literature using genetic algorithms

Genetic algorithms have been used to optimize fire spread models (particularly the Rothermel model), i.e., to find the set of input parameters that better adjusts the wildfire spread model predictions to the real observations.

The authors in [28] introduced a framework, illustrated in Figure 2.2, that consists of two stages: a calibration stage and a prediction stage. After the ignition, the calibration stage starts, at t_0. Sets of Rothermel's input parameters are generated and each one is evaluated, at instant t_1, by comparing the respective simulator propagation prediction with the real observed fire data for that time instant. The optimal set of input parameters is the one that minimizes the deviation between the predicted and the real fire perimeter. This process is repeated several times or until a certain solution criterion is reached. In the prediction stage, assuming that environmental conditions remain constant, the resulting optimal set of parameters from the calibration stage is used as input for the fire simulator to predict the fire spread at a following time instant t_2. Here, the prediction stage is

[1] `https://www.sciencedirect.com`

[2] `https://ieeexplore.ieee.org/Xplore/home.jsp`

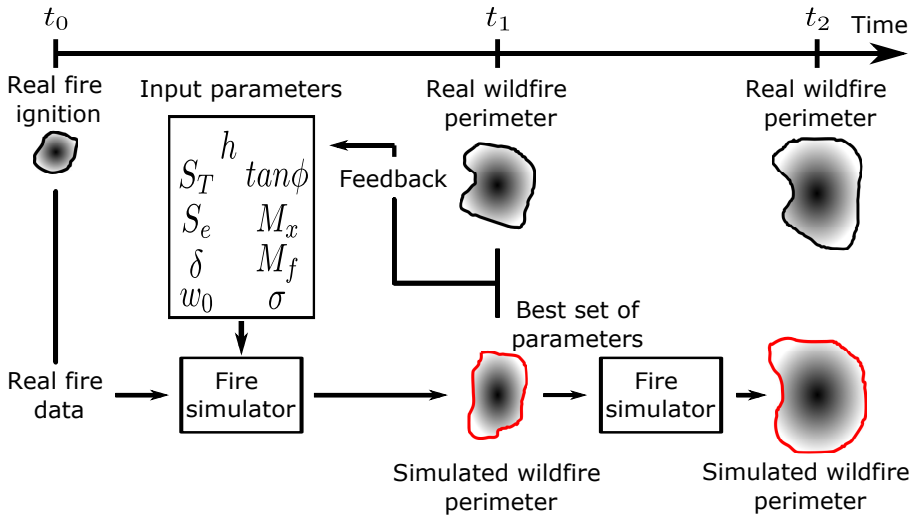


Figure 2.2: Two-stage framework for fire spread prediction, adapted from [24].

similar to the classical method/framework (Figure 2.1), except that now a tuned set of input parameters is used.

During the framework's calibration stage, the goal is to find an optimal solution for the input parameters. In a generic way, the optimization problem can be defined as:

$$\mathbf{x}^* = \arg\min_{\mathbf{x} \in S} F(\mathbf{x}), \tag{2.21}$$

where $F(\mathbf{x})$ represents the function to be minimized (by an optimization algorithm, such as GA), $\mathbf{x}$ represents the input parameters vector, S is the respective search space, and $\mathbf{x}^*$ represents the input parameters that minimize $F(\mathbf{x})$. A usual function to be optimized in wildfire spread calibration is the difference between the real wildfire rate of spread (measured from the real-time wildfire data) and the predicted rate of spread (obtained by the Rothermel model), or the difference between the real and the predicted burned area. The goal is to find the set of input parameters $\mathbf{x}$ of (2.21) that most accurately predicts the real fire propagation.

The majority of the works from the current state of the art on wildfire spread prediction calibration are based on the previously presented two-stage framework (Figure 2.2). Early works, such as [18] and [28], have proposed evolutionary algorithms as techniques that could be used to find an optimal set of input parameters for a fire simulator. Genetic algorithms are included in the group of evolutionary algorithms and, according to the performed search, they are the dominant optimization technique for input parameters calibration regarding the Rothermel model.

In [28], following the presentation of the two-stage framework, a sensitivity analysis was carried out in order to evaluate how the individual variation of each Rothermel input parameter across its range of possible values affects the model output: the bigger the sensitivity of one parameter, the more it affects the model's output. Based on the sensitivity results, an experimental study was conducted to confirm that calibrating parameters with larger sensitivities and fixing the others reduces the GA's search space and accelerates the optimization time. The results showed that, after 1000 generations, the scenarios in which only 6 input parameters were calibrated achieved an improvement in the objective function of approximately 33.3% in relation to the scenario in which 10 input parameters were calibrated. This reduction also matches the reduction in GA's search space from one scenario to the other.

In [18], the genetic algorithm's performance is tested against three other algorithms: random search, tabu search and simulated annealing. The tests were carried out by comparing the simulated fire line based on the sets of parameters generated by the algorithms against a fire line obtained by setting known values for all the inputs and running the *ISStest* simulator for 45 minutes. Each algorithm was executed 10 times up to 1000 iterations. The fire lines were compared using the Hausdorff distance H (2.22), which measures the degree of mismatch between two sets of points F_1 and F_2, representing the fire line simulated based on the optimized parameters and the fire line generated with known input parameters for comparison. H (2.22) is given by

$$H(F_1, F_2) = \max(h(F_1, F_2), h(F_2, F_1)), \tag{2.22}$$

where $h(F_1, F_2)$ and $h(F_2, F_1)$ represents the Hausdorff distance between two sets of points F_1 and F_2 at a specific point in F_2 and F_1, respectively ([18] contains more details on this). The results show that simulated annealing, tabu search and genetic algorithms presented similar results after the 500th generation.

In [25], a Dynamic Data-Driven Genetic Algorithm (DDDGA) was proposed to tune the fire simulator's input parameters based on the real fire behavior. The used simulator was fireLib and, through reverse engineering, it was possible to obtain equations for wind values (wind speed and direction). These equations are fed with terrain slope with the position (x, y) of the fire front with the maximum rate of spread. The obtained wind speed and direction values were used to steer the search for an optimal input parameter set carried out by the genetic algorithm. Afterward, in [29], the same research group proposed a new calibration steering method as an improvement to the previous strategy. Since this was highly dependent on the underlying simulator, the new approach consisted in generating a database with fire evolution information from both real and simulated (synthetic) fires. For the calibration stage, a DDDGA was proposed to define the best wind direction and wind speed values, by searching the database of previous fires that were similar in terms of rate of spread, slope and fuel model to the real observed fire spread, and using wind values from those fires to steer the genetic algorithm's search.

The authors in [24] introduced a system called SAPIFE (Spanish acronym for Adaptive System for Fire Prediction Based in Statistical-Evolutive Strategies) which is based on the two-stage fire spread prediction framework with a genetic algorithm implemented during the calibration stage. However, in SAPIFE, the genetic algorithm is coupled with another method called Statistical System for Forest Fire Management (S^2F^2M). This new method receives a certain population from the GA and analyzes almost all possible input parameter combinations from all individuals in the population. From this analysis, S^2F^2M evaluates the probability of each map cell to be burned or not and generates a probabilistic map. Then, based on these probabilities, the number of possible scenarios (parameter combinations between different individuals) is reduced, decreasing the calibration time required.

In [19], the two methods introduced in [25] and [29] were compared. The method introduced in [25] is named as "analytical method" and, as it was described above, based on the inversion of a fire simulator. The method introduced in [29] is named "computational method" and relies on a database with information from past fires. Both of these methods use ongoing fire propagation data to obtain wind speed and direction values and use it to steer the genetic algorithm's search. Two sets of tests were carried out: first, the two-stage framework was tested against the classical wildfire spread prediction method, which uses a single set of input parameters introduced in the fire simulator. This test used data from

past fires and confirmed that the two-stage framework with a genetic algorithm provides better results than the classical prediction without input parameter calibration. Then, the second set of tests compared the use of a simple non-guided genetic algorithm against genetic algorithms with different configurations of the proposed steering strategies. The guided genetic algorithm with the computational and analytical methods obtained similar results and improved prediction quality over the non-guided genetic algorithm.

The work developed by [27] is also based on the two-stage prediction framework with a genetic algorithm and introduces an approach for reducing the prediction errors caused by the variability of wind parameters (wind speed and direction). During the calibration stage, wind parameters are not calibrated; instead, real wind measurements from the fire location are taken in periodic time sub-intervals. These measurements are used as inputs for the fire simulator in the recurring simulations. Afterward, during the prediction stage, a numerical weather prediction model [30] is used to periodically estimate the wind parameters between sub-intervals of the prediction stage. The estimated wind parameters are introduced in the simulator and are updated at each sub-interval. The prediction result is obtained using the real wind measurements and the calibrated parameters, which are moisture contents and vegetation features. The test results showed that, when the wind conditions are stable, the basic two-stage framework with a genetic algorithm provides satisfactory results, in comparison with the new method of using measured and estimated wind values (prediction error of 0.4 vs. 0.29, respectively). However, when the wind conditions are more dynamic, the results obtained by the introduced method are significantly better compared to the basic two-stage framework with a genetic algorithm (prediction error of 0.19 vs. 0.58, respectively).

In [20], a calibration of the fuel models within the Rothermel's fire spread prediction model was carried out through the use of genetic algorithms. The GA's individuals consisted of the following Rothermel fuel parameters: oven-dry fuel load (w_0), surface-area-to-volume ratio (σ), fuel bed depth (δ), fuel moisture of extinction (M_x), and heat content (h). Two tests were performed to evaluate the proposed GA method. The first test consisted of using genetic algorithms for the fuel model calibration method, with the support of two works [31, 32] (grass and shrub fuels, respectively) that provided datasets of observed rate of spread R and other input parameters' data (fuel moisture, wind speed and slope steepness). The GA was configured for 9999 maximum iterations, 100 individuals, mutation probability and elitism factor equal to 0.1 and 0.05, respectively, and calibrated the fuel input parameters based on the parameter ranges given by the papers. Each individual was evaluated using the Root Mean Square Error (RMSE) between the observed and predicted rate of spread R. The second test consisted of implementing the GA for calibrating a fuel model for a type of vegetation (*Calluna* heath). Nine prescribed fire experiments were carried out in dry *Calluna* heathland vegetation and R, fire weather (1h fuels moisture, live woody fuel moisture and wind speed) and terrain data (ignition line length, fire plot size and slope) were recorded from each experiment. From the nine fire experiments, 4 were considered for GA calibration and 5 were considered for validation. The data from the calibration experiments was used to run the GA and calibrate the fuel parameters, similarly to the first test. Then, predicted rate of spread R values were calculated using different fuel models: GA calibrated fuel parameters, the Standard Fuel Model which provided the smaller RMSE when comparing predicted *vs.* observed R, a custom fuel model for *Calluna* vegetation and a "custom fuel model parameterized with modal values from fuels inventoried in each fire experiment". An additional prediction of the rate of spread R was obtained by a Rothermel model reformulation implemented in

Fuel Characteristics Classification System (FCCS) [33]. For the validation experiments data, the calibrated GA fuel parameters resulted in the lowest RMSE between predicted and observed rate of spread R, in comparison to the alternative models.

The study in [34] presents a dynamic data-driven genetic algorithm and introduces a new approach for predicting fire propagation based on WildFire Analyst (WFA) [35]. The paper describes the two-stage prediction framework with a genetic algorithm, where the fire propagation is simulated using the FARSITE fire simulator [9], and the fitness function corresponds to the symmetric difference between predicted and burned areas obtained by:

$$Difference = \frac{UnionCells - IntersectionCells}{RealCells - Init\ Cells}, \tag{2.23}$$

where *UnionCells* represents the sum of the number of cells that were burned in the predicted area and the real area, *IntersectionCells* is the number of cells burned simultaneously in the predicted area and the real area, *RealCells* is the final number of cells burned in the real area, and *InitCells* is starting number of cells burned in the real fire area. The newly introduced approach uses WildFire Analyst (WFA) and seeks the best R adjustment factors, minimizing the error between simulated fire and the real fire data. Both the FARSITE fire simulator and Wildfire Analyst use the Rothermel model. Afterward, the two-stage framework with the genetic algorithm and Wildfire Analyst are coupled together by overlapping their predicted fire spread maps. In order to test the two-stage framework and Wildfire Analyst, experiments were carried out with data from a real fire that occurred in Cardona, Catalonia, Spain in 2005. The results show that both methods adapt to drastic changes in the fire characteristics.

In [36], the two-stage framework was considered to reduce input parameter uncertainty and predict fire spread. However, when the wildfire is large, wind cannot be considered uniform throughout the whole wildfire area. So, this work introduced a wind field model (WindNinja), being represented by a cell map, to account for this variation. In essence, during the calibration phase, the obtained meteorological wind parameters are used to calculate the wind field for each scenario generated by the genetic algorithm. Then, having each individual's wind field, the corresponding fire propagation map is calculated and the error function is evaluated.

Finally, in [37], a statistical study was carried out to characterize the genetic algorithm in the calibration phase of the two-stage prediction method. The characterization refers to estimating which GA parameter configuration results in a better calibration within the imposed time restrictions. A statistical study was conducted based on the results of a genetic algorithm calibration on a simulated 5-hour fire obtained using FARSITE as the fire spread simulator. The results from this study were maximum adjustment errors which have different degrees of guarantee depending on the number of generations that the GA iterates. These results are important in understanding the compromise between the algorithm's execution time (number of generations) and the adjustment error, which is larger when the algorithm iterates fewer generations.

2.3.2 Wildfire spread calibration implementing parallel computing

Throughout Subsection 2.3.1, several works regarding fire spread prediction using genetic algorithms were described. Despite their focus being on improving prediction

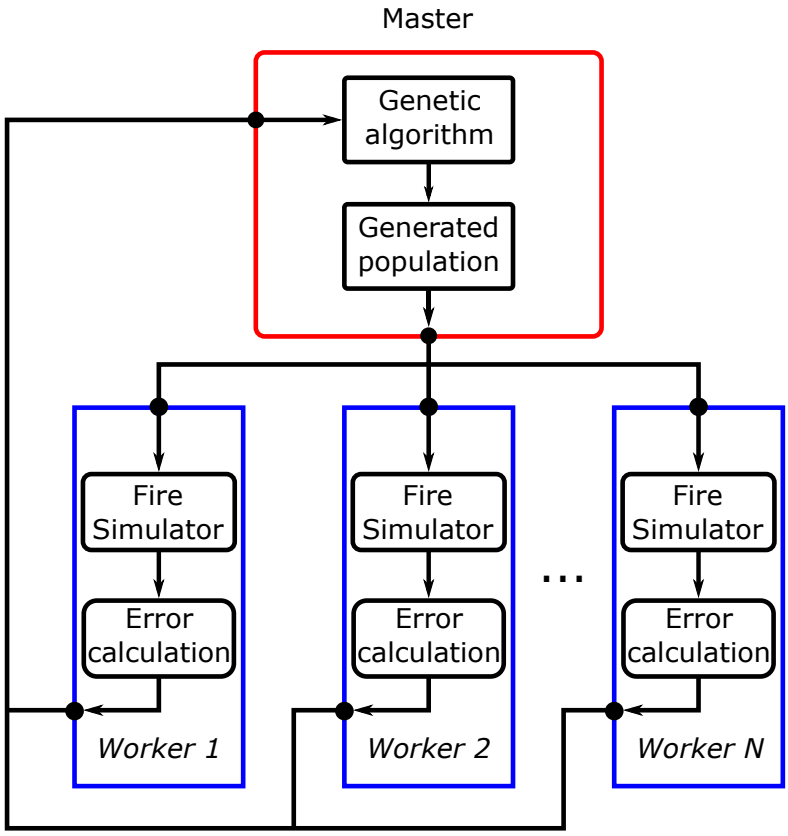


Figure 2.3: Genetic algorithm using the Master/Worker paradigm, adapted from [38].

accuracy, some works have proposed and adapted a Master/Worker paradigm (Figure 2.3) in order to reduce the calibration and prediction times.

Genetic algorithms, as any evolutionary algorithm, require the execution of a set of individual simulations through several iterations, which can be very time-consuming, and given the urgency and need for accuracy associated with wildfire spread prediction in real-time, it is important to reduce the execution time of the calibration phase while maintaining appropriate accuracy. One way to achieve this is through the parallel implementation of the used fire spread simulator. Parallel computing consists in breaking down large and complex problems into various smaller tasks and executing them simultaneously. Parallel computing takes advantage of multi-core computer architecture based on having two or more processing units (cores) in a single chip computer processor. A system with multi-core processing has the advantage of completing more tasks while consuming less or the same amount of energy. Given the better performance and efficiency, it is desirable to use multi-core processing when executing demanding computational tasks.

The authors in [39] presented a technique based on the parallel implementation of both the GA (used in the two-stage fire prediction framework) and the FARSITE fire simulator. An analysis of the FARSITE fire simulator was carried out, and one of its functions, *CrossCompare()*, was implemented in parallel using OpenMP[3]. For analysing the simulator's performance, a simulated fire was executed using a different number of cores, i.e. 2, 4, 6, and 12. The serial tasks of FARSITE take roughly the same time for the different number of cores. However, the parallel tasks are executed much faster with more cores: from 220 seconds with one core to less than 20 seconds with 12 cores. Afterward,

[3] `https://www.openmp.org/`

the two-stage framework with the simulator and the genetic algorithm implemented in parallel was tested by carrying out three experiments. Experiment A consisted of running the GA with 25 individuals for 10 generations with 25 cores available (one core for each individual). Experiment B was executed in the same conditions but with 4 cores available per individual. Finally, experiment C comprised a GA with 100 individuals and 100 cores (one core per individual). For each experiment, new reference fires were simulated for 20 seconds in FARSITE, using as base a terrain in Cap de Creus, Catalonia, Spain. For the first experiments, with fire simulations of 20 seconds, the results showed an improvement in GA execution time for reaching the same error (15%) when using more cores per individual. When replicating the experiments with longer fire simulations (120), the results showed that using more cores per individual still improved execution times for achieving the same error (approximately 14%). However, for the longer fire simulations, using more individuals (100) with one core per individual achieved the lowest error (approximately 8%).

Despite the strategy introduced in [39] improving the calibration time, there's still a drawback related to GA implementation. During the calibration phase, all of the GA individuals have to be simulated. The execution time of a fire simulation depends on the input parameters and, given the random nature of the generation of the population, some individuals will result in much longer simulation times than others. It would be possible to reduce the overall calibration time by dedicating more computing resources to the individuals with larger execution times and fewer resources to individuals that are executed faster. In order to achieve the said time reduction, it is necessary to predict each individual's simulation time to provide more computing resources to those whose predicted execution time is larger. The prediction must be based only on the individuals' genes - a set of input parameters. The study in [39] refers to [40], which introduces a method based on Decision Trees to characterize a fire simulator, allowing to estimate the execution time of one simulation, given a set of input parameters.

In [41], the method based on Decision Trees referenced in [39] is implemented and tested: Decision Trees are employed to classify each fire simulation based on a training dataset composed of several simulations. Each training simulation is classified according to its execution time so that the Decision Trees can label a new simulation by comparing its input parameters against those of the training simulations. In this work, four different execution time ranges are used, labeled as A, B, C, and D, where A represents the shortest execution time, and D represents the longest execution time. The FARSITE fire simulator is used and implemented in parallel using OpenMP. The core-allocation policy ensures that the individuals labeled with a longer execution time classification are simulated using more computing cores: A - One core, B - Two cores, C - Four cores, and D - Eight cores. To test this policy, topographic data from Cap de Creus, Spain was used and a fire was simulated. Then, a GA was executed to find a set of input parameters that would replicate the fire. It evolved 50 different populations for 10 generations, each with 25 individuals. The results showed that using the core-allocation policy reduced the execution time in 41%, in relation to not using any policy.

In [42], similarly to what was done in [41], GA individuals are labeled according to their estimated simulation time through the use of Decision Trees - A, B, C, D and E. The objective is to implement a core-allocation policy that can devote more cores to individuals whose execution times are longer. Additionally, in this work, an additional restraint is imposed: each GA generation has a limited amount of time to be executed (t_{avail}). So, each individual's execution time has to be less than it. Classes A, B, C and D

define the individuals whose normal execution times are not larger than $2.1 \times t_{avail}$. With a simple core-allocation strategy (A - One core, B - Two cores, C - Four cores, and D - Eight cores), these individuals can be executed in less than t_{avail}. Individuals labeled with E are expected not to be executed in time, and therefore three new strategies are developed to deal with these individuals. The first approach ("Time Aware Classification with replacement strategy") replaces E individuals with A individuals, which improves the execution time but restricts the search space and may diminish the calibration quality. In the second strategy ("Time Aware Classification without replacement strategy"), E individuals are executed using eight cores. At the end of the generation, the individuals still running will be interrupted, and only those that have been executed and evaluated are considered for the application of genetic operators. Finally, in the third approach ("Soft Time Aware Classification without replacement strategy"), E individuals are also executed using eight cores. However, if these individuals are still being executed at the end of the generation time, they will continue their execution and will not be considered in the current population. At the end of execution, these individuals will be reconsidered and used to generate new individuals. This approach allows for variable population size between generations. The three strategies were compared using data from a real fire in La Jonquera, Spain, and it was determined that the soft deadline strategy without replacement produces the best results.

More recently, the study in [38] introduced a new strategy to deal with individuals with long execution times. The methods already described are based on allocating more cores to individuals with longer estimated execution times during the calibration stage. Additionally, to improve this, a time deadline is imposed to guarantee that the simulation of individuals doesn't go beyond a preset value. Also, in order to deal with individuals that are still being executed when the available time runs out, some strategies were introduced in [42]. An alternative approach is introduced, based on the monitoring of the fire spread prediction error that, in this particular work, corresponds to the symmetric difference between the real fire and the simulated fire areas, shown in Eq. (2.23). During the execution of one individual, if the monitoring agent detects that the difference between the predicted and the simulated fires is larger than a predefined error threshold, the individual is interrupted. The fitness function is a weighted version of the symmetric difference, shown in Eq. (2.24),

$$Fitness = \frac{PredictionTime}{SimulationTime} \times SymDifference, \tag{2.24}$$

where *PredictionTime* represents the predicted time for the completion of the individual's simulation, *SimulationTime* is the the time of simulation until the individual is normal or early terminated, and *SymDifference* represents the symmetric difference from (2.23). This fitness function penalizes individuals that have been terminated early due to a large prediction error: they are not removed from the population, which ensures diversification, but are ranked worst due to lower fitness. This method was tested using fire data from a real fire in La Jonquera, Spain and it reduced the overall execution time in relation to the time aware core allocation technique from [39] by 60%.

2.4 Literature review summary

The review presented above showed that the majority of the works are based on the two-stage framework formally introduced in [28] combined with the use of genetic algorithms.

Genetic algorithms show very good suitability for being used as the optimization method in the referenced framework, not only based on their performance when compared to other optimization methods [18], but also because they have characteristics suited for being implemented in parallel. Implementing the two-stage framework with genetic algorithms and fire simulators in parallel is of great importance allowing the reduction of both calibration's and prediction's execution times [39].

Table 2.2 contains the above-cited works related to the literature review, organized by characteristics such as the focus of the paper, the source of the data used in experiments and tests and GA's parameters (number of individuals per generation, number of generations, operators probabilities and fitness functions).

Ref.	Focus	Source of datasets	Individuals	Gens.	Others
[28]	Input parameter calibration. Introduction of two-stage framework + input parameter sensitivity analysis	Simulation (*ISStest*)	1000	20	Fitness function is the XOR area (from *ISStest*) between real and simulated burned areas
[18]	Input parameter calibration using GAs, simulated annealing, random search and tabu search	Simulation (*ISStest*)	1000	-	Fitness function is the Hausdorff distance
[25]	Input parameter calibration	Simulation and 1 prescribed fire (Portugal)	50	5	—
[29]	Input parameter calibration. Two-stage framework with GA and guided search by past fires database	Real map $110 \times 110m^2$. *fireLib* simulation and 1 prescribed fire (Portugal)	Parallel: 512 Dynamic: 50	- 5	—
[24]	Input parameter calibration. Statistical integration to reduce search space	Real fire (California)	500	5	$elitism = 0.04$, $cross_{prob} = 0.2$, $mut_{prob} = 0.01$, Fitness function is symmetric difference (2.23)
[19]	Input parameter calibration. Two-stage framework with GA and comparison of the methods from [25] and [29]	1 simulated fire map using *fireLib* and 1 prescribed fire (Portugal)	Simulated: 50 Real: 500	5 5	Real fire case: $0.2 \leq mut_{prob} \leq 0.4$, Fitness function is cell-by-cell comparison of real and simulated fire maps
[27]	Input parameter calibration considering the rapid variation of wind speed and direction	Simulation (*FARSITE*)	50	10	Tests were performed 15 times
[20]	Rothermel fuel models calibration	1st test (GA-opt.): [31] [32]; 2nd test (Custom fuel model calibration): [43] [44]	100 for both	Max. 9999	$mut_{prob} = 0.1$, $elitism = 0.05$. Fitness function is RMSE of observed vs. experimental rate of spread R
[34]	Input parameter calibration. Two-stage framework with GA and WildFire Analyst	Real fire (Spain)	-	-	The fitness function is the symmetric difference (2.23)
[36]	Input parameter calibration, considering the spatial variation of wind in large fires	Real fire (Spain)	6	10	Tests were performed 15 times
[37]	Statistical study of genetic algorithms as the optimization algorithm in the two-stage framework	Simulation (*FARSITE*)	100	5	Tests were performed 50 times. $mut_{prob} = 0.1$, $elitism = 0.1$
[39]	Reduction of calibration time by parallel implementation	Simulation (*FARSITE*) based on a real terrain map (Spain)	25; 25; 100	10	Fitness function is the symmetric difference (2.23)
[41]	Reduction of calibration time by parallel implementation	Simulation (*FARSITE*) based on a real terrain map (Spain)	25	10	Tests were performed 50 times. Fitness function is the symmetric difference (2.23)
[42]	Reduction of calibration time by parallel implementation	Real fire (Spain)	-	10	$\#elitism = 10$, $cross_{prob} = 0.7$, $mut_{prob} = 0.3$. Tests were performed 10 times. Fitness function is the symmetric difference (2.23)
[38]	Reduction of calibration time by early terminating individuals based on prediction error in parallel implementation	Real fire (Spain)	100	10	$cross_{prob} = 0.7$, $mut_{prob} = 0.3$, Fitness function is a weighted version of the symmetric difference (2.24)

Table 2.2: Review of the literature on wildfire spread prediction calibration using genetic algorithms. **Gens.** column contains the number of GA's generations. **Others** column contains relevant information such as the GA's operators probabilities and fitness functions. **-** represents no relevant or existing data. ***elitism*** represents the percentage of the population's individuals selected for the GA's elitism operation. ***#elitism*** represents the number of individuals selected for the GA's elitism operation. $\boldsymbol{cross_{prob}}$ is the GA's crossover operation probability. $\boldsymbol{mut_{prob}}$ is the GA's mutation operation probability. **RMSE** represents the Root Mean Square Error.

Chapter 3

Overview of the implemented metaheuristic algorithms for wildfire spread model calibration

This chapter is dedicated to the description of the implemented metaheuristic algorithms for calibration of the Rothermel model. The algorithms were implemented in MATLAB® without resorting to specific toolboxes in order to guarantee the possibility of easily adapting each algorithm (they were developed from scratch). Section 3.1 presents the notation of the algorithms described in this chapter, Section 3.2 presents the genetic algorithm, Section 3.3 describes the differential evolution, and Section 3.4 is dedicated to simulated annealing. The algorithms here mentioned were also used in the work which resulted in the accepted conference paper "Wildfire Spread Prediction Model Calibration Using Metaheuristic Algorithms" [22] on the 48th Annual Conference of the IEEE Industrial Electronics Society (IECON 2022). Therefore, the following description coincides with some of the article's content.

3.1 Notation and definitions

It is important to clarify the reader regarding the notation which will be used in the description of the metaheuristic algorithms. Given that the genetic and the differential evolution algorithms are population-based, the notation used when referring to a population, to a candidate solution, and to its characteristics are shared and given as follows:

- The population is represented by P;
- The generation/iteration t of the population is represented by P_t
- A given candidate solution (individual) i of the population P is represented by i-th element of a population P, i.e. P_i;
- the j-th gene/element an individual i of the population P is referenced by $P_{i,j}$;
- N refers to the number of solutions (individuals) in the population P, and n is the number of elements (genes) of one candidate solution (individual).

Simulated annealing, which is not population-based, follows the following notation:

- A solution is represented by S;
- Indexes i and c refer to the initial (S_i) and current (S_c) solutions.

In this document, a given solution is composed of a set of Rothermel's input parameters $[x_1, x_2, \ldots, x_n]$, where for each input parameter x_j $(j = 1, \ldots, n)$ the interval of variation must be defined, i.e. for x_j $(j = 1, \ldots, n)$ the corresponding interval is $[x_{j_{min}}, x_{j_{max}}]$.

The algorithms which will be described in the following sections look for an optimal or near-optimal solution for a given problem. Frequently, this consists of finding the minima or maxima of a certain function in a determined domain. The description of the algorithms will be performed in the context of a minimization problem. The considered fitness/objective function $F(\cdot)$ is to be minimized, which means that solutions with smaller values of $F(\cdot)$ have more quality, are fitter.

3.2 Genetic algorithms

A genetic algorithm is a population-based stochastic search method introduced by [45] in 1975 inspired by natural selection and genetics. They are very useful in optimization problems by searching for the best solution in a specific space of possible problem solutions - search space [46, 47]. Every possible solution in the search space has an associated fitness value, which is obtained using a fitness function $F(\cdot)$. GA's look for the best solution (e.g., a minimum or a maximum of a given function), which is the fittest from the search space.

Genetic algorithms work by processing a set of elements of a given search space [46, 48]. This set is named population, and its elements are called individuals. Individuals, which represent the candidate solutions for the optimization problem, are also named chromosomes and are composed of genes. Genes are the primary parts of each solution. Individuals can have several representations, depending on the problem: they can be binary sequences of 0's and 1's, complex numbers, vectors, among others. The population is evolved/transformed during several generations in order to obtain a final population that contains individuals with the best possible quality for the problem at hand.

Algorithm 1 contains a generic scheme for a genetic algorithm, based on the proposed algorithms from [49, 46]. In the first generation ($t = 1$), an initial population P_1 of N individuals is randomly generated. Afterward, based on the evaluation by the fitness function $F(\cdot)$, the selection operator is applied to the current population to obtain a pair of parent chromosomes. Then, the crossover and mutation operators are applied to the parent pair to obtain a new pair of offspring. Crossover ensures the formation of new individuals from parent pairs by combining partial sequences of genes from each of the parents. Mutation acts on the individuals obtained by the crossover operator and alters some of their characteristics. This sub-process (selection, crossover and mutation) is repeated until achieving a new population of N individuals, P_{t+1}. The elitism operator is then applied, which consists of choosing at random a small fraction (*elitism*) of the new population to be replaced with the same number of the best individuals from the previous population [49]. The new population is evaluated, and the process is repeated up to the maximum number of generations (g_{max}). The resulting final solution is the fittest individual from the last population, i.e., the chromosome $P_{g_{max},i}$ with the optimal $F(\cdot)$ value.

In the work developed for this thesis, the specific GA operators used are the following:

Algorithm 1 Generic structure of a simple genetic algorithm.

Input:

1: Predefined individual structure and fitness function $F(\cdot)$.
2: Intervals of variation of the solutions' genes: $[x_{1_{min}}, x_{1_{max}}]$, $[x_{2_{min}}, x_{2_{max}}]$,..., $[x_{n_{min}}, x_{n_{max}}]$.
3: GA's parameters: N (number of individuals), g_{max} (maximum number of generations), n (number of genes), *elitism* (fraction of individuals to suffer elitism), selection, crossover and mutation operators.

Output: Optimal or near-optimal problem solution.

4: t $\leftarrow 1$
5: Randomly generate the initial population P_t.
6: **while** $t \leq g_{max}$ **do**
7: Evaluate all individuals $P_{t,i}$ $(i = 1, \ldots, N)$ from the population using a predefined fitness function $F(\cdot)$.
8: **repeat**
9: Select a pair of parents using Selection operator.
10: Generate a pair of offspring by applying Crossover operator.
11: Obtain the mutated offspring pair by applying Mutation operator.
12: **until** Obtain new population P_{t+1} of N individuals
13: Perform Elitism on P_{t+1}.
14: $t \leftarrow t + 1$.
15: **end while**

- Selection operator is the tournament selection [46], which consists of randomly selecting a certain number of individuals (tournament size $tour_{size}$) of the current population, creating a tournament. The winner of the tournament is the individual with the best fitness and it is selected to be a parent for the next generation. This process is repeated a second time, and a pair of parent individuals is obtained.

- The crossover operator is the single point crossover technique [46] and it is applied with a predefined probability of occurrence $cross_{prob}$. It is executed on the parent pair, by cutting the two chromosomes at corresponding points (the cutting point is randomly selected) and exchanging the sections after the cuts. This generates a new offspring pair.

- The mutation operator is the uniform operator [49]. This operator consists in altering the value of a random gene in the offspring by a uniform random value which fits the gene's respective search space, at a given probability of mutation mut_{prob}, a parameter defined at the beginning of the GA implementation.

3.3 Differential evolution

Differential evolution was first introduced in 1995 by Rainer Storn and Kenneth Price [50]. DE is similar to a GA in working by evolving a population of candidate solutions for a given problem. However, DE's search mechanism (differential mutation) is not based on any natural process.

DE initiates at iteration $t = 1$ by generating randomly an initial population P_1 with N individuals, each one containing n parameters. After this, the algorithm's main loop begins. First, a new mutant population is generated: the j-th element of the individual $P_{t,i}$ is obtained using the differential mutation operator [49]:

$$P'_{t,i,j} = P_{t,r_1,j} + f \times (P_{t,r_2,j} - P_{t,r_3,j}), \tag{3.1}$$

if $\gamma < C \vee j = \alpha_i$, otherwise $P'_{t,i,j} = P_{t,i,j}$, where $r_1, r_2, r_3 \in \{1, ..., N\}$ are three random integers, f is a user-defined scale factor which "controls the rate at which the population evolves" [51], $\gamma \in [0, 1]$ is a random uniform scalar, and $C \in [0, 1]$ is a user-defined number that controls the fraction of parameter values copied to the new mutant solution. The differential mutation operator is only applied to a given gene if $\gamma < C$, which means that the chance of applying the operator to more genes increases if C is closer to 1, with fewer parameter values copied to the new mutant solution. Additionally, $\alpha_i \in \{1, ..., n\}$ is a random uniform integer, which guarantees that at least one solution parameter is altered in the mutant solution. The algorithm iterates through every parameter of every individual until it obtains a new population of N individuals.

Afterward, the current and the new populations are compared: if the i-th individual from the new mutant population, $P'_{t,i}$ is less fit than the corresponding individual from the current population $P_{t,i}$, then the new individual is replaced by the current population's i individual. Finally, the main loop's stopping criterion is verified: if the fitness of the best individual $F(P_{t+1,b})$ of the new population does not improve in relation to the fitness of the best individual of the current population $F(P_{t,b})$, then the variable *count* is increased by one unity. The main loop stops if $count = count_{max}$ or when t reaches the maximum number of iterations. As in the GA, the final solution from the DE is the fittest individual from the last iteration's population. Algorithm 2 contains the generic structure of differential evolution, based on [49].

3.4 Simulated Annealing

Simulated annealing (SA) is a metaheuristic introduced in 1983 by Scott Kirkpatrick [52] and it is based on annealing, i.e., the process of heating a material and then slowly cooling it to obtain minimal energy states. As opposed to GA and DE, simulated annealing is not population-based.

The algorithm initiates by generating an initial solution, S_i. Then, S_i is evaluated using the defined fitness function $F(S_i)$ and set to the current solution, S_c. Furthermore, the temperature T is set to an initial value T_i, starting the main loop that lasts until the temperature reaches a final value T_f. For each value of T, the following process is repeated tr_{max} times: n_s neighboring solutions are generated from the current solution S_c by randomly selecting one of its elements and replacing its value by a new random value that fits the respective parameter range. Afterward, the n_s neighboring solutions are evaluated. The best of these new n_s solutions is selected and set to S_{new}. The process of generating neighboring solutions and selecting the fittest is based on greedy search [53]. If this new solution S_{new} is fitter than S_c or if a randomly chosen uniform number $\epsilon_{[0,1)}$ is smaller than the probability of acceptance $exp((F(S_c) - F(S_{new})/T)$, then S_c is replaced by S_{new}. In this work, the temperature T is updated by being multiplied by the cooling factor c_f: $T \leftarrow T \times c_f$. When the condition $T \leq T_f$ is verified, the algorithm is ceased and the current solution S_c is considered to be the best and final solution. Algorithm 3 shows the generic structure of the simulated annealing algorithm.

Algorithm 2 Generic structure of the differential evolution algorithm.

Input:

Predefined individual structure and fitness function $F(\cdot)$.
Intervals of variation of the solutions' parameters: $[x_{1_{min}}, x_{1_{max}}]$, $[x_{2_{min}}, x_{2_{max}}]$,..., $[x_{n_{min}}, x_{n_{max}}]$.
DE's parameters: N (number of individuals), n (number of solution parameters), C (fraction of parameters affected by the differential mutation), f (scale factor used in the differential mutation), t_{max} (maximum number of iterations), and $count_{max}$ (maximum number of iterations for non-improvement of the populations' best fitness).

Output: Optimal or near-optimal problem solution.

$t \leftarrow 1$, $count \leftarrow 0$
Randomly generate initial population P_t.
while $t < t_{max}$ **and** $count < count_{max}$ **do**
  for $i = 1, \ldots, N$ **do**
    Randomly generate $r_1, r_2, r_3 \in \{1, \ldots, N\}$.
    Randomly generate $\alpha_i \in \{1, \ldots, n\}$.
    for $j = 1, \ldots, n$ **do**
      Generate uniform random number $\gamma \in [0, 1]$.
      if $\gamma < C$ **or** $j = \alpha_i$ **then**
        Obtain new gene in the position j, $P'_{t,i,j}$, through differential mutation (3.1).
      else
        $P'_{t,i,j} = P_{t,i,j}$
      end if
    end for
  end for
  for $i = 1, \ldots, N$ **do**
    Using $F(\cdot)$, obtain the fitness values of $P_{t,i}$, $F(P_{t,i})$, and $P'_{t,i}$, $F(P'_{t,i})$.
    if $F(P'_{t,i}) \leq F(P_{t,i})$ **then**
      $P_{t+1,i} \leftarrow P'_{t,i}$
    else
      $P_{t+1,i} \leftarrow P_{t,i}$
    end if
  end for
  if $F(P_{t+1,b}) \geq F(P_{t,b})$ **then**
    $count \leftarrow count + 1$
  else
    $count = 0$
  end if
  $t \leftarrow t + 1$
end while

Algorithm 3 Generic structure of the simulated annealing algorithm.

Input:

Predefined individual structure and fitness function $F(\cdot)$.
Intervals of variation of the solutions' parameters: $[x_{1_{min}}, x_{1_{max}}]$, $[x_{2_{min}}, x_{2_{max}}]$,..., $[x_{n_{min}}, x_{n_{max}}]$.
SA's parameters: T_i (initial temperature), T_f (final temperature), c_f (cooling factor), tr_{max} (maximum number of tries for constant temperature), n_s (number of neighboring solutions).

Output: Optimal or near-optimal problem solution.

Randomly generate initial solution S_i.
Using $F(\cdot)$, evaluate initial solution S_i and set current solution to the initial solution: $S_c \leftarrow S_i$.
$T \leftarrow T_i$
while $T > T_f$ **do**
  for $t = 1, \ldots, tr_{max}$ **do**
    Generate n_s solutions by disturbing the current solution.
    Using $F(\cdot)$, evaluate the n_s neighboring solutions and select the best one, assigning it as S_{new}.
    if $[F(S_{new}) < F(S_c)]$ **or** $[\epsilon_{[0,1)} < exp(\frac{F(S_c)-F(S_{new})}{T})]$ **then**
      $S_c \leftarrow S_{new}$
    end if
  end for
  $T \leftarrow T \times c_f$
end while

Chapter 4

Calibration of the Rothermel model

This chapter explores/studies the feasibility of using three metaheuristic algorithms, genetic algorithm (GA), differential evolution (DE), and simulated annealing (SA), described in Chapter 3, for the calibration of the Rothermel model described in Section 2.1. The main contribution of this chapter is to validate two metaheuristic algorithms (DE and SA) for the calibration of the Rothermel model, in comparison with the already well-established GA, in the subject of wildfire spread prediction calibration. The Rothermel model calibration results are presented in Section 4.2 and show the potential for using differential evolution (DE) as a population-based alternative metaheuristic to genetic algorithms. This work resulted in the accepted conference paper "Wildfire Spread Prediction Model Calibration Using Metaheuristic Algorithms" [22] on the 48th Annual Conference of the IEEE Industrial Electronics Society (IECON 2022).

This chapter is structured the following way: Section 4.1 describes the methodology used, Section 4.2 presents the obtained results, and in Section 4.3 some conclusions are drawn.

4.1 Calibration methodology

In this section, the proposed methodology for the calibration of the Rothermel model is presented, where the input parameters to be calibrated are defined in Section 4.1.1 and the fitness function used to evaluate the solutions generated by the metaheuristic algorithms (GA, DE, and SA) is defined in Section 4.1.2. Furthermore, the overall calibration procedure is structured in Section 4.1.3.

4.1.1 Solution structure: input parameters to be calibrated

As explained in Chapter 3, the algorithms implemented in this thesis design candidate solutions (P_i for GA and DE or S_i for SA), which are represented by a vector with four different elements corresponding to the four input parameters to be calibrated: surface-area-to-volume ratio (σ), fuel bed depth (δ), fuel moisture (M_f), and midflame wind speed (U): $P_i \equiv S_i \equiv [\sigma_i, \delta_i, M_{f_i}, U_i]$.

The choice for calibrating these four parameters is justified as follows:

1. The first three parameters (σ, δ and M_f) are related to fuel characteristics, which in simulations are approximated using fuel models. A fuel model is a categorized set of values of fuel properties which are used as inputs for fire spread models, corresponding

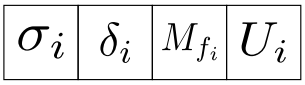


Figure 4.1: Representation of a candidate solution, corresponding to four input parameters.

to a well defined fuel type. These inputs can be used for predicting fire behavior in zones with similar fuel, thus eliminating the necessity for repeatedly measuring the properties [7]. Fuel models assume constant and uniform fuel characteristics inside a cell, which is a fair approximation for small cell sizes, a large variety of fuel models and accurate fitting of the model to the existing fuels. However, available fuel maps can suffer from low resolution (large cell sizes), low variety of models (the most commonly used standard NFFL fuel models [54] includes only 13 different fuel models) and low accuracy, therefore increasing the probability of fuel models failing to accurately depict the average characteristics of existing fuels.

2. Furthermore, the fire dynamics is known to induce local changes in the fuel characteristics, as well as wind speed and direction, in the close vicinity of the fire front [55, 56, 57] (fuel moisture drastically decreases while wind speed increases). To some extent, such changes are intrinsic to the semiempirical Rothermel model. However local variations in such parameters should be expected, which justifies their calibration.

3. These four input parameters are the ones that have the most influence on the final result (fire rate of spread), so their small variations are highly significant [58, 59].

4.1.2 Fitness function

The fitness of a given solution S_i generated by the three algorithms is evaluated by the relative error between a real observed value of rate of spread (R_{obs}) and the rate of spread from the Rothermel model when fed with the four input parameters values of the solution ($R(\sigma_i, \delta_i, M_{f_i}, U_i)$). The fitness is given by:

$$R^i_{Error} = \frac{|R(\sigma_i, \delta_i, M_{f_i}, U_i) - R_{obs}|}{R_{obs}}. \tag{4.1}$$

In this way, the goal of the algorithms is to find the best solutions with the lowest associated values of R_{Error}, i.e., the solutions whose values of $R(\sigma_i, \delta_i, M_{f_i}, U_i)$ approach the real R_{obs}.

4.1.3 Calibration algorithm

Algorithm 4 contains the calibration methodology carried out in this chapter. Each of three metaheuristic algorithms is executed for minimizing the fitness function R^i_{Error} (4.1), for each particular dataset. The inputs required for Algorithm 4 are the intervals of variation of the Rothermel model's input parameters to be calibrated, the dataset and each algorithm's specific parameters. Each algorithm provides the final calibrated set of input parameters, i.e., the solution with the best associated fitness (lowest value of R^i_{Error}).

Algorithm 4 Fire spread calibration methodology

Input:

1: Limits of the input parameters to be calibrated: σ_{min} and σ_{max}, δ_{min} and δ_{max}, $M_{f_{min}}$ and $M_{f_{max}}$, U_{min} and U_{max};

2: Experimental dataset, i.e. Rothermel input parameters values and R_{obs}.

3: GA's parameters: N (number of individuals), g_{max} (maximum number of generations), $elitism$ (fraction of individuals to suffer elitism), selection ($tour_{size}$), crossover ($cross_{prob}$) and mutation (mut_{prob}) operators.

4: DE's parameters: N (number of individuals), C (fraction of parameters affected by the differential mutation), f (scale factor used in the differential mutation), t_{max} (maximum number of iterations), and $count_{max}$ (maximum number of iterations for non-improvement of the populations' best fitness).

5: SA's parameters: T_i (initial temperature), T_f (final temperature), c_f (cooling factor), tr_{max} (maximum number of tries for constant temperature), and n_s (number of neighboring solutions).

Output: Calibrated Rothermel model.

6: Apply the metaheuristic algorithm (Algorithms 1 or 2 or 3) to minimize the fitness function R^i_{Error} (4.1).

4.2 Results

In this section, the results of the proposed methodology presented in Algorithm 4 for the calibration of the input parameters of the Rothermel model are presented and discussed. Section 4.2.1 describes the datasets used for calibration. Section 4.2.2 presents and discusses the results of the calibration.

4.2.1 Datasets

The datasets used for the calibration carried out in this were obtained through experimental prescribed fires in controlled conditions. 37 datasets were used, which were provided by ADAI. Each dataset contains information from a different controlled fire, which occurred in the center region of Portugal in the last five years, under various locations, different fuels, and weather conditions. Each dataset is a vector composed of values for Rothermel model's input parameters (2.1) according to the type of fuel burned (w_0, ρ_p, S_T, M_x, S_e, h, ϕ), measured values for w_0 ($w_{0_{obs}}$), δ (δ_{obs}), M_f ($M_{f_{obs}}$), U (U_{obs}) and the fire rate of spread R_{obs}.

As previously stated, the only Rothermel input parameters to be calibrated are σ (surface-area-to-volume ratio), δ (fuel bed depth), M_f (fuel moisture), and U (midflame wind speed). Despite being given (δ_{obs}, $M_{f_{obs}}$ and U_{obs}), δ, M_f and U are still calibrated since they may have an elevated associated error. For δ, the origin of the error is the fact that for the whole fire field, one average value for the fuel bed depth is assumed, based on a certain number of measurements. For M_f and U, the origin of the error is, as stated in Section 4.1.1, the fact that the fire itself induces variations on these parameters, so they do not remain constant throughout the fire. Therefore, initial values of M_f and U may not accurately represent the real fire conditions. For each calibrated parameter there is an interval of variation, from which the parameter can assume any value. For σ and δ, the intervals are defined based on the Northern Forest Fire Laboratory (NFFL) fuel

model corresponding to the fuel burned in the prescribed fires [54]. In the case of M_f and U, since there are approximate measured values for these parameters ($M_{f_{obs}}$ and U_{obs}), their intervals of variation are centered on these measured values. According to the ADAI experts, the intervals of variation of each parameter to be calibrated are:

- $\sigma \in [43, 80]$ [cm^{-1}];
- $\delta \in [0.25, 1.2]$ [m];
- $M_f \in [0.8 \times M_{f_{obs}}, 1.2 \times M_{f_{obs}}]$ [%];
- $U \in [0.75 \times U_{obs}, 1.25 \times U_{obs}]$ [m/s].

4.2.2 Results analysis

Since the proposed calibration methodology (Algorithm 4) is based on three metaheuristics algorithms (Algorithms 1, 2, and 3), which are stochastic optimization methods, the calibration methodology was executed 30 times for each dataset and for each metaheuristic algorithm. The parameters of each metaheuristic algorithm were fixed to the values shown in Table 4.1.

Table 4.1: Parameter settings for the calibration methodology, Algorithm 4, using GA (Algorithm 1), DE (Algorithm 2), and SA (Algorithm 3).

GA	**DE**	**SA**
$N = 300$	$N = 300$	$T_i = 1000$
$g_{max} = 150$	$C = 0.5$	$T_f = 0.001$
$tour_{size} = 3$	$f = 0.5$	$c_f = 0.99$
$cross_{prob} = 0.7$	$t_{max} = 500$	$tr_{max} = 2$
$mut_{prob} = 0.3$	$count_{max} = 20$	$n_s = 20$
$elitism = 0.05$		

To evaluate the algorithms performance on each dataset, R^{Final}_{Error} (4.2) is defined, which is the average of the best fitness values over 30 trials:

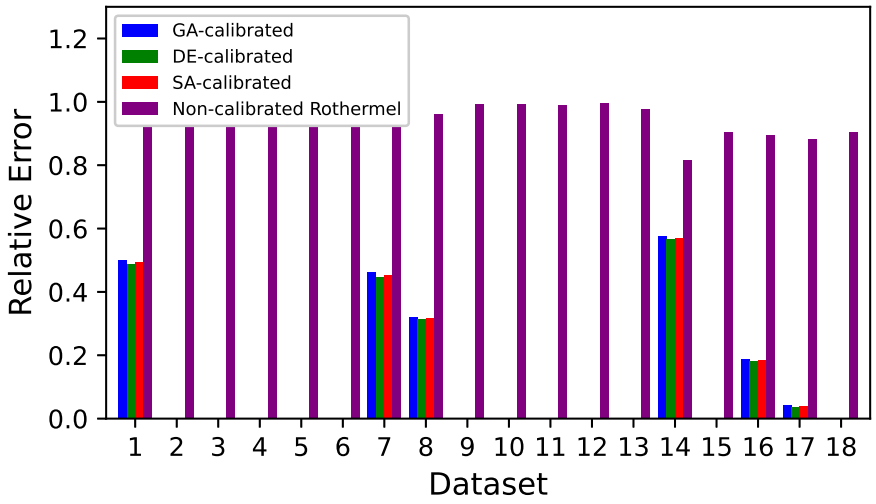
$$R^{Final}_{Error} = \frac{1}{30} \sum_{k=1}^{30} R^{k}_{Error}, \tag{4.2}$$

where R^{k}_{Error} is the fitness of the best solution given by (4.1) and provided by the k-th trial.

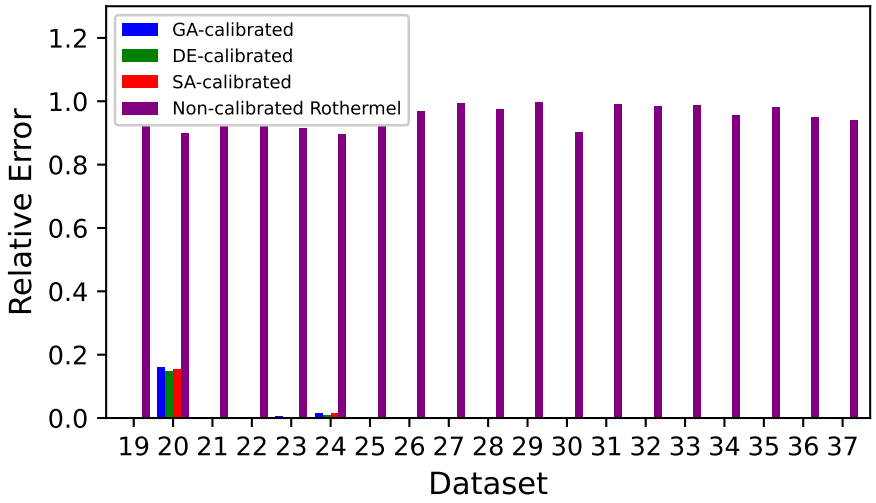
Figure 4.2 contains the average prediction error of the Rothermel model calibrated by each proposed algorithm for all datasets, R^{Final}_{Error} (4.2), and the relative error R^{NC}_{Error} (4.3) between the non-calibrated rate of spread R_{NC} and the observed rate of spread R_{obs}:

$$R^{NC}_{Error} = \frac{|R_{NC}(\sigma', \delta_{obs}, M_{f_{obs}}, U_{obs}) - R_{obs}|}{R_{obs}}, \tag{4.3}$$

where R_{NC} is calculated using the measured values provided in each dataset for δ, M_f and U, i.e., δ_{obs}, $M_{f_{obs}}$ and U_{obs}, respectively. For σ, given that its value isn't given in the



(a) Datasets 1 to 18.



(b) Datasets 19 to 37.

Figure 4.2: Calibration results from Algorithm 4: comparison of the GA-calibrated (Algorithm 1), DE-calibrated (Algorithm 2), and SA-calibrated (Algorithm 3) models against the non-calibrated Rothermel model, for every dataset.

dataset, the default value of $\sigma' = 57\,cm^{-1}$ was used. This σ value corresponds to NFFL fuel model no. 6 [54], which is the model that most accurately matches the fuel burned in the prescribed fires.

In Figure 4.2, in some datasets, only the non-calibrated model error bar is noticeable, since the proposed algorithms obtained, approximately, null relative error. Also, Figure 4.2 shows the significant difference between the prediction errors of the calibrated and non-calibrated models. Table 4.2 presents the average of the prediction errors R^{Final}_{Error} (4.2) of all datasets, R^{all}_{Error}, the best fitness result from all datasets, and the average of the non-calibrated model relative errors R^{NC}_{Error} (4.3) for all datasets, $R^{NC_{all}}_{Error}$. Table 4.2 shows that the three metaheuristics algorithms achieved similar calibration performances. Furthermore, GA and DE had the same best fitness results, despite DE performing slightly better than GA in R^{all}_{Error}. Additionally, for some datasets (1-st, 7-th, 8-th, 14-th, 16-th,

Table 4.2: Results of the proposed calibration algorithm (Algorithm 4) using: GA (Algorithm 1), DE (Algorithm 2), and SA (Algorithm 3).

Algorithms	R^{all}_{Error}	Best fitness	First occurrence		$R^{NC_{all}}_{Error}$
			Iteration	Time (s)	
GA	0.250	3.23×10^{-4}	37	3.166	0.951
DE	0.244	3.23×10^{-4}	3	0.037	
SA	0.248	6.33×10^{-4}	25	0.412	

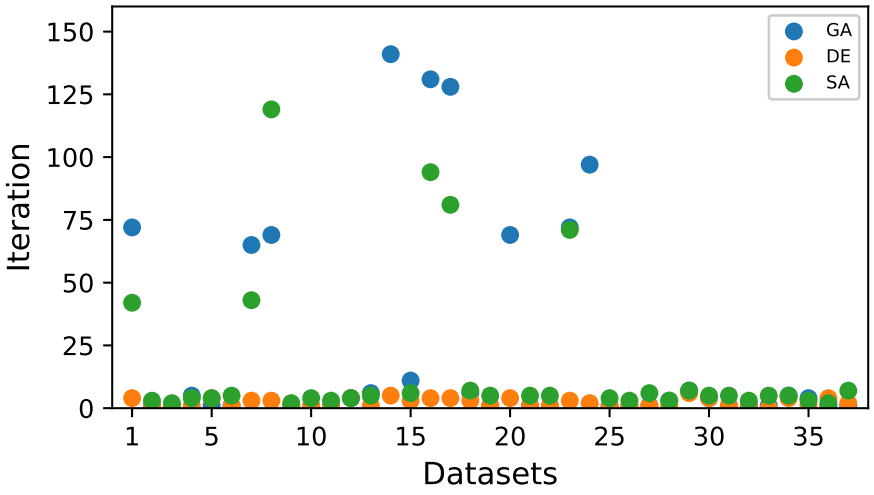


Figure 4.3: Iteration of first occurrence of the best fitness value, for each algorithm and dataset.

17-th and 20-th), the three algorithms could not obtain a near-zero relative error, despite obtaining similar results. This may be due to a bad suitability of the considered fuel model to the real fuel burned in those fire experiments or, simply, the model's intrinsic incapacity of accurately replicating the real fire behavior in those specific conditions. Finally, Tables 4.3 and 4.4 contain, for each dataset and algorithm, the mean μ and the standard deviation σ parameter values from the 30 calibrated solutions, which resulted from the 30 runs of each algorithm. From these tables, it can been seen that the parameters calibrated on some datasets are very similar in both algorithms and with a small variation between the 30 trials (ie. low σ), as for example datasets 1, 7 and 8. However, in same dataset, the calibrated parameters have higher variation, as for example datasets 3, 6, and 11.

As pointed out in Section 2.3.2, another important aspect in wildfire spread prediction is the calibration time. The calibration of the model should be performed on time to obtain usable fire spread predictions. To evaluate the time performance of the algorithms, we consider the time and number of iterations that led to the first occurrence of the best fitness value provided by the algorithms, as shown in Table 4.2. Figure 4.3 contains the iterations of the first occurrences of the best fitness values for each algorithm and each dataset. From Figure 4.3, it can be observed a clear pattern for the differential evolution, which takes a small number of iterations to obtain a first value of the best fitness. The number of iterations is more dispersed for the genetic algorithm and simulated annealing. Additionally, it is important to refer that for three datasets (14-th, 20-th, and 24-th), the

Table 4.3: Calibration results of σ and δ, for the three metaheuristics and for each dataset.

Dataset	$\sigma\,[cm^{-1}]$			$\delta\,[m]$		
	GA	**DE**	**SA**	**GA**	**DE**	**SA**
	$\mu \pm \sigma$	$\mu \pm \sigma$	$\mu \pm \sigma$	$\mu \pm \sigma$	$\mu \pm \sigma$	$\mu \pm \sigma$
1	43.104 ± 0.089	$43.000 \pm 4.5 \times 10^{-4}$	43.075 ± 0.071	0.251 ± 0.001	$0.250 \pm 9.5 \times 10^{-7}$	$0.251 \pm 4.9 \times 10^{-4}$
2	51.209 ± 5.671	52.176 ± 8.562	49.867 ± 6.581	0.288 ± 0.029	0.286 ± 0.036	0.308 ± 0.033
3	63.621 ± 9.819	57.392 ± 9.852	59.0715 ± 9.366	0.563 ± 0.219	0.638 ± 0.263	0.503 ± 0.214
4	48.848 ± 4.370	49.134 ± 5.649	47.246 ± 3.834	0.269 ± 0.016	0.264 ± 0.015	0.283 ± 0.019
5	48.803 ± 3.485	52.025 ± 5.648	46.718 ± 3.373	0.272 ± 0.016	0.263 ± 0.017	0.290 ± 0.017
6	61.404 ± 10.088	55.174 ± 11.406	60.083 ± 10.434	0.629 ± 0.151	0.718 ± 0.179	0.645 ± 0.165
7	43.127 ± 0.135	43.002 ± 0.004	43.065 ± 0.057	0.252 ± 0.002	$0.250 \pm 1.6 \times 10^{-5}$	$0.251 \pm 5.0 \times 10^{-4}$
8	43.105 ± 0.101	43.016 ± 0.022	43.072 ± 0.064	0.251 ± 0.002	$0.250 \pm 5.7 \times 10^{-5}$	$0.250 \pm 3.3 \times 10^{-4}$
9	65.123 ± 8.072	54.222 ± 11.779	59.722 ± 10.660	0.568 ± 0.122	0.768 ± 0.241	0.633 ± 0.187
10	69.118 ± 6.820	67.114 ± 8.795	65.080 ± 9.258	0.930 ± 0.147	1.025 ± 0.158	0.959 ± 0.178
11	66.868 ± 8.932	51.883 ± 9.799	58.418 ± 9.298	0.424 ± 0.106	0.559 ± 0.159	0.466 ± 0.090
12	74.379 ± 4.006	73.932 ± 5.520	73.620 ± 3.821	1.095 ± 0.082	1.097 ± 0.099	1.081 ± 0.077
13	51.644 ± 5.602	54.926 ± 7.729	48.919 ± 5.859	0.285 ± 0.026	0.177 ± 0.027	0.311 ± 0.030
14	43.106 ± 0.078	$43.000 \pm 2.8 \times 10^{-4}$	43.026 ± 0.021	$0.251 \pm 8.2 \times 10^{-4}$	$0.250 \pm 3.5 \times 10^{-7}$	$0.250 \pm 1.9 \times 10^{-4}$
15	46.398 ± 2.609	46.950 ± 3.334	45.370 ± 1.876	0.264 ± 0.010	0.259 ± 0.010	0.267 ± 0.012
16	43.079 ± 0.073	$43.000 \pm 5.7 \times 10^{-4}$	43.028 ± 0.019	$0.251 \pm 9.8 \times 10^{-4}$	$0.250 \pm 1.9 \times 10^{-7}$	$0.250 \pm 2.9 \times 10^{-4}$
17	43.078 ± 0.064	43.000 ± 0.000	43.058 ± 0.042	0.251 ± 0.001	0.250 ± 0.000	$0.250 \pm 2.4 \times 10^{-4}$
18	47.613 ± 3.384	49.073 ± 3.587	46.144 ± 2.877	0.275 ± 0.015	0.264 ± 0.016	0.279 ± 0.018
19	59.239 ± 10.228	54.359 ± 11.940	59.144 ± 10.709	0.413 ± 0.092	0.465 ± 0.138	0.424 ± 0.107
20	43.098 ± 0.083	43.002 ± 0.002	43.064 ± 0.071	0.252 ± 0.002	$0.250 \pm 1.4 \times 10^{-5}$	$0.251 \pm 5.4 \times 10^{-4}$
21	45.079 ± 9.146	58.756 ± 11.441	59.982 ± 10.370	0.485 ± 0.082	0.529 ± 0.110	0.533 ± 0.099
22	55.743 ± 8.926	56.328 ± 9.314	50.945 ± 5.146	0.293 ± 0.037	0.277 ± 0.034	0.312 ± 0.037
23	43.095 ± 0.114	43.048 ± 0.049	43.075 ± 0.052	0.252 ± 0.001	$0.250 \pm 1.4 \times 10^{-4}$	$0.251 \pm 4.8 \times 10^{-4}$
24	43.111 ± 0.089	$43.000 \pm 4.5 \times 10^{-4}$	43.076 ± 0.051	0.251 ± 0.001	$0.250 \pm 2.8 \times 10^{-6}$	$0.251 \pm 4.9 \times 10^{-4}$
25	61.712 ± 11.609	54.891 ± 10.632	61.807 ± 11.682	0.465 ± 0.103	0.522 ± 0.098	0.447 ± 0.129
26	60.775 ± 11.371	59.832 ± 12.971	60.401 ± 12.278	0.459 ± 0.122	0.496 ± 0.155	0.460 ± 0.126
27	66.964 ± 7.717	67.612 ± 7.845	67.126 ± 8.445	0.949 ± 0.143	0.975 ± 0.137	0.998 ± 0.139
28	61.336 ± 10.568	53.837 ± 9.981	60.372 ± 10.349	0.646 ± 0.271	0.479 ± 0.249	0.528 ± 0.269
29	74.344 ± 3.618	74.245 ± 4.515	74.542 ± 3.097	1.135 ± 0.054	1.121 ± 0.060	1.131 ± 0.042
30	46.085 ± 2.033	47.590 ± 3.596	45.131 ± 1.648	0.268 ± 0.010	0.260 ± 0.012	0.274 ± 0.012
31	66.413 ± 9.545	68.598 ± 7.856	68.492 ± 7.156	0.972 ± 0.142	0.957 ± 0.115	0.947 ± 0.156
32	60.418 ± 9.640	57.884 ± 11.419	57.640 ± 10.538	0.715 ± 0.154	0.805 ± 0.230	0.796 ± 0.194
33	62.620 ± 9.298	57.425 ± 10.861	61.399 ± 9.786	0.786 ± 0.148	0.852 ± 0.206	0.827 ± 0.182
34	51.891 ± 5.852	53.360 ± 7.592	52.222 ± 6.942	0.291 ± 0.029	0.281 ± 0.031	0.299 ± 0.043
35	64.754 ± 8.346	54.908 ± 9.464	63.152 ± 12.705	0.731 ± 0.170	0.903 ± 0.177	0.778 ± 0.234
36	57.360 ± 7.845	58.176 ± 10.574	54.724 ± 10.781	0.305 ± 0.044	0.325 ± 0.064	0.344 ± 0.055
37	45.980 ± 2.318	46.735 ± 2.735	44.972 ± 1.816	0.265 ± 0.011	0.256 ± 0.008	0.270 ± 0.011

Table 4.4: Comparison of the calibration results of M_f and U, for the three metaheuristics.

Dataset	M_f [%]			U $[m/s]$		
	GA	**DE**	**SA**	**GA**	**DE**	**SA**
	$\mu \pm \sigma$	$\mu \pm \sigma$	$\mu \pm \sigma$	$\mu \pm \sigma$	$\mu \pm \sigma$	$\mu \pm \sigma$
1	15.350 ± 0.009	$15.360 \pm 2.0 \times 10^{-4}$	15.350 ± 0.009	2.151 ± 0.032	$2.115 \pm 9.5 \times 10^{-4}$	2.148 ± 0.032
2	13.793 ± 0.893	13.784 ± 1.279	14.287 ± 0.893	2.853 ± 0.414	2.824 ± 0.563	2.865 ± 0.414
3	20.589 ± 2.277	20.416 ± 2.318	19.398 ± 2.277	3.810 ± 0.462	3.911 ± 0.632	3.928 ± 0.462
4	18.820 ± 0.474	18.670 ± 0.691	19.006 ± 0.474	2.670 ± 0.324	2.615 ± 0.487	2.636 ± 0.324
5	10.565 ± 0.407	10.662 ± 0.673	10.890 ± 0.407	5.999 ± 0.789	5.711 ± 1.077	6.008 ± 0.789
6	9.605 ± 1.208	9.736 ± 1.271	9.583 ± 1.208	5.464 ± 0.663	5.335 ± 0.913	5.112 ± 0.663
7	9.592 ± 0.007	9.599 ± 0.001	9.592 ± 0.007	1.594 ± 0.019	1.577 ± 0.004	1.597 ± 0.019
8	9.593 ± 0.005	9.598 ± 0.003	9.594 ± 0.005	1.288 ± 0.025	1.287 ± 0.020	1.306 ± 0.025
9	8.117 ± 0.976	8.371 ± 1.017	8.007 ± 0.976	2.124 ± 0.245	2.111 ± 0.372	2.119 ± 0.245
10	7.719 ± 0.900	8.039 ± 1.050	7.412 ± 0.900	2.443 ± 0.322	2.399 ± 0.425	2.449 ± 0.322
11	8.398 ± 0.918	8.345 ± 1.016	8.203 ± 0.918	2.738 ± 0.317	2.716 ± 0.391	2.687 ± 0.317
12	7.112 ± 0.375	7.072 ± 0.425	6.985 ± 0.375	3.421 ± 0.429	3.408 ± 0.645	3.404 ± 0.429
13	8.710 ± 0.549	8.837 ± 0.614	8.994 ± 0.549	3.127 ± 0.026	3.017 ± 0.595	2.944 ± 0.383
14	8.995 ± 0.003	$9.000 \pm 7.3 \times 10^{-5}$	8.997 ± 0.003	1.371 ± 0.062	1.352 ± 0.003	1.412 ± 0.062
15	6.938 ± 0.235	6.857 ± 0.259	6.902 ± 0.235	3.296 ± 0.419	3.149 ± 0.580	3.341 ± 0.419
16	6.237 ± 0.002	$6.240 \pm 4.7 \times 10^{-5}$	6.238 ± 0.002	1.592 ± 0.111	1.578 ± 0.004	1.679 ± 0.111
17	5.996 ± 0.005	6.000 ± 0.000	5.995 ± 0.005	2.486 ± 0.094	$2.475 \pm 4.5 \times 10^{-16}$	2.562 ± 0.094
18	8.547 ± 0.391	8.397 ± 0.330	8.466 ± 0.391	3.018 ± 0.315	2.819 ± 0.488	2.892 ± 0.315
19	10.832 ± 1.287	10.854 ± 1.616	10.981 ± 1.289	2.051 ± 0.253	1.837 ± 0.301	2.064 ± 0.253
20	10.791 ± 0.009	10.799 ± 0.001	10.790 ± 0.009	6.094 ± 0.075	6.081 ± 0.009	6.161 ± 0.075
21	9.401 ± 1.110	9.338 ± 1.165	9.527 ± 1.110	9.370 ± 1.452	9.229 ± 1.652	9.078 ± 1.452
22	9.663 ± 0.894	9.332 ± 1.182	9.677 ± 0.894	4.281 ± 0.525	4.264 ± 0.696	4.466 ± 0.525
23	11.389 ± 0.008	11.393 ± 0.007	11.388 ± 0.008	4.445 ± 0.051	4.488 ± 0.068	4.486 ± 0.051
24	9.593 ± 0.010	$9.600 \pm 7.2 \times 10^{-5}$	9.591 ± 0.010	5.944 ± 0.088	5.926 ± 0.003	6.016 ± 0.088
25	10.586 ± 1.401	10.656 ± 1.225	10.081 ± 1.401	2.169 ± 0.242	2.012 ± 0.352	2.115 ± 0.243
26	15.523 ± 1.896	15.941 ± 2.113	15.428 ± 1.896	2.891 ± 0.420	2.801 ± 0.524	2.725 ± 0.420
27	19.274 ± 1.114	19.520 ± 0.810	19.543 ± 1.114	6.700 ± 0.828	6.447 ± 1.094	6.556 ± 0.828
28	22.538 ± 1.507	20.930 ± 2.004	21.821 ± 1.507	6.371 ± 0.947	6.539 ± 1.106	6.612 ± 0.947
29	18.655 ± 0.397	18.540 ± 0.441	18.651 ± 0.397	6.388 ± 0.748	7.084 ± 0.970	6.446 ± 0.748
30	15.201 ± 0.340	15.099 ± 0.559	15.319 ± 0.340	2.527 ± 0.331	2.551 ± 0.448	2.654 ± 0.331
31	12.423 ± 1.252	12.801 ± 1.622	12.559 ± 1.252	2.567 ± 0.308	2.686 ± 0.442	2.523 ± 0.308
32	8.559 ± 0.890	8.847 ± 0.957	8.821 ± 0.890	2.826 ± 0.381	2.933 ± 0.491	2.851 ± 0.381
33	8.139 ± 0.833	7.867 ± 0.822	8.243 ± 0.833	3.343 ± 0.420	3.231 ± 0.612	3.172 ± 0.420
34	7.476 ± 0.667	7.392 ± 0.658	7.652 ± 0.667	1.475 ± 0.156	1.521 ± 0.231	1.440 ± 0.156
35	16.610 ± 1.876	16.963 ± 1.712	16.574 ± 1.876	5.591 ± 0.717	5.585 ± 1.036	5.771 ± 0.717
36	6.191 ± 0.573	6.495 ± 0.587	6.571 ± 0.573	4.592 ± 0.572	4.706 ± 0.784	4.630 ± 0.572
37	12.821 ± 0.339	12.580 ± 0.468	12.856 ± 0.339	1.964 ± 0.288	1.990 ± 0.316	1.987 ± 0.288

simulated annealing algorithm ran for more than 150 iterations until the first occurrence of the best fitness value (314, 961 and 432 iterations, respectively). Consequently, these points are not shown in Figure 4.3 to ensure a more consistent and accurate viewing. From Table 4.2, we verify that the differential evolution is the fastest algorithm, with an average duration of 3 iterations and $0.03707\,s$ until the first occurrence of the best final fitness value, in comparison with 37 iterations ($3.166\,s$) from the GA and 25 iterations ($0.4119\,s$) from the SA.

4.3 Conclusions

As stated in the beginning of this chapter, the wildfire spread prediction area has been dominated by the use of genetic algorithms as the main tool for the calibration of the Rothermel model. However, the results obtained in this chapter show that differential evolution is also a very suitable algorithm for the calibration of the Rothermel model, mainly due to its time performance, which is critical in wildfire spread prediction.

Regarding the results: the near-zero relative error obtained in the majority of the calibrations reveals the quality of the metaheuristic algorithms in finding fit solutions which allow to obtain accurate rate of spread R values. The results also show that for relatively stable fire conditions without any extreme behavior, and for well defined or calibrated input parameters, the Rothermel model can produce very exact predictions. However, at the same time, the Rothermel model considered in this work was directed at finding only the rate of spread R. It did not offer any information on the lateral and backward fire rate of spread, nor any information on the fire shape, which, in a real wildfire application are entirely necessary. In a tool which considers a more broad and complete fire behavior, the complexity increases and so, the prediction errors increase as well.

Chapter 5

Calibration of fire spread prediction model

In this chapter, the application of the well established two-stage methodology (Figure 2.2) presented in the literature review is performed. For this, the three implemented algorithms - GA, DE and SA - were adapted for calibrating the input parameters of a Rothermel-based fire spread platform (FIRESTATION) using data from a prescribed fire which took place in Castanheira de Pêra, center of Portugal. FIRESTATION is a platform that can predict real-time wildfire propagation to decision support.

This chapter is organized as follows: Section 5.1 presents the wildfire spread simulator, Section 5.2 describes the methodology used for the calibration, Section 5.3 presents the obtained results, and in Section 5.4 some conclusions are drawn.

5.1 Fire spread simulator

The fire spread simulator is the FIRESTATION [10] which was provided by ADAI. As stated in Section 2.3, this simulator is based on the Rothermel model. Therefore, the simulator's input parameters are essentially the same as the ones referred in Section 2.1, with some adaptations.

The Rothermel model calculates the fire rate of spread R along the main direction of propagation. However, in a real wildfire situation, it is important to obtain information about the shape and size of the wildfire. For that reason, in FIRESTATION, the Rothermel model is coupled with two models [60, 61] which provide the mathematical description of the fire shape.

Moreover, the growth of the fire area is simulated based on a raster approximation. The terrain is divided into squared cells over which the fuel characteristics are considered to be constant. Then, the fire spread is represented by the contagion between burning cells and non-burning cells (for more details on how the contagion process is performed, see [10]). Using a raster approach as the basis of the FIRESTATION simulator means that its input parameters can't simply be represented by a single value, as in the Rothermel model, in Chapter 4. Instead, the simulator takes as inputs files which contain the characteristics of the fire environment. Specifically, the input files are the following:

- **Terrain file** - ASCII data file consisting of the Digital Elevation Map (DEM) of the fire location. It follows the Esri ASCII raster format;

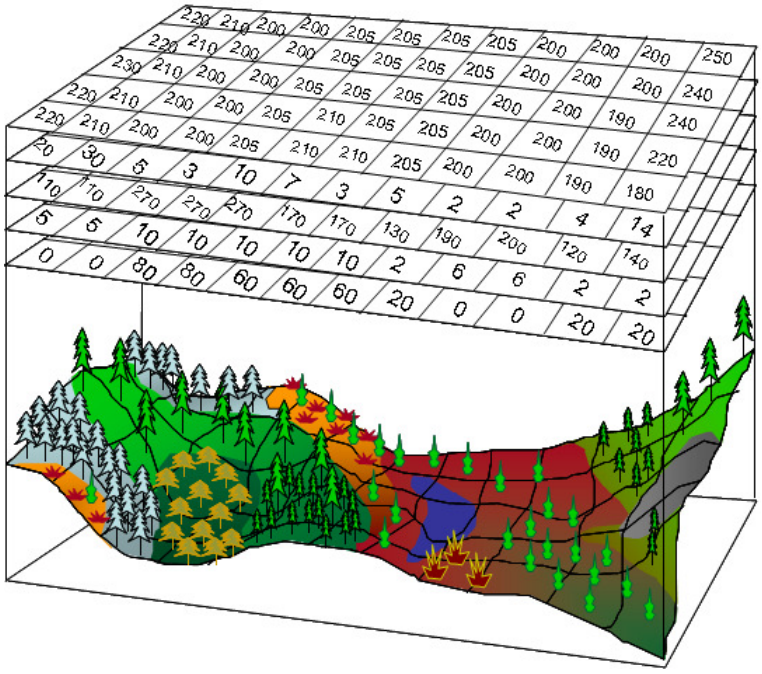


Figure 5.1: Illustration of how various layers which describe the location where the fire occurs and serve as input for the fire spread simulator.

Source: ADAI.

- **Fuel distribution file** - ASCII data file containing the fuel model code number for each cell. It follows the Esri ASCII raster format;
- **Nuatmos file** - file containing various layers with the wind field, generated using the Nuatmos model [62];
- **Ignition file** - contains information for the fire simulation initiation: time instant of initiation and the coordinates of the ignition cells;
- **Control file** - a text file that specifies the stopping criteria for the fire propagation simulations;
- **Fuel models file** - contains the values of fuel parameters for each fuel model.

Figure 5.1 illustrates how the overlapping of the various raster layers occurs in FIRESTATION.

The output of the fire simulator is a file containing a list of all of the simulated burned cells at the end of the simulation, along with other relevant information such as the time instant at which each cell burned and the respective rate of spread R value.

5.2 Methodology

In this section, the methodology used for testing the two-stage framework is presented. The input parameters to be calibrated and the calibration process are defined in Section 5.2.1, the fitness function used to evaluate the solutions in this chapter is defined in Section 5.2.2 and, finally, the used dataset and the overall calibration and prediction algorithm are presented in Section 5.2.3.

5.2.1 Solution structure

For this part of the work, a candidate solution generated by one of the three calibration algorithms is represented by a vector with two parameters: surface-area-to-volume ratio (σ) and fuel bed depth (δ): $P_i \equiv S_i \equiv [\sigma_i, \delta_i]$.

As opposed to what was done in Chapter 4, fuel moisture (M_f) and midflame wind speed (U) are not considered for calibration in this chapter. This is due to two reasons: first, in Chapter 4, the fuel moisture (M_f) was calibrated due to the existence of a measured fuel moisture value $M_{f_{obs}}$ around which an interval of variation was formed, for each dataset; secondly, using FIRESTATION, the wind input is not represented by a single parameter U and its respective value. Instead, as was described in Section 5.1, the wind input corresponds to a three-dimensional wind field which is generated using the Nuatmos model and wind measurements.

Moreover, since the fire spread simulator used is based on the Rothermel model, the justifications presented in Section 4.1.1 for choosing the parameters surface-area-to-volume ratio (σ) and fuel bed depth (δ) to be calibrated remain valid.

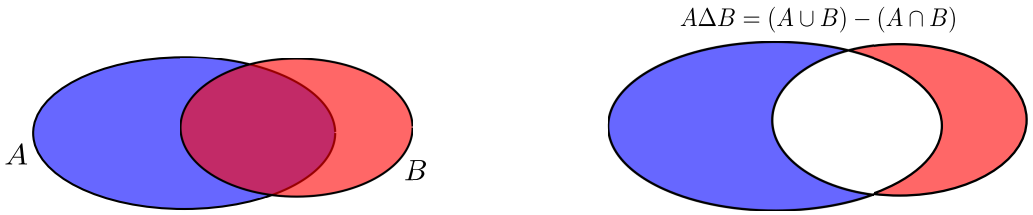
5.2.2 Fitness function

Similarly to the majority of the works in the literature review, the fitness of a given solution S_i, generated by the three calibration algorithms is based on the symmetric difference between the corresponding simulated fire area and the real fire area.

For two sets A and B, the symmetric difference $A\Delta B$ (5.1) corresponds to the union of the two sets minus their intersection:

$$A\Delta B = (A \cup B) - (A \cap B) \tag{5.1}$$

Figures 5.2a and 5.2b illustrate the concept of symmetric difference between two sets A and B:



(a) Two sets A and B. **(b)** Symmetric difference of A and B.

Figure 5.2: Illustrations of the symmetric difference between two sets A and B.

Since both the fire spread simulations' results and the real prescribed fire data consist of sets of the respective burned cells. Therefore, in order to calculate the symmetric difference between the two sets and evaluate the quality of the a solution S_i generated by the algorithms, the fitness function (5.2) is given by:

$$RSD_i = \frac{\cup_{cells_i} - \cap_{cells_i}}{R_{cells} - I_{cells}} \tag{5.2}$$

Equation (5.2) ends up corresponding to Equation (2.23) which was presented in the literature review. $\cup_{cells_i}$ corresponds to the number of cells in the union of the two sets (simulated fire cells corresponding to the solution S_i and real fire burned cells) ans $\cap_{cells_i}$

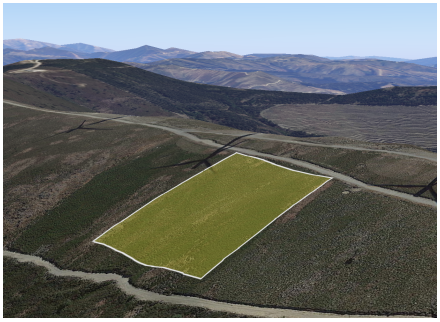
corresponds to the number of cells in the intersection of the two sets. In the denominator of Eq. (5.2), R_{cells} is the number of burned cells in the real fire and I_{cells} is the number of ignition cells of the real fire. The purpose of having $R_{cells} - I_{cells}$ in the denominator is so that fitness is calculated in relation to the number of the real fire burned cells, similarly to relative error (4.1) used as fitness function in the previous chapter, described in Section 4.1.2 (this is also the reason of having RSD_i as the function's variable, it corresponds to Relative Symmetric Difference, thus considering the presence of the denominator).

Similarly to what was described in Section 4.1.2, the goal of the algorithms in this chapter is also to find the best solutions with the lowest associated values of RSD_i, i.e., solutions whose associated simulated fire map shape is as close as possible to the real fire shape, so that the symmetric difference between the two is smallest possible.

5.2.3 Calibration and prediction methodology

As described in this chapter's introduction, the goal here is to put into application the two-stage fire spread prediction methodology (Figure 2.2) presented in Section 2.3.1, which uses algorithms for calibrating the wildfire spread model parameters, and compare its prediction against the real wildfire.

In order to do this, we use the data from a prescribed fire that took place in Castanheira de Pêra, in the center of Portugal, in 2022. The data, which was kindly provided by ADAI, was obtained through an experimental prescribed fire which took place in Castanheira de Pêra, in the center of Portugal, in 2022. Figure 5.3a shows a perspective on the fire field location and Figure 5.3b was obtained during the fire and shows the fire progression.



(a) Perspective of the fire field location on Google Earth.

(b) On-going prescribed fire.

Figure 5.3: Images regarding the prescribed fire used as test case for this work.

Source: ADAI.

The data consists of a set of four fire spread maps, each one corresponding to a different time instant: 4 min, 8 min and 12 min after the beginning of the fire. Moreover, it contains the fuel distribution file and with the digital elevation map (DEM) of the fire ground, the wind fields corresponding to the wind conditions registered at the same four time instants from the fire spread maps. The wind fields required for input of the fire spread simulator were obtained using the Nuatmos model and based of the wind measurements which were taken during the fire. As stated in Section 5.2.1, the input parameters to be calibrated are σ (surface-area-to-volume ratio) and δ (fuel bed depth). The rest of the fuel input parameters are set to the default values according to the respective fuel model. According to ADAI, the field where the prescribed fire occurred consists essentially of shrubs which

are best described by the NFFL fuel model no. 5 [54]. According to the ADAI experts, for this model, the intervals of variation of σ and δ are:

- $\sigma \in [56.100, 79.500]$ [cm^{-1}];
- $\delta \in [0.305, 0.915]$ [m].

As explained in Section 2.3.1, the two-stage methodology is divided into two parts: the calibration stage and the prediction stage. In the calibration stage the fire spread simulator's input parameters are calibrated using observed fire data from instant t_1. After this, and based on the assumption that the values of the input parameters values remain constant between t_1 and t_2, the calibrated parameters are used for obtaining fire spread predictions for the time step t_2. Considering the available real fire data, there are two possible scenarios for testing the two-stage methodology:

1. Scenario 1: use the data from instant $t_1 = 4\,min$ for **calibration** and then obtain fire spread **predictions** for $t_2 = 8\,min$;
2. Scenario 2: use the data from instant $t_1 = 4\,min$ for **calibration** and then obtain fire spread **predictions** for $t_2 = 12\,min$;
3. Scenario 3: use the data from instant $t_1 = 8\,min$ for **calibration** and obtain a fire spread **prediction** for $t_2 = 12\,min$.

The fire spread prediction simulations for $t_2 = 8\,min$ and $t_2 = 12\,min$ are obtained, not only using the calibrated values for σ and δ but also using the real fire data available from the respective calibration instant. For example, in the prediction for $t_2 = 8\,min$ using calibrated parameters from $t_1 = 4\,min$, the simulation initiates using the wind field corresponding to the instant $t = 4\,min$ and the ignition area is set to the real fire burned area from that time instant as well. This is performed in a similar way for the remaining predictions and the justification is the fact that, in a real wildfire situation, having obtained the real wildfire data from an intermediate time instant for calibration, that data is available to be used as input for next instant's respective prediction. This not only uses more accurate and updated input data but also significantly reduces the required prediction time because the simulation does not have to initiate from the first fire instant ($t = 0$). obtained employing the two-stage methodology are compared with the respective predictions obtained from the simulator without using any calibrated input parameters' values.

Algorithm 5 summarizes the two-stage methodology for wildfire spread prediction, for the specific real dataset described above.

5.3 Results

In this section, the results of the proposed methodology presented in Section 5.2.3 for the calibration of the fire simulator's input parameters and prediction of the fire spread are presented and discussed.

The stochastic nature the three metaheuristic algorithms was considered and each one was executed 5 times. Due to the computational time they were not executed 30 times, as performed in Chapter 4. The parameters of each metaheuristic algorithm were fixed to the values shown in Table 5.1.

Algorithm 5 Fire spread calibration methodology.

Input:

1: Limits of the input parameters to be calibrated: σ_{min} and σ_{max}, δ_{min} and δ_{max}; and the desired scenario.

2: Experimental dataset, i.e., ignition cells for $t = 0\,min$, burned cells maps for $t = 4\,min$, $t = 8\,min$ and $t = 12\,min$.

3: GA's parameters: N (number of individuals), g_{max} (maximum number of generations), *elitism* (fraction of individuals to suffer elitism), selection ($tour_{size}$), crossover ($cross_{prob}$) and mutation (mut_{prob}) operators and parameters.

4: DE's parameters: N (number of individuals), C (fraction of parameters affected by the differential mutation), f (scale factor used in the differential mutation), t_{max} (maximum number of iterations), and $count_{max}$ (maximum number of iterations for non-improvement of the populations' best fitness).

5: SA's parameters: T_i (initial temperature), T_f (final temperature), c_f (cooling factor), tr_{max} (maximum number of tries for constant temperature), and n_s (number of neighboring solutions).

Output: Calibrated input parameters.

6: Apply the metaheuristic algorithm (Algorithms 1 or 2 or 3) to minimize the fitness function RSD_i (5.2) using the observed fire propagation from $t_0 = 0\,min$ to t_1 in two situations: $t_1 = 4\,min$ or $t_1 = 8\,min$.

7: Perform predictions using the respective calibrated input parameters for $t_2 = 8\,min$ (Scenario 1) and $t_2 = 12\,min$ (Scenarios 2 and 3).

Table 5.1: Parameter settings for the calibration methodology, Algorithm 5, using GA (Algorithm 1), DE (Algorithm 2), and SA (Algorithm 3).

GA	**DE**	**SA**
$N = 25$	$N = 25$	$T_i = 1000$
$g_{max} = 150$	$C = 0.5$	$T_f = 0.001$
$tour_{size} = 3$	$f = 0.5$	$c_f = 0.99$
$cross_{prob} = 0.7$	$t_{max} = 500$	$tr_{max} = 2$
$mut_{prob} = 0.3$	$count_{max} = 20$	$n_s = 20$
$elitism = 0.05$		

The calibration and prediction results from the methodology presented in Section 5.2.3 are presented in Tables 5.2 and 5.3. Table 5.2 contains the results based on the calibration performed using the real fire area from $t_1 = 4\,min$ (Scenarios 1 and 2) and Table 5.3 shows the results when using the real fire area from $t_1 = 8\,min$ for calibration (Scenario 3). In both tables, $RSD*$ (5.3) is the best final fitness achieved by each algorithm from the 5 trials and $\overline{RSD}$ (5.4) is the mean of the final fitness values from the 5 trials, for each algorithm.

$$RSD* = \min\{RSD^1, RSD^2, \ldots, RSD^5\} \tag{5.3}$$

$$\overline{RSD} = \frac{1}{5}\sum_{k=1}^{5} RSD^k, \tag{5.4}$$

where RSD^k is the fitness of the best solution given by (5.2), resulting from the algorithm's k-th trial. Moreover, $\sigma*$ and $\delta*$ correspond to the best solution obtained by each algorithm, which corresponds to the best fitness $RSD*$. $\overline{\sigma}$ and $\overline{\delta}$ are the mean values of σ and δ from the final best individuals from each of the 5 trials, for each algorithm, and σ_σ and σ_δ are the standard deviations of the 5 calibrated solutions which resulted from the 5 runs of each algorithm. Finally, RSD_C is the prediction error of the best solution $[\sigma*, \delta*]$, i.e., the symmetric difference as presented in (5.2), between the simulated fire area resulting from $[\sigma*, \delta*]$ and the respective real fire area.

Table 5.2: Calibration and prediction results from the three algorithms when performing calibration using fire data of $t_1 = 4\,min$ and prediction for $t_2 = 8\,min$ and $t_2 = 12\,min$. Values for $\sigma*$ and $\overline{\sigma}$ are in cm^{-1} and for $\delta*$ and $\overline{\delta}$ are in m.

Algorithm	**Calibration results** $t_1 = 4\,min.$								**Prediction results**	
	Fitness		**Best solution**		**Mean solution**		**Std. deviations**		$t_2 = 8$	$t_2 = 12$
	$RSD*$	$\overline{RSD}$	$\sigma*$	$\delta*$	$\overline{\sigma}$	$\overline{\delta}$	σ_σ	σ_δ	RSD_C	
GA	0.520	0.528	57.617	0.384	57.092	0.399	0.819	0.009	0.330	0.485
DE	0.520	0.522	57.404	0.385	57.038	0.392	0.740	0.012	0.334	0.499
SA	0.531	0.535	56.555	0.405	57.876	0.408	2.657	0.029	0.360	0.534

Table 5.3: Calibration and prediction results from the three algorithms when performing calibration using fire data of $t_1 = 8\,min$ and prediction for $t_2 = 12\,min$. Values for $\sigma*$ and $\overline{\sigma}$ are in cm^{-1} and for $\delta*$ and $\overline{\delta}$ are in m.

Algorithm	**Calibration results** $t_1 = 8$								**Prediction results**
	Fitness		**Best solution**		**Mean solution**		**Std. deviations**		$t_2 = 12$
	$RSD*$	$\overline{RSD}$	$\sigma*$	$\delta*$	$\overline{\sigma}$	$\overline{\delta}$	σ_σ	σ_δ	RSD_C
GA	0.440	0.442	56.159	0.339	56.268	0.362	0.154	0.033	0.982
DE	0.440	0.441	56.150	0.340	56.183	0.350	0.133	0.020	0.985
SA	0.440	0.447	56.282	0.387	56.584	0.373	0.539	0.018	1.193

Table 5.4 contains the results of the fire spread prediction using the non-calibrated, default values for σ and δ inputs, σ' and δ'. Their values, according to the NFFL fuel model no. 5 [54], are $\sigma' = 66.000\,cm^{-1}$ and $\delta' = 0.610\,m$. RSD_{NC} corresponds to the symmetric difference between the real fire area and the non-calibrated simulated fire area (it can be interpreted as the fitness of the non-calibrated solution $[\sigma', \delta'] = [66.000, 0.610]$).

In order to complement the results analysis, Figures 5.4, 5.5 and 5.6 compare the non-calibrated simulated fire areas with the algorithms' best solutions' ($[\sigma*, \delta*]$) resulting simulated fire areas. Figures 5.4 and 5.5 concern the fire spread predictions for $t_2 = 8\,min.$ and $t_2 = 12\,min.$ using the real fire area of $t_1 = 4\,min.$ for calibration, and Figure 5.6

Table 5.4: Fire spread prediction results using the default (non-calibrated) values of σ and δ (σ' and δ').

Simulation time	$\mathbf{RSD_{NC}}$
$8\,min$	1.639
$12\,min$	2.931

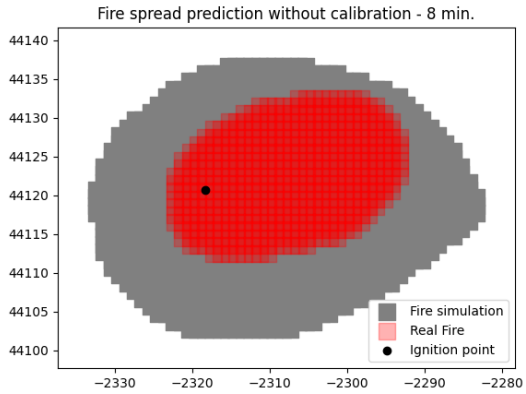
concerns the fire spread predictions for $t_2 = 12\,min.$ using the real fire area of $t_1 = 8\,min.$ for calibration.

First of all, comparing the results from Tables 5.2 and 5.3 with the results in Table 5.4 it is possible to verify that the implementation of a calibration stage for tuning the simulator's input parameters dramatically improved the fire spread predictions for the time instants that followed. This result was expected, and it can also be verified in Figures 5.4, 5.5 and 5.6, where the simulated fire areas resulting from calibrated values for inputs σ and δ are much similar to the respective real fire areas than the simulations resulting from the non-calibrated input values for σ and δ (σ' and δ', respectively).

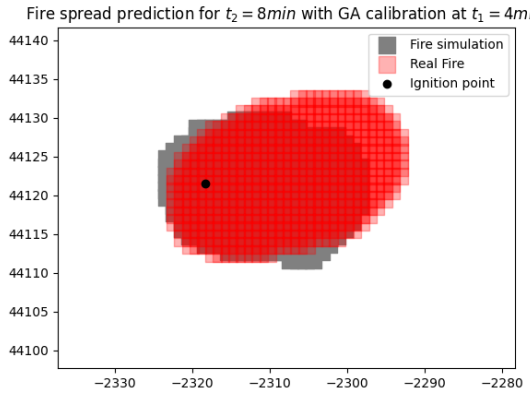
Considering the fire spread predictions for $t = 8\,min$ (Scenario 1): while using the non-calibrated input parameters resulted in an elevated RSD_{NC} value of 1.639 (Figure 5.4a shows an over-prediction of the burned area), the use of calibrated input parameters resulted in much lower values of RSD_C. In fact, the minimum reduction of the prediction error occurred for the simulated annealing (SA) algorithm, which obtained a RSD_C value of 0.360, which consists of a reduction in the prediction error of 78% (GA and DE were slightly better, obtaining reductions of 79.9% and 79.6%). Regarding the fire spread predictions for $t = 12\,min$, two different tests were performed, as already explained: calibration using the real fire data from $t = 4\,min$ (Scenario 2) and using the data from $t = 8\,min$ (Scenario 3). The fire spread prediction without calibration also resulted in an elevated prediction error, ie. $RSD_{NC} = 2.391$. On Scenario 2, the largest value of RSD_C was obtained again from the simulated annealing calibration (0.534, with a reduction in the prediction error of 81.79%) against 0.485 from the genetic algorithm (reduction of 83.45%) and 0.499 from the differential evolution (reduction of 82.99%). On Scenario 3, the prediction errors RSD_C were not as better as in the previous situation, but still, some considerable reductions from the non-calibrated prediction errors were obtained: 0.982 (reduction of 66.48%) for the GA, 0.985 (reduction of 66.39%) for the DE and 1.193 (reduction of 59.31%) for the SA.

Comparing the three metaheuristics calibration results, it is possible to verify that the differential evolution (DE) algorithm had a slightly better performance than the other two algorithms. $RSD*$ value for DE is the same as for the GA for the calibration performed with data from $t = 4\,min$ and is the same as the other two algorithms for the calibration performed with data from $t = 8\,min$. However, DE obtained better $\overline{RSD}$ values than the other algorithms for both calibrations, which means that over the 5 trials performed for each calibration, the final solutions obtained by the differential evolution (DE) algorithm were consistently very fit. The consistency and reproducibility of the differential evolution algorithm is also shown by the resulting standard deviations of σ and δ, σ_σ and σ_δ, which are the lowest (except for σ_δ in Table 5.2). The low σ_σ and σ_δ values indicate that in the 5 trials the DE produced 5 final solutions whose parameters' values had low dispersion around the respective average values $\overline{\sigma}$ and $\overline{\delta}$.

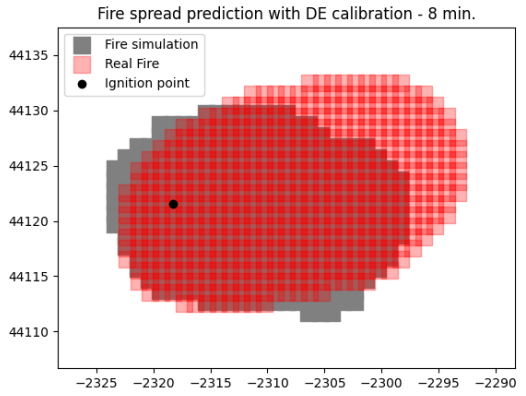
Regarding the overall two-stage methodology, as stated before, the results show that



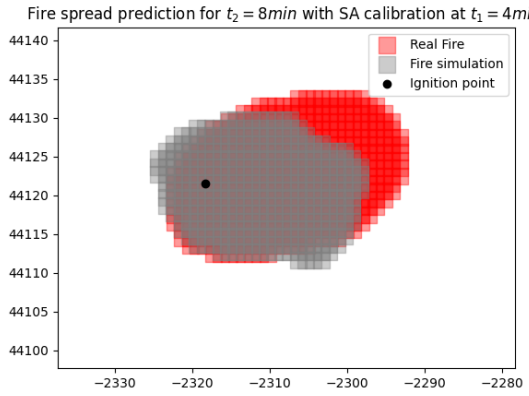
(a) Non-calibrated prediction.



(b) GA calibration.

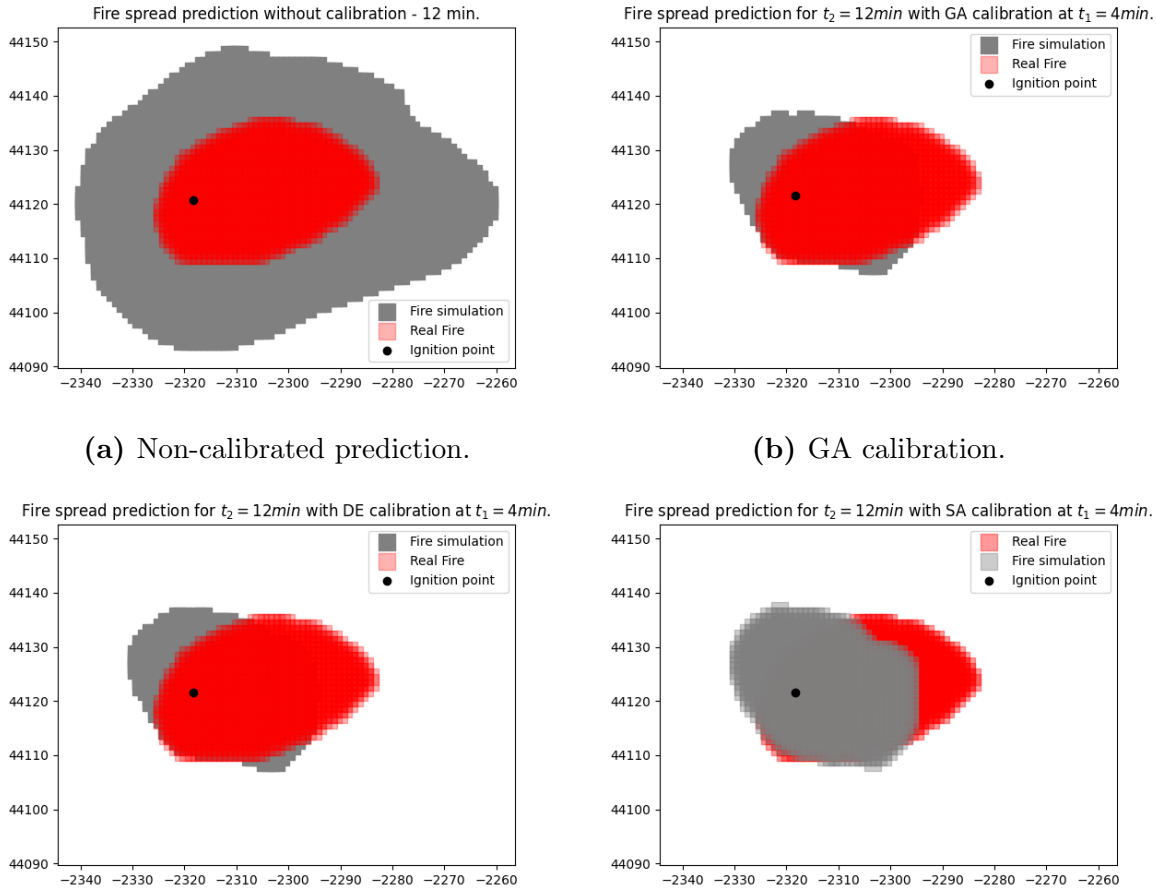


(c) DE calibration.



(d) SA calibration.

Figure 5.4: Scenario 1: fire spread predictions for $t_2 = 8\,min$, with calibration performed using the real fire area from $t_1 = 4\,min$.



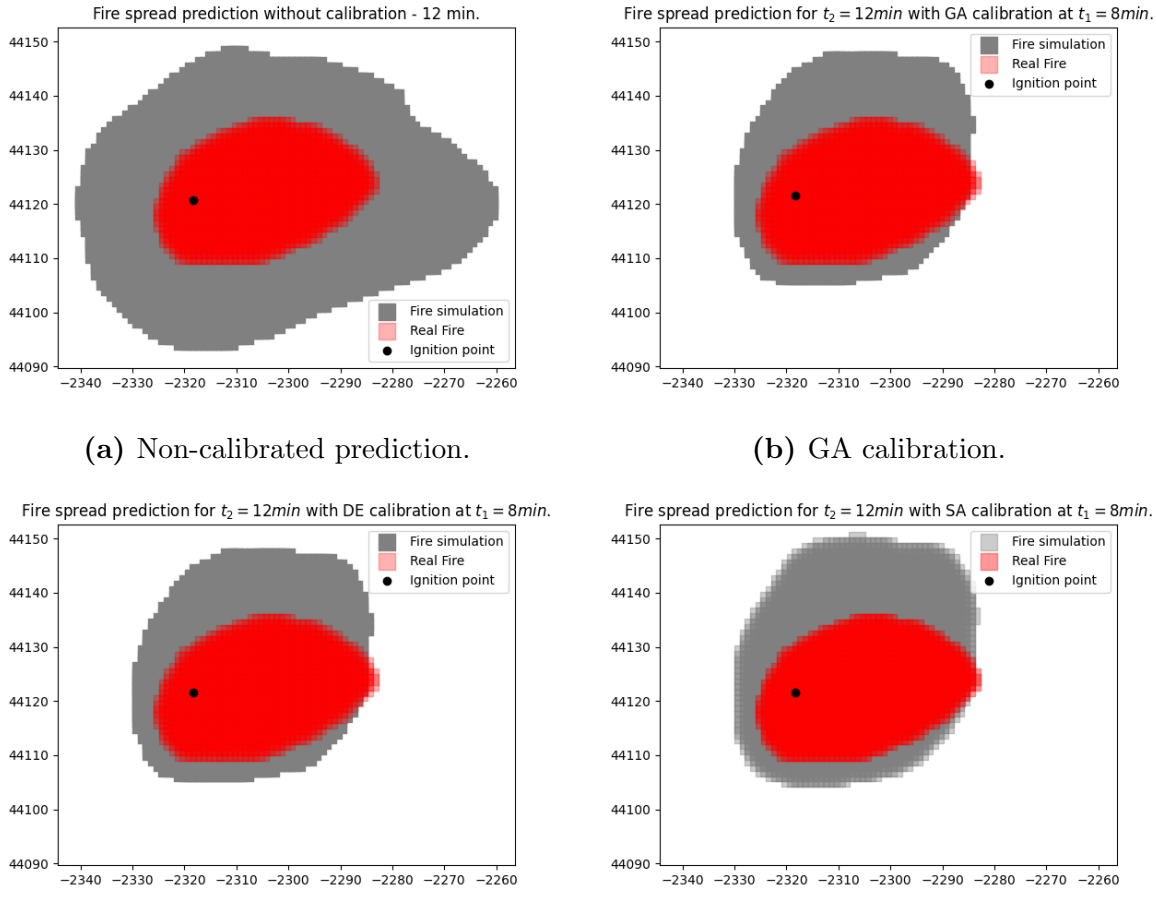
(a) Non-calibrated prediction.

(b) GA calibration.

(c) DE calibration.

(d) SA calibration.

Figure 5.5: Scenario 2: fire spread predictions for $t_2 = 12\,min$, with calibration performed using the real fire area from $t_1 = 4\,min$.



(a) Non-calibrated prediction.

(b) GA calibration.

(c) DE calibration.

(d) SA calibration.

Figure 5.6: Scenario 3: fire spread predictions for $t_2 = 12\,min$, with calibration performed using the real fire area from $t_1 = 8\,min$.

having an intermediate stage for calibrating the simulator using the obtained real fire data results in much better fire spread predictions. This result proves the importance of the two-stage methodology described in Section 2 and is inline with the results found in the literature review. When analyzing more carefully the prediction results for $t_2 = 12\,min$, it is possible to verify that executing calibration with the data from $t_1 = 4\,min$ resulted in better predictions (lower values of RSD_C) than for the calibrations performed with the data from $t_1 = 8\,min$. A possible explanation for this may be the error propagation inside FIRESTATION: the small errors present in the $4\,min$ simulations become successfully larger for the $8\,min$ and $12\,min$ simulations. However, the prediction results RSD_C were, for every situation, better than the corresponding predictions results without calibration RSD_{NC}.

5.4 Conclusions

Similarly to Chapter 4, the results from this chapter demonstrate the quality of the differential evolution algorithm as a calibration algorithm for fire spread models, which is an area where genetic algorithm are the predominant calibration technique. Moreover, the importance of calibrating the FIRESTATION's input parameters became clear, as was already suggested in the literature review. Consequently, the advantage of using the two-stage methodology in real wildfire applications was shown by the results of this chapter. Additionally, it is important to comment on Figures 5.4, 5.5 and 5.6. These figures have the objective of showing the difference between the real (in red) and simulated (in grey) fire propagations. However, they do not accurately depict the terrain in which the controlled fire occurred, specifically, the slope of the terrain and the geographical orientation. The images were obtained by scattering the burned cells from the real fire and the simulation. The two scattered sets were then overlapped.

Chapter 6

Conclusions

6.1 Conclusions

The initial main objective of the work presented in this document was to improve the accuracy of fire propagation prediction by calibrating the Rothermel model's input parameters. The three implemented metaheuristic algorithms produced quality input parameter calibrations which resulted in accurate fire propagation predictions, in relation to the non-calibrated model, as demonstrated in Chapter 5. Additionally, the three algorithms were implemented in a practical way, as proposed in the requirements definition for they allow the user to easily change the calibrated Rothermel input parameters and the algorithms intrinsic parameters, and perform adaptations to their operating structure.

In Section 1.4, the project requirements were defined. It is then important to analyze the results of the work developed and understand if they comply with the requirements. Regarding the first non-functional requirement defined in Section 1.4: the developed algorithms were implemented in parallel, as far as possible. The genetic algorithm and differential evolution are population-based algorithms, so they are suited to this in the sense that the individuals in a population are independent from each other so they can be evaluated separately, as suggested in Section 2.3.2 of the literature review. Simulated annealing was implemented using greedy search which, as described in Section 3.4, generates a small population of new solutions in each iteration. Similarly to the other two algorithms, this small population was also evaluated and managed in parallel given that the solutions are also independent from each other. The next non-functional requirement was that the average prediction error of the calibrated model should have an improvement of at least 40% in relation to the error from the non-calibrated model. Since calibrated fire propagation predictions were only performed in Chapter 5 and as described in Section 5.3, it is possible to verify that the all predictions using calibrated parameters resulted in error reductions larger than 40%. In fact, the smallest error reductions, from the non calibrated model to the calibrated model were 66.48% for the GA, 66.39% for the DE and 59.31% for the SA, all in Scenario 3. These results confirm that the work developed complies with the second non-functional requirement. Finally, the third non-functional requirement was that the average model calibration time should be inferior to 30 minutes. For this analysis, only the work from Chapter 5 should be considered because the FIRESTATION fire simulator provides more complete information about the fire propagation and is closer to being used in a real fire event. For that matter, the calibrations performed in Chapter 5 were significantly longer than 30 minutes, which means that the third requirement was not fulfilled. Similarly to the previous requirement,

the 30 min calibration requirement was defined based on what would be acceptable in a real wildfire situation. The long calibration times are an important issue to be addressed because the real world functionality of the two-stage methodology for wildfire spread prediction requires reasonable calibration times (in which the model simulations' times are included) and also requires the real-time acquisition of the real wildfire data. For this matter, not only the algorithms should be improved (and their parallel implementation) but mainly the propagation model should be improved, given that the fire propagation simulations consume the biggest part of time during the calibration stage of the two-stage methodology. In fact, as part of the "IMFire" project, the model is being redesigned to be faster. An increase in the number of available computing cores is also necessary, specially considering that the final goal of the "IMFire" project is to produce a decision support system in which there's urgency in obtaining wildfire spread predictions, in a real situations.

Relatively to the objectives and requirements defined for this work, the outcome can be considered as positive. The main objective was accomplished, i.e., the development and application of the metaheuristic algorithms for calibration of the fire spread model. In addition, the validation of the the two-stage methodology using real fire data and a Rothermel-based fire spread model (Chapter 5) and the parallel implementation of the metaheuristic algorithms had the goal of integrating the most important knowledge obtained from the literature review in the "IMFire" project.

6.2 Future work

One of the aspects which should be improved is the parallel implementation of the algorithms. Given that the three algorithms were implemented using MATLAB®, there was never the possibility of using tools such as OpenMP, which was referenced in the literature review and is platform dedicated to parallel programming. So, when integrating the algorithms in a more complete wildfire propagation prediction system, the possibility of improving the developed algorithms and using faster and more robust techniques should be considered.

Additionally, with the aim of improving the results obtained in Chapter 5, more parameters should be considered for calibration. It is known that wind has a heavy influence on fire propagation and it is also a significant source of prediction error due to the possibility of sudden changes in its behavior. Some works in the literature review proposed methodologies in which the wind parameters were calibrated.

The two-stage methodology was tested in Chapter 5 using data from a past controlled fire. A possible and interesting next step for testing the two-stage methodology would be to execute it during a real controlled fire, in real time. As suggested in the previous section, an increase in computing power is probably required in order to obtain solid results, as well as the possibility of obtaining, processing and feeding the real fire data to the model and to the algorithms during the fire itself, dynamically.

Bibliography

[1] Jesus San-Miguel-Ayanz, Tracy Durrant, Roberto Boca, Pieralberto Maianti, Giorgio Libertà, Tomas Artés-Vivancos, Duarte Oom, Alfredo Branco, Daniele de Rigo, Davide Ferrari, Hans Pfeiffer, Rosanna Grecchi, Daniel Nuijten, Marco Onida, and Peter Löffler. Forest Fires in Europe, Middle East and North Africa 2020. Technical report, 2021.

[2] Jorge S. S. Júnior, João Paulo, Jérôme Mendes, Daniela Alves, and Luís Mário Ribeiro. Automatic calibration of forest fire weather index for independent customizable regions based on historical records. In *2020 IEEE Third International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*, pages 1–8. IEEE, December 2020.

[3] Jorge S.S. Júnior, João Ruivo Paulo, Jérôme Mendes, Daniela Alves, Luís Mário Ribeiro, and Carlos Viegas. Automatic forest fire danger rating calibration: Exploring clustering techniques for regionally customizable fire danger classification. *Expert Systems with Applications*, page 116380, 2022.

[4] UNEP and GRID-Arendal. Spreading like Wildfire: The Rising Threat of Extraordinary Landscape Fires. Technical report, United Nations Environment Programme, 2022.

[5] François-Nicolas Robinne. Impacts of disasters on forests, in particular forest fires. Technical report, United Nations Forum on Forests Secretariat, 2021.

[6] Elsa Pastor, Luis Zárate, Eulalia Planas, and Josep Arnaldos. Mathematical models and calculation systems for the study of wildland fire behaviour. *Progress in Energy and Combustion Science*, 29(2):139–153, December 2003.

[7] Richard C. Rothermel. *A mathematical model for predicting fire spread in wildland fuels*. Forest Service, U. S. Department of Agriculture, 1972.

[8] Andrew Sullivan. Wildland surface fire spread modelling, 1990–2007. 2: Empirical and quasi-empirical models. *International Journal of Wildland Fire*, 18:369–386, July 2009.

[9] Mark A. Finney. FARSITE: Fire Area Simulator-model development and evaluation. Technical report, U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station, 1998.

[10] Antonio Gameiro Lopes, Miguel Cruz, and Domingos Viegas. FireStation — an integrated software system for the numerical simulation of fire spread on complex topography. *Environmental Modelling & Software*, 17:269–285, December 2002.

[11] Collin D. Bevins. *FireLib User Manual and Technical Reference*. 1996.

[12] Frank A. Albini. Estimating wildfire behavior and effects. Technical report, USDA Forest Service, Intermountain Forest and Range Experiment Station, 1976.

[13] Martin E. Alexander and Miguel G. Cruz. Limitations on the accuracy of model predictions of wildland fire behaviour: A state-of-the-knowledge overview. *The Forestry Chronicle*, 89(03):372–383, 2013.

[14] Jérôme Mendes, Ricardo Seco, and Rui Araújo. Automatic extraction of the fuzzy control system for industrial processes. In *Proc. 16th IEEE International Conference on Emerging Technologies and Factory Automation*, pages 1–8. IEEE, September 2011.

[15] Jérôme Mendes, Rui Araújo, Tiago Matias, Ricardo Seco, and Carlos Belchior. Evolutionary Learning of a Fuzzy Controller for Industrial Processes. In *Proc. of The 40th Annual Conference of the IEEE Industrial Electronics Society (IECON 2014)*, pages 139–145, October 29 - November 1 2014.

[16] Luís Laím, Jérôme Mendes, Hélder D. Craveiro, Aldina Santiago, and Carlos Melo. Structural optimization of closed built-up cold-formed steel columns. *Journal of Constructional Steel Research*, 193:107266, 2022.

[17] Ricardo Maia, Jérôme Mendes, and Rui Araújo. Electric vehicle physical parameters identification by evolutionary algorithm. In *48th Annual Conference of the IEEE Industrial Electronics Society (IECON 2022)*, pages 1–6. IEEE, October 2022.

[18] Baker Abdalhaq, Ana Cortés, Tomàs Margalef, Emilio Luque, and Domingos Xavier Viegas. Optimization of Parameters in Forest Fire Propagation Models. *Forest Fire Research & Wildland Fire Safety*, pages 1–13, January 2002.

[19] Mónica Denham, Kerstin Wendt, Germán Bianchini, Ana Cortés, and Tomàs Margalef. Dynamic Data-Driven Genetic Algorithm for forest fire spread prediction. *Journal of Computational Science*, 3(5):398–404, September 2012.

[20] Davide Ascoli, Giovanni Bovio, and Giorgio Vacchiano. *Calibrating Rothermel's fuel models by genetic algorithms*, pages 102–106. Imprensa da Universidade de Coimbra, Coimbra, 2014.

[21] Jorge Pereira, Jérôme Mendes, Jorge S. S. Júnior, Carlos Viegas, and João Ruivo Paulo. A Review of Genetic Algorithm Approaches for Wildfire Spread Prediction Calibration. *Mathematics*, 10(3), 2022.

[22] Jorge Pereira, Jérôme Mendes, Jorge S. S. Júnior, Carlos Viegas, and João Ruivo Paulo. Wildfire spread prediction model calibration using metaheuristic algorithms. In *48th Annual Conference of the IEEE Industrial Electronics Society (IECON 2022)*, pages 1–6. IEEE, October 2022.

[23] Patricia L. Andrews. *The Rothermel Surface Fire Spread Model and Associated Developments: A Comprehensive Explanation*. Rocky Mountain Research Station, Forest Service, United States Department of Agriculture, Fort Collins, Colorado, 2018.

[24] Roque Rodríguez, Ana Cortés, and Tomàs Margalef. Injecting dynamic real-time data into a DDDAS for forest fire behavior prediction. In *Proc. 9th International Conference on Computational Science*, pages 489–499. Springer Berlin Heidelberg, May 2009.

[25] Mónica Denham, Ana Cortés, Tomàs Margalef, and Emilio Luque. Applying a dynamic data driven genetic algorithm to improve forest fire spread prediction. In *Proc. 8th International Conference on Computational Science*, pages 36–45, June 2008.

[26] Stefano Chelli, Pierluigi Maponi, Giandiego Campetella, Paolo Monteverde, Monica Foglia, Eleonora Paris, Andreas Lolis, and Thomas Panagopoulos. Adaptation of the canadian fire weather index to mediterranean forests. *Natural Hazards*, 75:1795—1810, September 2014.

[27] Carlos Brun, Tomàs Artés, Tomàs Margalef, and Ana Cortés. Coupling Wind Dynamics into a DDDAS Forest Fire Propagation Prediction System. In *Proc. 12th International Conference on Computational Science*, pages 1110–1118, June 2012.

[28] Baker Abdalhaq, Ana Cortés, Tomàs Margalef, and Emilio Luque. Enhancing wildland fire prediction on cluster systems applying evolutionary optimization techniques. *Future Generation Computer Systems*, 21(1):61–67, January 2005.

[29] Mónica Denham, Ana Cortés, and Tomàs Margalef. Computational steering strategy to calibrate input variables in a dynamic data driven genetic algorithm for forest fire spread prediction. In *Proc. 9th International Conference on Computational Science*, pages 479–488, Berlin, Heidelberg, May 2009. Springer Berlin Heidelberg.

[30] W. W. Group. Weather research and forecasting (WRF) model. Technical report, Director (INT-115), 2015.

[31] Richard Sneeuwjagt and William Frandsen. Behavior of experimental grass fires vs. predictions based on rothermel's fire model. *Canadian Journal of Forest Research*, 7(2):357–367, June 1977.

[32] Brian Van Wilgen, David Le Maitre, and FJ Kruger. Fire behaviour in south african fynbos (macchia) vegetation and predictions from rothermel's fire model. *Journal of Applied Ecology*, 22(1):207–216, 1985.

[33] David Sandberg, Cynthia Riccardi, and Mark Schaaf. Reformulation of rothermel's wildland fire behaviour model for heterogeneous fuelbeds. *Canadian Journal of Forest Research*, 37:2438–2455, December 2007.

[34] Tomás Artès, Adrián Cardil, Ana Cortés, Tomàs Margalef, Domingo Molina, Lucas Pelegrín, and Joaquín Ramírez. Forest Fire Propagation Prediction Based on Overlapping DDDAS Forecasts. In *Proc. 15th International Conference on Computational Science*, pages 1623–1632, June 2015.

[35] Joaquin Ramirez, Santiago Monedero, and David Buckley. New approaches in fire simulations analysis with Wildfire Analyst. In *5th International Wildland fire conference*, Sun City, South Africa, May 2011.

[36] Tomàs Artés, Ana Cortés, and Tomàs Margalef. Large forest fire spread prediction: Dataand computational science. In *Proc. 12th International Conference on Computational Science*, pages 909–918, June 2016.

[37] Andrés Cencerrado, Ana Cortés, and Tomàs Margalef. Genetic algorithm characterization for the quality assessment of forest fire spread prediction. In *Proc. 12th International Conference on Computational Science*, pages 312–320, June 2012.

[38] Edigley Fraga, Ana Cortés, Andrés Cencerrado, Porfidio Hernández, and Tomàs Margalef. Early adaptive evaluation scheme for data-driven calibration in forest fire spread prediction. In *Proc. 20th International Conference on Computational Science*, pages 17–30. Springer, June 2020.

[39] Tomàs Artés, Andrés Cencerrado, Ana Cortés, and Tomàs Margalef. Relieving the Effects of Uncertainty in Forest Fire Spread Prediction by Hybrid MPI-OpenMP Parallel Strategies. In *Proc. 13th International Conference on Computational Science*, pages 2278–2287, June 2013.

[40] Andrés Cencerrado, Ana Cortés, and Tomàs Margalef. On the way of applying urgent computing solutions to forest fire propagation prediction. In *Proc. 12th International Conference on Computational Science*, pages 1657–1666. Elsevier BV, December 2012.

[41] Andrés Cencerrado, Tomàs Artés, Ana Cortés, and Tomàs Margalef. Relieving uncertainty in forest fire spread prediction by exploiting multicore architectures. In *Proc. 15th International Conference on Computational Science*, pages 1752–1761, June 2015.

[42] Tomàs Artés, Andrés Cencerrado, Ana Cortés, and Tomàs Margalef. Time aware genetic algorithm for forest fire propagation prediction: exploiting multi-core platforms. *Concurrency and Computation: Practice and Experience*, 29(9):1–18, January 2016.

[43] Davide Ascoli, Michele Lonati, Raffaella Marzano, Giovanni Bovio, Andrea Cavallero, and Giampiero Lombardi. Prescribed burning and browsing to control tree encroachment in southern european heathlands. *Forest Ecology and Management*, 289:69–77, September 2013.

[44] Giorgio Vacchiano, Renzo Motta, Giovanni Bovio, and Davide Ascoli. Calibrating and testing the forest vegetation simulator to simulate tree encroachment and control measures for heathland restoration in southern europe. *Forest Science*, 60:241–252, April 2014.

[45] John Henry Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.

[46] S.N. Sivanandam and S. N. Deepa. *Introduction to Genetic Algorithms*. Springer-Verlag Berlin Heidelberg, 2008.

[47] Jérôme Mendes, Francisco Souza, Rui Araújo, and Nuno Gonçalves. Genetic fuzzy system for data-driven soft sensors design. *Applied Soft Computing*, 12(10):3237–3245, October 2012.

[48] Jérôme Amaro Pires Mendes. *Computational Intelligence Methodologies for Control of Industrial Processes*. Ph.d. thesis, Department of Electrical and Computer Engineering, University of Coimbra, Coimbra, Portugal, October 2014.

[49] António Gaspar-Cunha, Ricardo Takahashi, and Carlos Henggeler Antunes. *Manual de computação evolutiva e metaheurística*. Coimbra University Press, June 2012.

[50] Rainer Storn and Kenneth Price. Differential Evolution: A Simple and Efficient Adaptive Scheme for Global Optimisation Over Continuous Spaces. *Journal of Global Optimization*, 23, 1995.

[51] Kenneth Price, Rainer Storn, and Jouni Lampinen. *Differential Evolution-A Practical Approach to Global Optimization*. Springer, 2005.

[52] Scott Kirkpatrick, C. D. Gelatt, and M. Vecchi. Optimization by Simulated Annealing. *Science*, 220:671–680, 1983.

[53] Sangmin Lee and Seoung Bum Kim. Parallel Simulated Annealing with a Greedy Algorithm for Bayesian Network Structure Learning. *IEEE Transactions on Knowledge and Data Engineering*, 32(6):1157–1166, 2020.

[54] Hal Anderson. Aids to determining fuel models for estimating fire behavior. Technical report, US Department of Agriculture, Forest Service, Intermountain Forest and Range, 1982.

[55] Sérgio Lopes, Domingos Xavier Viegas, Luís Teixeira de Lemos, and Maria Teresa Viegas. Equilibrium moisture content and timelag of dead pinus pinaster needles. *International Journal of Wildland Fire*, 23:721–732, June 2014.

[56] Carlos G Rossa. A generic fuel moisture content attenuation factor for fire spread rate empirical models. *Forest Systems*, 27:1–8, June 2018.

[57] Domingos Xavier Filomeno Carlos Viegas, Jorge Rafael Nogueira Raposo, Carlos Fernando Morgado Ribeiro, Luís Carlos Duarte Reis, Abdelrahman Abouali, and Carlos Xavier Pais Viegas. On the non-monotonic behaviour of fire spread. *International journal of wildland fire*, 30(9):702–719, 2021.

[58] Paulo A Martins Fernandes. Fire spread prediction in shrub fuels in portugal. *Forest ecology and management*, 144(1-3):67–74, 2001.

[59] Patricia L. Andrews. The Rothermel surface fire spread model and associated developments: A comprehensive explanation. Technical report, U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station, 2018.

[60] Hal E. Anderson. Predicting wind-driven wild land fire size and shape, 1983.

[61] Martin Alexander. Estimating the length-to-breadth ratio of elliptical forest fire patterns. In *Proceedings of the Eighth Conference on Fire and Forest Meteorology*, pages 287–304, 1985.

[62] D. G. Ross, I. N. Smith, P. C. Manins, and D. G. Fox. Diagnostic wind field modeling for complex terrain: Model development and testing. *Journal of Applied Meteorology and Climatology*, 27(7):785 – 796, 1988.

Appendix A

Paper published on Mathematics journal

mathematics

MDPI

Review

A Review of Genetic Algorithm Approaches for Wildfire Spread Prediction Calibration

Jorge Pereira [1], Jérôme Mendes [1,*], Jorge S. S. Júnior [1], Carlos Viegas [2] and João Ruivo Paulo [1]

[1] Department of Electrical and Computer Engineering, Institute of Systems and Robotics, University of Coimbra, Pólo II, 3030-290 Coimbra, Portugal; jorge.pereira@isr.uc.pt (J.P.); jorge.silveira@isr.uc.pt (J.S.S.J.); jpaulo@isr.uc.pt (J.R.P.)
[2] Association for the Development of Industrial Aerodynamics, University of Coimbra, 3030-289 Coimbra, Portugal; carlos.viegas@uc.pt
* Correspondence: jermendes@isr.uc.pt

Abstract: Wildfires are complex natural events that cause significant environmental and property damage, as well as human losses, every year throughout the world. In order to aid in their management and mitigate their impact, efforts have been directed towards developing decision support systems that can predict wildfire propagation. Most of the available tools for wildfire spread prediction are based on the Rothermel model that, apart from being relatively complex and computing demanding, depends on several input parameters concerning the local fuels, wind or topography, which are difficult to obtain with a minimum resolution and degree of accuracy. These factors are leading causes for the deviations between the predicted fire propagation and the real fire propagation. In this sense, this paper conducts a literature review on optimization methodologies for wildfire spread prediction based on the use of evolutionary algorithms for input parameter set calibration. In the present literature review, it was observed that the current literature on wildfire spread prediction calibration is mostly focused on methodologies based on genetic algorithms (GAs). Inline with this trend, this paper presents an application of genetic algorithms for the calibration of a set of the Rothermel model's input parameters, namely: surface-area-to-volume ratio, fuel bed depth, fuel moisture, and midflame wind speed. The GA was validated on 37 real datasets obtained through experimental prescribed fires in controlled conditions.

Keywords: wildfire; wildfire spread prediction; calibration; genetic algorithm; evolutionary algorithms

check for updates

Citation: Pereira, J.; Mendes, J.; Júnior, J.S.S.; Viegas, C.; Paulo, J.R. A Review of Genetic Algorithm Approaches for Wildfire Spread Prediction Calibration. *Mathematics* **2022**, *10*, 300. https://doi.org/10.3390/math10030300

Academic Editor: Ioannis G. Tsoulos

Received: 16 December 2021
Accepted: 13 January 2022
Published: 19 January 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Wildfires are one of nature's most dangerous hazards and, in the last few years, their impact has been increasing significantly, as reported by the European Commission's 20th issue of the annual wildfire report [1–3]. This report, from 2019, shows a total burned area of 789,730 (ha) registered for 40 countries from Europe, the Middle East, and North Africa. This number is nearly four times larger than the records for the previous year (2018). Wildfires can impact ecosystems by destroying natural habitats, resources, and wildlife. Furthermore, they cause significant damage to society, being responsible for numerous fatalities, accidents, injuries, health problems, and the destruction of human infrastructures. These damages bear a significant economic impact, not only due to the fire damage but also the large investments in prevention, preparedness, fire suppression and recovery efforts [4]. It is essential to direct efforts towards understanding the behavior of wildfires and improving their management. In this sense, knowledge of how wildfires propagate is critical, allowing the prediction of where the fire will be and taking the appropriate measures to mitigate its impact.

Theoretical, empirical and semiempirical models have been developed to predict the wildfire behavior [5]. The semiempirical Rothermel model [6] is the most widely used model for wildfire spread prediction [5], particularly in Mediterranean European countries [7], being the core of some of the most cited fire simulators such as FARSITE [8]

and FIRESTATION [9]. The Rothermel model uses several input parameters related to the available forest fuels, such as trees, grass or bushes (surface-area-to-volume ratio, height and moisture content), the terrain configuration (slope), and atmospheric conditions (wind speed and direction).

The quality of the fire spread prediction depends on the quality of the propagation model, and on the accuracy of the input parameters [10]. The present work focuses on the latter cause of uncertainty in wildfire spread predictions. As a matter of fact, while some variables remain constant throughout the whole fire event or can be obtained with a high degree of accuracy (e.g., terrain slope), other variables may change due to fire and cannot be obtained with enough temporal or spatial resolution (e.g., fuel characteristics and wind speed/direction). This uncertainty in the input parameters results in considerable deviations between the predicted and the real fire spread. In order to improve the fire spread simulations/predictions, it is essential to deal with this uncertainty in the Rothermel model input parameters. In an effort to find the accurate input parameters values for the wildfire prediction, some methodologies based on Evolutionary Algorithms (EAs) have been proposed to calibrate the Rothermel model [11]. EAs, such as genetic algorithms (GA), ant colony optimization (ACO), and particle swarm optimization (PSO), have proven their effectiveness for optimization/calibration problems [12–14].

In this paper, we present a review of genetic algorithm approaches for wildfire spread prediction calibration. The main contributions of the paper are:

- A literature review focused on wildfire spread prediction calibration using GAs is performed. The GA was chosen as a technique for the calibration due to its predominance in research works that used EAs to calibrate the wildfire spread prediction model;
- Based on the presented literature review, in a didactic way, wildfire spread calibration using genetic algorithm is described, in which a specific GA framework for Rothermel model calibration is presented. Moreover, the parameters to be calibrated are discussed, namely the surface-area-to-volume ratio (σ), fuel bed depth (δ), fuel moisture (M_f), and midflame wind speed (U);
- The actual feasibility of using GAs for the calibration of the Rothermel model for wildfire spread prediction is explored/studied on 37 real datasets.

The results show a significant error reduction in the wildfire spread prediction, i.e., from 95% to 10%.

This paper is organized as follows. Section 2 contains a description of the Rothermel model, as well as an insight into the current state of the art regarding methods of wildfire spread prediction using genetic algorithms. In Section 3, GAs are revised, and the method used in this paper to calibrate the Rothermel model is presented. In Section 4, the results of the proposed calibration are presented and analyzed. Finally, Section 5 presents the final conclusions.

2. Literature Review of Wildfire Spread Prediction Calibration

Genetic algorithms are the most adopted technique for calibration of the Rothermel model's input parameters. Due to the importance of this subject for wildfire spread prediction, and due to the number of latest developments in this particular field, a literature review of the most relevant work in this area is fundamental.

The search process for the presented literature review was performed by using the Science Direct and IEEE Xplore databases and defining the following search keywords: ("fire spread" OR "fire prediction" OR "fire rate of spread" OR "Rothermel model") AND ("genetic algorithm" OR "evolutionary algorithm" OR "calibration" OR "tuning"). The years considered for the search were from 2000 until 2021. Additionally, the references of the selected papers were also analyzed and served as a source for finding new papers. The literature review rationale for article selection was based on the following criteria:

- Acceptance
  1. The article uses the Rothermel model or a Rothermel model-based simulator for fire propagation prediction/simulation;

2. The article uses evolutionary algorithms for Rothermel model calibration;
3. The article focuses on improving the prediction results or its execution time.

- Rejection
  1. The article's method for fire propagation prediction is not based on the Rothermel model;
  2. The article implements calibration techniques other than evolutionary algorithms.

Based on this process, 15 papers were obtained.

2.1. Rothermel Model

The Rothermel model, proposed in [6], estimates a Rate Of Spread R of a fire front, given by

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

which is measured in units of distance per unit of time ([m/s] or [ft/min]), and it represents the linear velocity of a fire, in a given direction and set of conditions. The equations of the associated factors in (1) $I_R(\rho_p, \sigma, \delta, w_0, S_T, h, M_x, M_f, S_e)$, $\xi(\sigma, \rho_p, w_0, \delta)$, $\phi_w(\rho_p, w_0, \delta, \sigma, U)$, $\phi_s(\rho_p, w_0, \delta, tan\phi)$, $\rho_b(w_0, \delta)$, $\varepsilon(\sigma)$, and $Q_{ig}(M_f)$ depend on several input parameters and are given by:

$$I_R = \Gamma' w_n h \eta_M \eta_S \tag{2}$$

$$\Gamma' = \Gamma'_{max} \left(\frac{\beta}{\beta_{op}}\right)^A exp\left[A\left(1 - -\frac{\beta}{\beta_{op}}\right)\right] \tag{3}$$

$$A = 133\sigma^{-0.7913} \tag{4}$$

$$\beta = \frac{\rho_b}{\rho_p} \tag{5}$$

$$\rho_b = \frac{w_0}{\delta} \tag{6}$$

$$\Gamma'_{max} = \frac{\sigma^{1.5}}{(495 + 0.0594\sigma^{1.5})} \tag{7}$$

$$\beta_{op} = 3.348\sigma^{-0.8189} \tag{8}$$

$$w_n = w_0(1 - S_T) \tag{9}$$

$$\eta_M = 1 - -2.59 r_M + 5.11(r_M)^2 - -3.52(r_M)^3 \tag{10}$$

$$r_M = \frac{M_f}{M_x}\ (max = 1.0) \tag{11}$$

$$\eta_S = 0.174 S_e^{-0.19}\ (max = 1.0) \tag{12}$$

$$\xi = \frac{exp[(0.792 + 0.681\sigma^{0.5})(\beta + 0.1)]}{(192 + 0.2595\sigma)} \tag{13}$$

$$\phi_w = C U^B \left(\frac{\beta}{\beta_{op}}\right)^{-E} \tag{14}$$

$$C = 7.47 exp(-0.133\sigma^{0.55}) \tag{15}$$

$$B = 0.02526\sigma^{0.54} \tag{16}$$

$$E = 0.715 exp(-3.59 \times 10^{-4}\sigma) \tag{17}$$

$$\phi_S = 5.275\beta^{-0.3}(tan\phi)^2 \tag{18}$$

$$\varepsilon = exp\left(\frac{-138}{\sigma}\right) \tag{19}$$

$$Q_{ig} = 250 + 1116 M_f \tag{20}$$

where the description of the respective parameters is presented in Table 1.

Table 1. Identification of the parameters in Equations (2)–(20) [6,15].

Parameter	Description
I_R	Reaction intensity (Btu/ft^{2}min)
Γ'	Optimum reaction velocity (min^{-1})
β	Packing ratio
ρ_b	Oven-dry bulk density (lb/ft^3)
Γ'_{max}	Maximum reaction velocity (min^{-1})
β_{op}	Optimum packing ratio
w_n	Net fuel load (lb/ft^2)
η_M	Moisture damping coefficient
η_S	Mineral damping coefficient
ξ	Propagating flux ratio
ϕ_w	Wind factor
ϕ_S	Slope factor
ε	Effective heating number
Q_{ig}	Heat of preignition (Btu/lb)

The input parameters of the Rothermel model (1) can be separated into three categories: fuel properties, topography and wind properties. The fuel properties are heat content (h), mineral content (S_T (total) and S_e (effective)), oven-dry particle density (ρ_p), oven-dry fuel load (w_0), surface-area-to-volume ratio (σ), fuel bed depth (δ), dead fuel moisture of extinction (M_x) and fuel moisture (M_f). Topography is represented by slope steepness ($\tan\phi$), and wind properties correspond to the midflame wind speed (U). A deeper insight into the Rothermel model can be seen in [6,15].

2.2. The Need for a Fire Spread Model Calibration

Figure 1 presents a general illustration for wildfire spread prediction, which consists in feeding a fire simulator with a set of input parameters that aim to represent the initial real fire conditions, at t_0. The result of the fire simulator, i.e., the simulated wildfire perimeter, at t_1, should match the propagation of the real wildfire, i.e., the real wildfire perimeter [16]. However, the input parameters are related to the environmental conditions, e.g., fuel, weather, and terrain characteristics as described in Section 2.1, and obtaining them becomes a difficult task in order to provide an accurate prediction.

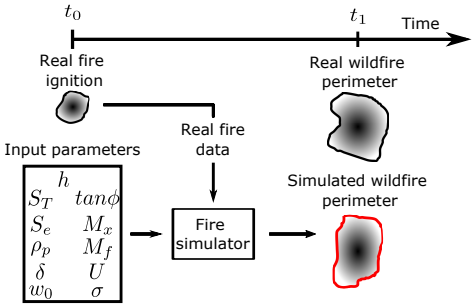


Figure 1. Illustration of fire spread prediction using only one set of non-calibrated input parameters. Adapted from [17].

In more detail, some input parameters can be directly measured, such as terrain slope, which can also be obtained based on previous topographical information. However, other parameters, such as fuel-specific parameters, require detailed knowledge about the local vegetation, which might not be available. Some input parameters, such as fuel moisture, are calculated using models based on meteorological data [18], while wind field maps are

estimated based on point observations from the available meteorological stations closer to the fire location. These estimations introduce a great amount of error in the prediction. In terms of behavior change, characteristics such as the terrain slope and the type of vegetation in a certain region are constant in time and space, while others, such as wind speed and direction, have very sudden variations during the wildfire [10]. Therefore, finding a set of input parameters that produces accurate results solely based on previous knowledge about the wildfire location and weather conditions is a challenging task. Due to the uncertainty and the consequent inaccuracy in wildfire spread simulation, there is a need to calibrate the input parameters.

2.3. Wildfire Spread Calibration Literature Overview

The Rothermel model is the most used and recognized fire spread prediction model, serving as the base for several fire simulators (FARSITE [8] and FIRESTATION [9]). Research works that deal with Rothermel model calibration and wildfire spread prediction mostly use genetic algorithms. Initially, works such as [19,20] have proved the performance of genetic algorithms by comparing them against other optimization techniques and with implementation in a parallel two-stage prediction framework. More recently, other works such as [17,21] aim to improve the calibration by merging the algorithms with other tools that complement their performance, such as the Statistical System for Forest Fire Management (S^2F^2M) and WildFire Analyst (WFA) (a component of the Tecnosylva Incident Management software suite designed to directly support multi-agency wild-fire incident management). Given that the quality of genetic algorithms was proven early, works evolved into directing efforts to improve their performance.

One of the areas explored to improve the performance of genetic algorithms is parallel computing. Several works used parallel implementations of genetic algorithms to reduce calibration time. In general, these strategies consisted of implementing a simulator's intrinsic functions in parallel and allocating more processing cores to individuals (elements of a population that represent one possible solution for the problem) with longer predicted execution times.

In the following sections, the main works dealing with this topic are provided, providing a perspective of the philosophy currently being pursued in this research field.

2.4. Wildfire Spread Calibration Literature Using Genetic Algorithms

Genetic algorithms have been used to find the set of input parameters that better adjusts the wildfire spread model predictions to the real observations. In other words, optimizing the model using a framework for wildfire spread prediction tuning.

The authors in [20] introduced a framework, illustrated in Figure 2, that consists of two stages: a calibration stage and a prediction stage. After the ignition, the calibration stage starts, at t_0. Sets of Rothermel's input parameters are generated (using an optimization approach). Each set of input parameters is evaluated, at instant of time t_1, by comparing the simulator prediction with the real observed fire data for that time instance. The optimal set of input parameters is the one that minimizes the deviation between the predicted and the real fire perimeter. This process is repeated several times or until a certain solution criterion is reached. In the prediction stage, assuming that environmental conditions remain constant, the resulting optimal set of parameters from the calibration stage is used as input for the fire simulator to predict the fire spread at every instant of time t_i ($i \in \mathbb{N}$). Here, the prediction stage is similar to the classical method/framework (Figure 1), except that now a tuned set of input parameters is used.

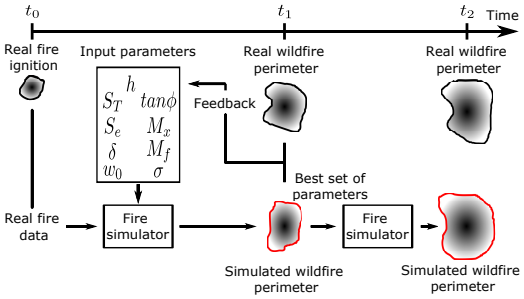


Figure 2. Two-stage method for fire spread prediction, adapted from [17].

During the calibration stage, the goal is to find an optimal solution for the input parameters. In a generic way, the optimization problem can be defined as:

$$\mathbf{x}^* = \arg\min_{\mathbf{x} \in S} F(\mathbf{x}), \tag{21}$$

where $F(\mathbf{x})$ represents the function to be minimized (by an optimization algorithm, such as GA), $\mathbf{x}$ represents the input parameters vector, S is the respective search space, and $\mathbf{x}^*$ represents the input parameters that minimize $F(\mathbf{x})$. A usual function to be optimized in wildfire spread calibration is the difference between the real wildfire rate of spread (measured from the real-time wildfire data) and the predicted rate of spread (obtained by the Rothermel model), or the difference between the real and the predicted burned area. The goal is to find the set of input parameters $\mathbf{x}$ of (21) that most accurately predicts the real fire propagation.

The majority of the works from the current state of the art on wildfire spread prediction are based on the previously presented Two-Stage framework (Figure 2). Early works, such as [19,20], have proposed evolutionary algorithms as techniques that could be used to find an optimal set of input parameters for a fire simulator. Genetic algorithms are included in the group of evolutionary algorithms and they are the dominant optimization technique for input parameter calibration.

In [20], following the presentation of the two-stage framework, a sensitivity analysis was carried out in order to evaluate how the individual variation of each Rothermel input parameter across its range of possible values affects the model output: the bigger the sensitivity of one parameter, the more it affects the model's output. Based on the sensitivity results, an experimental study was conducted to confirm that calibrating parameters with larger sensitivities and fixing the others reduces the GA's search space and accelerates the optimization time. The results showed that, after 1000 generations, the scenarios in which only 6 input parameters were calibrated achieved an improvement in the objective function (XOR area between the real and simulated burned areas) of approximately 33.3% (one third) in relation to the scenario in which 10 input parameters were calibrated. This reduction also matches the reduction in GA's search space from one scenario to the other.

In [19], the genetic algorithm's performance is tested against three other algorithms: Random Search, Tabu Search and Simulated Annealing. The tests were carried out by comparing the simulated fire line based on the sets of parameters generated by the algorithms against a fire line obtained by setting known values for all the inputs and running the *ISStest* simulator for 45 min. Each algorithm was executed 10 times up to 1000 iterations. The fire lines were compared using the Hausdorff distance H (22), which measures the degree of mismatch between two sets of points F_1 and F_2, representing the fire line simulated based on the optimized parameters and the fire line generated with known input parameters for comparison. H (22) is given by

$$H(F_1, F_2) = \max(h(F_1, F_2), h(F_2, F_1)), \tag{22}$$

where $h(F_1, F_2)$ and $h(F_2, F_1)$ represents the Hausdorff distance between two sets of points F_1 and F_2 at a specific point in F_2 and F_1, respectively (see [19] for more details). The results show that simulated annealing, tabu search and genetic algorithms presented similar results after the 500th generation.

In [16], a dynamic data-driven genetic algorithm was proposed to tune the fire simulator's input parameters based on the real fire behavior. The simulator used was *fireLib* and, through reverse engineering, it was possible to obtain equations for wind values (wind speed and direction). These equations are fed with terrain slope with the position (x, y) of the fire front with the maximum rate of spread. The obtained wind speed and direction values were used to steer the search for an optimal input parameter set carried out by the genetic algorithm. Afterwards, in [22], the same research group proposed a new calibration steering method as an improvement to the previous strategy. Since this was highly dependent on the underlying simulator, the new approach consisted of generating a database with fire evolution information from both real and simulated (synthetic) fires. For the calibration stage, a dynamic data-driven genetic algorithm (DDDGA) was proposed to define the best wind direction and wind speed values, by searching the database of previous fires that were similar in terms of rate of spread, slope and fuel model to the real observed fire spread, and using wind values from those fires to steer the genetic algorithm's search.

The authors in [17] introduced a system called *SAPIFE* (Spanish acronym for *Adaptive System for Fire Prediction Based in Statistical-Evolutive Strategies*) which is based on the two-stage fire spread prediction framework with a genetic algorithm implemented during the calibration stage. However, in *SAPIFE*, the genetic algorithm is coupled with another method called the Statistical System for Forest Fire Management (S^2F^2M) [23]. This new method receives a certain population from the GA and analyzes almost all possible input parameter combinations from all individuals in the population. From this analysis, S^2F^2M evaluates the probability of each map cell to be burned or not and generates a probabilistic map. Then, based on these probabilities, the number of possible scenarios (parameter combinations between different individuals) is reduced, decreasing the calibration time required.

In [24], the two methods introduced in [16,22] were compared. The method introduced in [16] is named as the "analytical method" and, as was described above, is based on the inversion of a fire simulator. The method introduced in [22] is named as the "computational method" and relies on a database with information from past fires. Both of these methods use ongoing fire propagation data to obtain wind speed and direction values and use them to steer the genetic algorithm's search. Two sets of tests were carried out: first, the two-stage framework was tested against the classical wildfire spread prediction method, which uses a single set of input parameters introduced in the fire simulator. This test used data from past fires and confirmed that the two-stage framework with a genetic algorithm provides better results than the classical prediction without input parameter calibration. Then, the second set of tests compared the use of a simple non-guided genetic algorithm against genetic algorithms with different configurations of the proposed steering strategies. The guided genetic algorithm with the computational and analytical methods obtained similar results and improved prediction quality over the non-guided genetic algorithm.

The work developed by [10] is also based on the two-stage prediction framework with a genetic algorithm and introduces an approach for reducing the prediction errors caused by the variability of wind parameters (wind speed and direction). During the calibration stage, wind parameters are not calibrated; instead, real wind measurements from the fire location are taken in periodic sub-intervals. These measurements are used as inputs for the fire simulator in the recurring simulations. Afterwards, during the prediction stage, a numerical weather prediction (NWP) model [25] is used to periodically estimate the wind parameters between sub-intervals of the prediction stage. The estimated wind parameters are introduced in the simulator and are updated at each sub-interval. The prediction result is obtained using the real wind measurements and the calibrated parameters, which are moisture contents and vegetation features. The test results showed that, when the wind

conditions are stable, the basic two-stage framework with a genetic algorithm provides satisfactory results, in comparison with the new method of using measured and estimated wind values (prediction error of 0.4 vs. 0.29, respectively). However, when the wind conditions are more dynamic, the results obtained by the introduced method are significantly better compared to the basic two-stage framework with a genetic algorithm (prediction error of 0.19 vs. 0.58 m, respectively).

In [26], a calibration of the fuel models within the Rothermel's fire spread prediction model was carried out through the use of genetic algorithms. The GA's individuals consisted of the following Rothermel fuel parameters: oven-dry fuel load (w_0), surface-area-to-volume ratio (σ), fuel bed depth (δ), fuel moisture of extinction (M_x), and heat content (h). Two tests were performed to evaluate the proposed GA method. The first test consisted of using GAs for the fuel model calibration method, with the support of two works [27,28] (grass and shrub fuels, respectively) that provided datasets of observed rate of spread R and other input parameters' data (fuel moisture, wind speed and slope steepness). The GA was performed with 9999 maximum iterations, 100 individuals, mutation probability and elitism factor equal to 0.1 and 0.05, respectively, and the fuel input parameters calibrated based on the parameter ranges given by the papers. Each individual was evaluated using the Root Mean Square Error (RMSE) between the observed and predicted rate of spread R. The second test consisted of implementing the GA for calibrating a fuel model for a type of vegetation (*Calluna* heath). Nine prescribed fire experiments were carried out in dry *Calluna* heathland vegetation and R, fire weather (1 h fuels moisture, live woody fuel moisture and wind speed) and terrain data (ignition line length, fire plot size and slope) were recorded from each experiment. From the nine fire experiments, four were considered for GA calibration and five were considered for validation. The calibration experiments data were used to run the GA and calibrate the fuel parameters, similarly to the first test. Then, predicted rate of spread R values were calculated using different fuel models: GA calibrated fuel parameters, the Standard Fuel Model which provided the smaller RMSE when comparing predicted vs. observed R, a custom fuel model for *Calluna* vegetation and a "custom fuel model parameterized with modal values from fuels inventoried in each fire experiment". An additional prediction of the rate of spread R was obtained by a Rothermel model reformulation implemented in the Fuel Characteristics Classification System (FCCS) [29]. For the validation experiments data, the calibrated GA fuel parameters resulted in the lowest RMSE between predicted and observed rate of spread R, in comparison to the alternative models.

The study in [21] presents a dynamic data-driven genetic algorithm and introduces a new approach for predicting fire propagation based on Wildfire Analyst (WFA) [30]. The paper describes the two-stage prediction framework with a genetic algorithm, where the fire propagation is simulated using the FARSITE fire simulator [8], and the fitness function corresponds to the symmetric difference between predicted and burned areas obtained by:

$$Difference = \frac{UnionCells - -IntersectionCells}{RealCells - -Init\ Cells}, \tag{23}$$

where *UnionCells* represents the sum of the number of cells that were burned in the predicted area and the real area, *IntersectionCells* is the number of cells burned simultaneously in the predicted area and the real area, *RealCells* is the final number of cells burned in the real area, and *InitCells* is the starting number of cells burned in the real fire area. The newly introduced approach uses WildFire Analyst (WFA) and seeks the best R (Rate of Spread) adjustment factors, minimizing the error between simulated fire and the real fire data. Both the FARSITE fire simulator and Wildfire Analyst use the Rothermel model. Afterwards, the two-stage framework with the genetic algorithm and Wildfire Analyst are coupled together by overlapping their predicted fire spread maps. In order to test the two-stage framework and Wildfire Analyst, experiments were carried out with data from a real fire that occurred in Cardona, Catalonia, Spain in 2005. The results show that both methods adapt to drastic changes in the fire characteristics.

In [31], the two-stage framework was considered to reduce input parameter uncertainty and predict fire spread. However, when the wildfire is large, wind cannot be considered uniform throughout the whole wildfire area. So, this work introduced a wind field model (*WindNinja*), being represented by a cell map, to account for this variation. In essence, during the calibration phase, the obtained meteorological wind parameters are used to calculate the wind field for each scenario generated by the genetic algorithm. Then, having each individual's wind field, the corresponding fire propagation map is calculated and the error function is evaluated.

Finally, in [32], a statistical study was carried out to characterize the genetic algorithm in the calibration phase of the two-stage prediction method. The characterization refers to estimating which GA parameter configuration results in a better calibration within the imposed time restrictions. A statistical study was conducted based on the results of a genetic algorithm calibration on a simulated five-hour fire obtained using FARSITE as the fire spread simulator. The results from this study were maximum adjustment errors which have different degrees of guarantee depending on the number of generations that the GA iterates. These results are important in understanding the compromise between the algorithm's execution time (number of generations) and the adjustment error, which is larger when the algorithm iterates fewer generations.

2.5. Calibration through Parallel Computing

Throughout Section 2.4, several works regarding fire spread prediction using genetic algorithms were described. Despite their focus being on improving prediction accuracy, some works have proposed/adapted a Master/Worker paradigm (Figure 3) in order to reduce the calibration and prediction times.

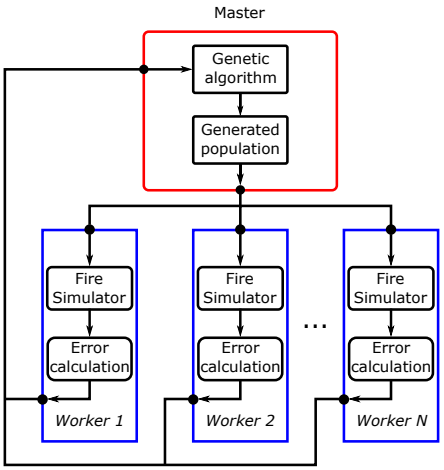


Figure 3. Genetic algorithm using the Master/Worker paradigm, adapted from [33].

GAs, as with any evolutionary algorithm, require the execution of a set of individual simulations through several iterations, which can be very time-consuming, and given the urgency and need for accuracy associated with wildfire spread prediction in real-time, it is important to reduce the execution time of the calibration phase while maintaining appropriate accuracy. One way to achieve this is through the parallel implementation of the fire spread simulator used for the GA individuals' simulation.

The authors in [34] presented a technique based on the parallelization of both the GA (used in the two-stage fire prediction framework) and the FARSITE fire simulator. For the first experiments, with fire simulations of 20 s, the results showed an improvement in GA execution time for reaching the same error (15%) when using more cores per individual.

When replicating the experiments with longer fire simulations (120), the results showed that using more cores per individual still improved execution times for achieving the same error (approximately 14%). However, for the longer fire simulations, using more individuals (100) with one core per individual achieved the lowest error (approximately 8%).

Despite the strategy introduced in [34] improving the calibration time, there is still a drawback related to GA implementation. During the calibration phase, all of the GA individuals have to be simulated. The execution time of a fire simulation depends on the input parameters and, given the random nature of the generation of the population, some individuals will result in much longer simulation times than others. It would be possible to reduce the overall calibration time by dedicating more computing resources to the individuals with larger execution times and fewer resources to individuals that are executed faster. In order to achieve the said time reduction, it is necessary to predict each individual's simulation time to provide more computing resources to those whose predicted execution time is larger. The prediction must be based only on the individuals' genes—a set of input parameters. The study in [34] refers to [35], which introduces a method based on Decision Trees to characterize a fire simulator, allowing estimation of the execution time of one simulation, given a set of input parameters.

In [36], the method referenced in [34] is implemented and tested: Decision Trees are employed to classify each fire simulation according to its execution time so that the Decision Trees can label a new simulation. The core-allocation policy ensures that the individuals labeled with a longer execution time classification are simulated using more computing cores. The results showed that using the core-allocation policy reduced the execution time by 41%, in relation to not using any policy. In [37], similarly to what was done in [36], GA individuals are labeled according to their estimated simulation time through the use of Decision Trees—A, B, C, D and E. Additionally, in this work, an additional restraint is imposed: each GA generation has a limited amount of time to be executed.

More recently, the study in [33] introduced a new strategy to deal with individuals with long execution times. An alternative approach is introduced, based on the monitoring of the fire spread prediction error that, in this particular work, corresponds to the symmetric difference between the real fire and the simulated fire areas, shown in Equation (23). During the execution of one individual, if the monitoring agent detects that the difference between the predicted and the simulated fires is larger than a predefined error threshold, the individual is interrupted. The fitness function is a weighted version of the symmetric difference, shown in Equation (24),

$$Fitness = \frac{PredictionTime}{SimulationTime} \times SymDifference, \tag{24}$$

where *PredictionTime* represents the predicted time for the completion of the individual's simulation, *SimulationTime* is the the time of simulation until the individual is terminated normally or early, and *SymDifference* represents the symmetric difference from (23). This fitness function penalizes individuals that have been terminated early due to a large prediction error: they are not removed from the population, which ensures diversification, but are ranked worst due to lower fitness. This method was tested using fire data from a real fire in La Jonquera, Spain and it reduced the overall execution time in relation to the *Time Aware Core allocation* technique from [34] by 60%.

2.6. Literature Review Summary

The review presented above showed that the majority of the works are based on the two-stage framework formally introduced in [20] in conjunction with the use of genetic algorithms. Genetic algorithms show very good suitability for use as the optimization method in the referenced framework, not only based on their performance when compared to other optimization methods [19], but also because they have characteristics suited for being implemented in parallel. Implementing the two-stage framework with genetic algorithms and fire simulators in parallel is of great importance allowing the reduction of both calibration and prediction execution times [34].

Table 2 contains the above-cited works related to the literature review, organized by characteristics such as the focus of the paper, the source of the data used in experiments and tests and GA's parameters (number of individuals per generation, number of generations, operators probabilities and fitness functions).

Table 2. Review of the literature on wildfire spread prediction calibration using genetic algorithms. The Gens. column contains the number of GA's generations. The Others column contains relevant information such as the GA's operators probabilities and fitness functions. — represents no relevant or existing data. *elitism* represents the percentage of the population's individuals selected for the GA's elitism operation. *#elitism* represents the number of individuals selected for the GA's elitism operation. $cross_{prob}$ is the GA's crossover operation probability. mut_{prob} is the GA's mutation operation probability. RMSE represents the Root Mean Square Error.

Ref.	Focus	Source of Datasets	Individuals	Gens.	Others
[20]	Input parameter calibration. Introduction of two-stage framework + input parameter sensitivity analysis	Simulation (*ISStest*)	1000	20	Fitness function is the XOR area (from *ISStest*) between real and simulated burned areas
[19]	Input parameter calibration using GAs, simulated annealing, random search and tabu search	Simulation (*ISStest*)	1000	-	Fitness function is the Hausdorff distance
[16]	Input parameter calibration	Simulation and 1 prescribed fire (Portugal)	50	5	—
[22]	Input parameter calibration. Two-stage framework with GA and guided search by past fires database	Real map 110×110 m^2. *fireLib* simulation and 1 prescribed fire (Portugal)	Parallel: 512 Dynamic: 50	- 5	—
[17]	Input parameter calibration. Statistical integration to reduce search space	Real fire (California)	500	5	$elitism = 0.04$, $cross_{prob} = 0.2$, $mut_{prob} = 0.01$, Fitness function is symmetric difference (23)
[24]	Input parameter calibration. Two-stage framework with GA and comparison of the methods from [16,22]	1 simulated fire map using *fireLib* and 1 prescribed fire (Portugal)	Simulated: 50 Real: 500	5 5	Real fire case: $0.2 \leq mut_{prob} \leq 0.4$, Fitness function is cell-by-cell comparison of real and simulated fire maps
[10]	Input parameter calibration considering the rapid variation of wind speed and direction	Simulation (FARSITE)	50	10	Tests were performed 15 times
[26]	Rothermel fuel models calibration	1st test (GA-opt.): [27,28]; 2nd test (Custom fuel model calibration): [38,39]	100 for both	Max. 9999	$mut_{prob} = 0.1$, $elitism = 0.05$. Fitness function is RMSE of observed vs. experimental rate of spread R
[21]	Input parameter calibration. Two-stage framework with GA and WildFire Analyst	Real fire (Spain)	-	-	Fitness function is the symmetric difference (23)
[31]	Input parameter calibration, considering the spatial variation of wind in large fires	Real fire (Spain)	6	10	Tests were performed 15 times

Table 2. *Cont.*

Ref.	Focus	Source of Datasets	Individuals	Gens.	Others
[32]	Statistical study of genetic algorithms as the optimization algorithm in the two-stage framework	Simulation (FARSITE)	100	5	Tests were performed 50 times. $mut_{prob} = 0.1$, $elitism = 0.1$
[34]	Reduction of calibration time by parallel implementation	Simulation (FARSITE) based on a real terrain map (Spain)	25; 25; 100	10	Fitness function is the symmetric difference (23)
[36]	Reduction of calibration time by parallel implementation	Simulation (FARSITE) based on a real terrain map (Spain)	25	10	Tests were performed 50 times. Fitness function is the symmetric difference (23)
[37]	Reduction of calibration time by parallel implementation	Real fire (Spain)	–	10	$\#elitism = 10$, $cross_{prob} = 0.7$, $mut_{prob} = 0.3$. Tests were performed 10 times. Fitness function is the symmetric difference (23)
[33]	Reduction of calibration time by early terminating individuals based on prediction error in parallel implementation	Real fire (Spain)	100	10	$cross_{prob} = 0.7$, $mut_{prob} = 0.3$, Fitness function is a weighted version of the symmetric difference (24)

3. Wildfire Spread Calibration Using Genetic Algorithm

From the literature review we verified that, in some articles, there is a lack of details on how the genetic algorithm is implemented for the particular case of wildfire spread prediction calibration, which affects potential attempts for replicability. In this way, based on the presented literature review (Section 2), this section, in a didactic way, presents the use of a genetic algorithm for wildfire spread prediction calibration, where Section 3.1 summarily describes the genetic algorithm, and Section 3.2 presents the application of a genetic algorithm for wildfire spread prediction calibration.

3.1. Genetic Algorithms Overview

Genetic algorithms have proved to be useful in solving a variety of search and optimization problems [40]. In a general way, GAs are stochastic search methods introduced by [41] in 1975 inspired by natural selection and genetics. GAs work by processing a set of elements of a given search space, i.e., a large domain with several possible problem solutions. This set is named the population, and its elements are called individuals. Individuals, which represent the candidate solutions for the optimization problem, are also named chromosomes and are composed of genes. Genes are the primary parts of each solution. Individuals can have several representations depending on the problem: they can be binary sequences of zeros and ones, complex numbers, vectors, among others. The population is evolved/transformed during several generations in order to obtain a final population that contains individuals with the best possible quality for the problem at hand.

A GA generic structure is shown in Algorithm 1. After the encoding of the chromosomes (individuals), usually, a random initialization of the population is performed. Then, all of the individuals are evaluated according to a defined fitness function which measures the ability of a solution (individual) to optimize the fitness function that is specific to the problem being solved. Based on the fitness values of each individual, the selection process occurs where new individuals are chosen to be parents. The Crossover and Mutation reproduction operators and the Replacement operator are applied to the parents in order to breed the offspring and build the next generation. The above GA's operators are repeated until a certain criterion is achieved.

Algorithm 1 General genetic algorithm steps.

1: $g \leftarrow 1$.
2: Generate initial population $P(g)$.
3: **repeat**
4: Evaluate the population $P(g)$ using the defined Fitness Function.
5: Select pair of parents for $P(g+1)$ from $P(g)$ by the defined Selection operator.
6: Generate new population $P(g+1)$ by applying the genetic operators (Crossover, Mutation, and Replacement) to $P(g)$.
7: $g \leftarrow g+1$.
8: **until** Stopping criteria is reached.
9: **Output:** Final Population.

3.2. Calibration Methodology Using Genetic Algorithms

In order to calibrate the Rothermel model (1), the genetic algorithm starts by randomly generating an initial population of N individuals. Each individual is composed of genes, which in this paper consist of Rothermel input parameters to be calibrated. In this paper, four input parameters were selected to be calibrated: σ (surface-area-to-volume ratio), δ (fuel bed depth), M_f (fuel moisture) and U (midflame wind speed). Three main reasons motivated this parameter choice:

(1) the fact that the first three parameters are related to fuel characteristics, which in simulations are approximated using fuel models. Fuel models assume constant and uniform fuel characteristics inside a cell, which is a fair approximation for small cell sizes, a large variety of fuel models and accurate fitting of the model to the existing fuels. However, available fuel maps can suffer from low resolution (large cell sizes), low variety of models (the most commonly used standard NFFL fuel models [42] includes only 13 different fuel models) and low accuracy, therefore increasing the probability of fuel models failing to accurately depict the average characteristics of existing fuels.
(2) Furthermore, the fire dynamics are known to induce local changes in the fuel characteristics, as well as wind speed and direction, in the close vicinity of the fire front [43–45] (fuel moisture drastically decreases while wind speed increases). To some extent, such changes are intrinsic to the semi-empirical Rothermel model. However, local variations in such parameters should be expected.
(3) These four input parameters are the ones that have the most influence on the final result (fire spread rate), so their small variations are highly significant [15,46].

For the parameters concerning the fuels, a specific search space was defined as the boundaries of the fuel class, assuming that fuel classes are well identified. For instance, grass-dominated fuels can be short grass (NFFL model 1), grass understory (NFFL model 2) or tall grass (NFFL model 3), each with their own parameters. The boundaries of the parameters for the grass-dominated fuel class were defined as the search space, in case the cell fuel is any of these three models. Concerning the midflame wind speed, we considered the search space to be within the interval $\pm 25\%$ of the dataset value, which is an average of the wind speed recordings during the fire drill, obtained with a weather station installed on-site.

In this way, an individual n ($n = 1, \ldots, N$) is represented by the chromosome presented in Figure 4, where σ^n, δ^n, M_f^n, U^n are the input parameters σ, δ, M_f, U present on individual n, respectively.

σ^n	δ^n	M_f^n	U^n

Figure 4. Illustration of a chromosome.

To evaluate each n individual, the fitness function

$$R^n_{Error} = \frac{|R(\sigma^n, \delta^n, M^n_f, U^n) - R_{obs}|}{R_{obs}} \tag{25}$$

was defined. The fitness function (25) consists on the relative error between $R^n(\sigma^n, \delta^n, M^n_f, U^n)$, the rate of spread given by the Rothermel model (1) using the input parameters given by the individual n, and a real observed rate of spread value R_{obs}. The goal of GA is to minimize the fitness function.

The GA operators were chosen as follows.

- The selection operator is the tournament selection [47], which consists of randomly selecting a certain number of individuals of the current population, creating a tournament. The winner of the tournament is the individual with the best fitness and it is selected to be a parent for the next generation. This process is repeated a second time, and a pair of parent individuals is obtained.
- The crossover operator is the single point crossover technique [47]. It is executed on the parent pair by cutting the two chromosomes at corresponding points and exchanging the sections after the cuts. This generates a new offspring pair.
- The mutation operator is the uniform operator [48]. This operator consists of altering the value of a random gene in the offspring by a uniform random value which fits the gene's respective search space, at a given probability of mutation mut_{prob}, a parameter defined at the beginning of the GA implementation.
- The elitism is applied to the whole new population, i.e., a small percentage of the best individuals (*elitism*) of the previous generation replaces random individuals in the new population [48].

The new population is evaluated at each generation g ($g = 1, \ldots, g_{max}$) and the whole cycle is repeated until the maximum number of generations g_{max} is reached. After the algorithm finishes, the final solution is the individual with the best fitness from the final population. This individual is the one that, when used as input for the Rothermel model (1), results in the closest rate of spread value to the real measured value provided from the experimental data. The used algorithm is represented in Algorithm 2.

Algorithm 2 Genetic algorithm for wildfire spread calibration.

Input:
Range (minimum and maximum values), of the input parameters to be calibrated: σ_{min} and σ_{max}, δ_{min} and δ_{max}, $M_{f_{min}}$ and $M_{f_{max}}$, U_{min} and U_{max};
GA's parameters: N, g_{max}, $tour_{length}$, $cross_{prob}$, mut_{prob}, and $elitism$
Experimental dataset, includes the predefined Rothermel input parameters values and R_{obs}.
Output: Calibrated Rothermel model.
$g \leftarrow 1$
Generate initial population $P(g)$.
while $g \leq g_{max}$ **do**
  For all individuals n ($n = 1, \ldots, N$), evaluate the population $P(g)$ using R^n_{Error} (25).
  repeat
    Select pair of parents for $P(g+1)$ from $P(g)$ using Tournament Selection operator.
    Generate pair of offspring by applying Crossover operator (single point crossover).
    Obtain mutated offspring pair by applying Mutation operator (uniform mutation).
  until New population $P(g+1)$ of N individuals is obtained
  Perform Elitism on $P(g+1)$.
  $g \leftarrow g + 1$.
end while

4. Results

This section presents the validation and results of the calibration of the Rothermel model (1) using the Algorithm 2 on real datasets obtained through experimental prescribed fires in controlled conditions.

The datasets used for the calibration carried out in this work were obtained through experimental prescribed fires in controlled conditions—each dataset corresponds to a different controlled fire. There were 37 valid datasets, each one being a vector composed of the constant values for the fixed input parameters (w_0, ρ_p, S_T, M_f, M_x, S_e, h, U, ϕ) (1), observed delta (δ_{obs}) and observed oven-dry fuel load ($w_{0_{obs}}$), and the measured values for the experimental rate of spread R_{obs}. According to Algorithm 2, four input parameters are calibrated: σ (surface-area-to-volume ratio), δ (fuel bed depth), M_f (fuel moisture), and U (midflame wind speed). The remaining input parameters of Rothermel model (1) have fixed values which are the ones provided by the datasets. Despite M_f (fuel moisture) and U (midflame wind speed) being parameters that are calibrated in this paper, they are provided on the dataset, M'_f and U', respectively, based on the initial experimental conditions. However, these parameters can vary significantly during the fire itself, making it difficult for a single constant value to represent the real conditions. For each input parameter to be calibrated, there is a specific search space, i.e., a range of values that its respective gene could assume, according to the experts:

- $\sigma^n \in [43, 80]$ [cm^{-1}];
- $\delta \in [0.25, 1.2]$ [m];
- $M_f \in [0.8 \times M'_f, 1.2 \times M'_f]$ [%];
- $U \in [0.75 \times U', 1.25 \times U']$ [m/s].

The Algorithm 2 was configured in the following way: population size $N = 200$ which were evolved for $g_{max} = 100$ generations; tournament selection length $tour_{length} = 3$; crossover probability $cross_{prob} = 0.7$, mutation probability $mut_{prob} = 0.3$; and elitism factor $elitism = 0.05$. The genetic algorithm was executed 30 times for each dataset and the final fitness R^{Final}_{Error} for each dataset consisted of the average final error of the 30 GA runs:

$$R^{Final}_{Error} = \frac{1}{30} \sum_{i=1}^{30} R^i_{Error}, \tag{26}$$

where R^i_{Error} is given by (25).

Figure 5 shows the evolution of the 30 run average of the best fitness values, throughout the 100 generations, for each of the 37 datasets.

In order to compare the calibration method (Algorithm 2) to the prediction without calibration, a rate of spread R_{ini} was obtained for each dataset by running the Rothermel model (1) without calibration, i.e., the input parameters provided by the dataset were used, except for σ, whose value was not provided in the data set. The default value used was $\sigma' = 57$ cm^{-1}, which is the default value for NFFL fuel model no. 6 [42]. For the prediction without calibration, the relative error associated with the rate of spread R_{ini} for each dataset was obtained through:

$$R^{ini}_{Error} = \frac{|R_{ini}(\sigma', \delta_{obs}, M'_f, U') - R_{obs}|}{R_{obs}}. \tag{27}$$

Figure 6 shows two relative error values for each dataset, where R^{Final}_{Error} represents the final fitness given by (26), and R^{ini}_{Error} (27) represents the relative error between R_{obs} and the rate of spread value obtained without GA calibration R_{ini}, given by Equation (27). For 29 of the 37 datasets, the best rate of spread value R_{best} obtained through GA calibration resulted in a null error. This means that, if a fire was to occur in the same conditions, the final individuals could serve as input for the Rothermel model and generate very good predictions. The mean prediction error from all of the datasets without GA calibration

is 0.9510 (95%). With GA calibration, the mean error is 0.0603 (6.03%). This shows the importance of input parameters calibration, as seen in the literature.

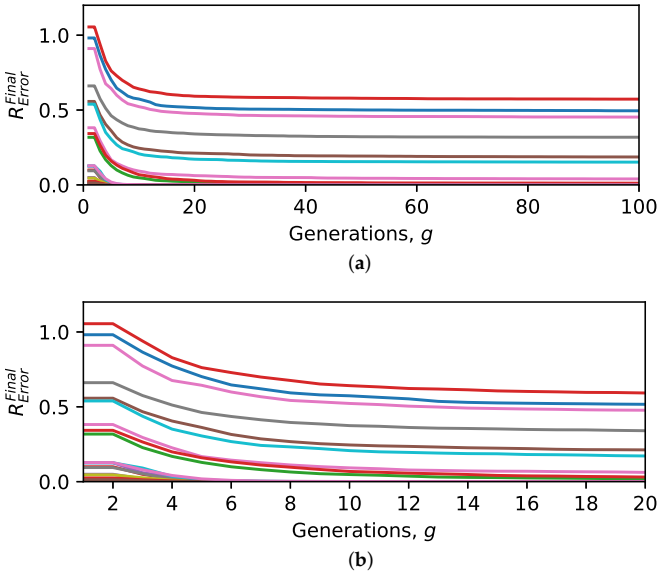


Figure 5. Evolution of the 30-run average of the best fitness values for every calibrated dataset. (**a**) Evolution of the 30-run average of the best fitness values for 100 generations. (**b**) Evolution of the 30-run average of the best fitness values for 20 generations.

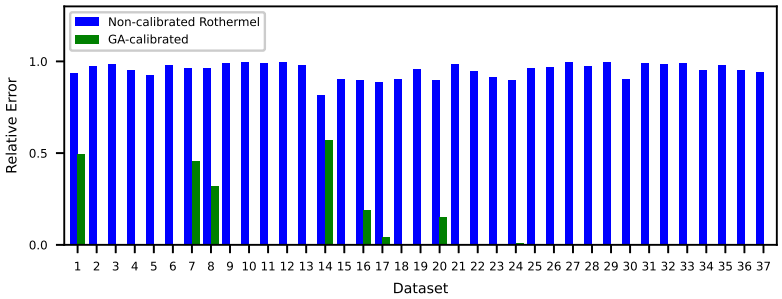


Figure 6. Relative error between the predicted and observed rate of spread R for non calibrated vs. calibrated input parameters.

5. Conclusions

Due to the physical complexity of wildfires, their prediction models require the definition of several input parameters. However, some of them are very difficult to obtain accurately or, due to their nature, present significant variations over a short period of time, due to weather or fire-driven dynamics (e.g., fuel and wind properties). Therefore, the use of optimization methodologies—specifically, genetic algorithms—to calibrate the model and to overcome input parameter uncertainty has shown to be a valid strategy to obtain accurate prediction results. This strategy will pave the way to improved fire spread simulators, capable of adapting to the particular and constantly evolving conditions of

each location, producing vital data for the decision makers and potentially mitigating the impact of wildfires.

In this work, a literature review of research works on fire spread prediction using genetic algorithms was presented, showing that genetic algorithms are the most well-accepted methodology for this application, being well-suited techniques for Rothermel model calibration. More recently, some works focused on coupling genetic algorithms with other methods to improve the prediction quality. However, due to the nature of genetic algorithms and the complexity of the model, the calibration process can be very computationally demanding. Therefore, other works also explore the possibility of reducing genetic algorithms' execution time by using parallel computing and core-allocation techniques.

Furthermore, in this work, a calibration of the Rothermel model using a genetic algorithm implementation was carried out on real datasets. The calibration was performed on four input parameters: σ (surface-area-to-volume ratio), δ (fuel bed depth), M_f (fuel moisture) and U (midflame wind speed). The results of the fire spread prediction using the calibrated model were compared to the fire spread prediction without calibration. The results showed that calibration improves prediction quality by 93.66%.

As future work, based on the literature review, we intend to extend the prediction to the domain of a two-dimensional grid in order to improve the model's applicability to real fire situations, where cells represent a squared area of the terrain through which fire propagates. This will result in the prediction of real fire behavior in the form of a map of burned cells over time. Furthermore, the parallel implementation of a genetic algorithm for the calibration of the two-dimensional Rothermel model based on the two-stage framework should be considered, which is validated by the review performed in this paper. Lastly, the framework should be tested and applied on data obtained through prescribed fires.

Author Contributions: Conceptualization, methodology, formal analysis: J.P. and J.M.; writing—original draft preparation: J.P., software, J.P. and J.S.S.J.; validation, writing—review and editing, J.M., C.V. and J.R.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research was carried out under the project IMFire–Intelligent Management of Wildfires, ref. PCIF/SSI/0151/2018, and was fully funded by national funds through the Ministry of Science, Technology and Higher Education.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. San-Miguel-Ayanz, J.; Durrant, T.; Boca, R.; Maianti, P.; Libertà, G.; Vivancos, T.A.; Oom, D.J.F.; Branco, A.; Rigo, D.D.; Ferrari, D.; et al. *Forest Fires in Europe, Middle East and North Africa 2019*; Publications Office of the European Union: Luxembourg, 2019.
2. Júnior, J.S.S.; Paulo, J.; Mendes, J.; Alves, D.; Ribeiro, L.M. Automatic Calibration of Forest Fire Weather Index for Independent Customizable Regions Based on Historical Records. In Proceedings of the 2020 IEEE Third International Conference on Artificial Intelligence and Knowledge Engineering (AIKE), Laguna Hills, CA, USA, 9–13 December 2020; pp. 1–8.
3. Júnior, J.S.; Paulo, J.R.; Mendes, J.; Alves, D.; Ribeiro, L.M.; Viegas, C. Automatic forest fire danger rating calibration: Exploring clustering techniques for regionally customizable fire danger classification. *Expert Syst. Appl.* **2022**, *193*, 116380. [CrossRef]
4. Robinne, F.N. *Impacts of Disasters on Forests, in Particular Forest Fires*; Technical Report; United Nations Forum on Forests Secretariat: New York, NY, USA, 2021.
5. Pastor, E.; Zárate, L.; Planas, E.; Arnaldos, J. Mathematical models and calculation systems for the study of wildland fire behaviour. *Prog. Energy Combust. Sci.* **2003**, *29*, 139–153. [CrossRef]
6. Rothermel, R.C. *A Mathematical Model for Predicting Fire Spread in Wildland Fuels*; Forest Service, United States Department of Agriculture: Ogden, UT, USA, 1972.
7. Sullivan, A. Wildland surface fire spread modelling, 1990–2007. 2: Empirical and quasi-empirical models. *Int. J. Wildland Fire* **2009**, *18*, 369–386. [CrossRef]
8. Finney, M.A. *FARSITE: Fire Area Simulator-Model Development and Evaluation*; Technical Report; U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station: Ogden, UT, USA, 1998.
9. Lopes, A.G.; Cruz, M.; Viegas, D. FireStation—An integrated software system for the numerical simulation of fire spread on complex topography. *Environ. Model. Softw.* **2002**, *17*, 269–285. [CrossRef]
10. Brun, C.; Artés, T.; Margalef, T.; Cortés, A. Coupling Wind Dynamics into a DDDAS Forest Fire Propagation Prediction System. In Proceedings of the 12th International Conference on Computational Science, Salvador, Bahia, Brazil, 18–21 June 2012; Volume 9, pp. 1110–1118.

11. Jain, P.; Coogan, S.; Subramanian, S.G.; Crowley, M.; Taylor, S.; Flannigan, M. A review of machine learning applications in wildfire science and management. *Environ. Rev.* **2020**, *28*, 478–505. [CrossRef]
12. Mendes, J.; Seco, R.; Araújo, R. Automatic Extraction of the Fuzzy Control System for Industrial Processes. In Proceedings of the 16th IEEE International Conference on Emerging Technologies and Factory Automation, Toulouse, France, 5–11 September 2011; pp. 1–8.
13. Mendes, J.; Araújo, R.; Souza, F. Adaptive Fuzzy Identification and Predictive Control for Industrial Processes. *Expert Syst. Appl.* **2013**, *40*, 6964–6975. [CrossRef]
14. Mendes, J.; Araújo, R.; Matias, T.; Seco, R.; Belchior, C. Evolutionary Learning of a Fuzzy Controller for Industrial Processes. In Proceedings of the 40th Annual Conference of the IEEE Industrial Electronics Society (IECON 2014), Dallas, TX, USA, 29 October–1 November 2014; IEEE: Dallas, TX, USA, 2014; pp. 139–145.
15. Andrews, P.L. *The Rothermel Surface Fire Spread Model and Associated Developments: A Comprehensive Explanation*; Technical Report; Rocky Mountain Research Station, Forest Service: Fort Collins, CO, USA; United States Department of Agriculture: Fort Collins, CO, USA, 2018.
16. Denham, M.; Cortés, A.; Margalef, T.; Luque, E. Applying a Dynamic Data Driven Genetic Algorithm to Improve Forest Fire Spread Prediction. In Proceedings of the 8th International Conference on Computational Science, Kraków, Poland, 23–25 June 2008; Volume 5103, pp. 36–45.
17. Rodríguez, R.; Cortés, A.; Margalef, T. Injecting Dynamic Real-Time Data into a DDDAS for Forest Fire Behavior Prediction. In Proceedings of the 9th International Conference on Computational Science, Baton Rouge, LA, USA, 25–27 May 2009; Springer: Berlin/Heidelberg, Germany, 2009, pp. 489–499.
18. Chelli, S.; Maponi, P.; Campetella, G.; Monteverde, P.; Foglia, M.; Paris, E.; Lolis, A.; Panagopoulos, T. Adaptation of the Canadian Fire Weather Index to Mediterranean forests. *Nat. Hazards* **2014**, *75*, 1795–1810. [CrossRef]
19. Abdalhaq, B.; Cortés, A.; Margalef, T.; Luque, E.; Viegas, D.X. Optimization of Parameters in Forest Fire Propagation Models. In *Forest Fire Research Wildland Fire Safety*; Springer: Berlin/Heidelberg, Germany, 2002; pp. 1–13.
20. Abdalhaq, B.; Cortés, A.; Margalef, T.; Luque, E. Enhancing wildland fire prediction on cluster systems applying evolutionary optimization techniques. *Future Gener. Comput. Syst.* **2005**, *21*, 61–67. [CrossRef]
21. Artès, T.; Cardil, A.; Cortés, A.; Margalef, T.; Molina, D.; Pelegrín, L.; Ramírez, J. Forest Fire Propagation Prediction Based on Overlapping DDDAS Forecasts. In Proceedings of the 15th International Conference on Computational Science, Banff, AB, Canada, 22–25 June 2015; Volume 51, pp. 1623–1632.
22. Denham, M.; Cortés, A.; Margalef, T. Computational Steering Strategy to Calibrate Input Variables in a Dynamic Data Driven Genetic Algorithm for Forest Fire Spread Prediction. In Proceedings of the 9th International Conference on Computational Science, Baton Rouge, LA, USA, 25–27 May 2009; Springer: Berlin/Heidelberg, Germany, 2009; Volume 5545, pp. 479–488.
23. Bianchini, G.; Cortés, A.; Margalef, T.; Luque, E. S2F2M—Statistical System for Forest Fire Management. In Proceedings of the 5th International Conference on Computational Science, Atlanta, GA, USA, 22–25 May 2005; Springer: Berlin/Heidelberg, Germany, 2005; pp. 427–434.
24. Denham, M.; Wendt, K.; Bianchini, G.; Cortés, A.; Margalef, T. Dynamic Data-Driven Genetic Algorithm for forest fire spread prediction. *J. Comput. Sci.* **2012**, *3*, 398–404. [CrossRef]
25. Group, W.W. *Weather Research and Forecasting (WRF) Model*; Technical Report, Director (INT-115); UCAR: Boulder, CO, USA, 2015.
26. Ascoli, D.; Bovio, G.; Vacchiano, G., Calibrating Rothermel's fuel models by genetic algorithms. In *Advances in Forest Fire Research*; Imprensa da Universidade de Coimbra: Coimbra, Portugal, 2014; pp. 102–106.
27. Sneeuwjagt, R.; Frandsen, W. Behavior of experimental grass fires vs. predictions based on Rothermel's fire model. *Can. J. For. Res.* **1977**, *7*, 357–367. [CrossRef]
28. Wilgen, B.V.; Maitre, D.L.; Kruger, F. Fire behaviour in South African fynbos (macchia) vegetation and predictions from Rothermel's fire model. *J. Appl. Ecol.* **1985**, *22*, 207–216. [CrossRef]
29. Sandberg, D.; Riccardi, C.; Schaaf, M. Reformulation of Rothermel's wildland fire behaviour model for heterogeneous fuelbeds. *Can. J. For. Res.* **2007**, *37*, 2438–2455. [CrossRef]
30. Ramirez, J.; Monedero, S.; Buckley, D. New approaches in fire simulations analysis with Wildfire Analyst. In Proceedings of the 5th International Wildland Fire Conference, Sun City, South Africa, 9–13 May 2011.
31. Artés, T.; Cortés, A.; Margalef, T. Large Forest Fire Spread Prediction: Data and Computational Science. In Proceedings of the 12th International Conference on Computational Science, Beijing, China, 4–7 July 2016; Volume 80, pp. 909–918.
32. Cencerrado, A.; Cortés, A.; Margalef, T. Genetic Algorithm Characterization for the Quality Assessment of Forest Fire Spread Prediction. In Proceedings of the 12th International Conference on Computational Science, Salvador, Brazil, 18–21 June 2012; Volume 9, pp. 312–320.
33. Fraga, E.; Cortés, A.; Cencerrado, A.; Hernández, P.; Margalef, T. Early Adaptive Evaluation Scheme for Data-Driven Calibration in Forest Fire Spread Prediction. In Proceedings of the 20th International Conference on Computational Science, Amsterdam, The Netherlands, 3–5 June 2020; Springer: Berlin/Heidelberg, Germany, 2020; Volume 12142, pp. 17–30.
34. Artés, T.; Cencerrado, A.; Cortés, A.; Margalef, T. Relieving the Effects of Uncertainty in Forest Fire Spread Prediction by Hybrid MPI-OpenMP Parallel Strategies. In Proceedings of the 13th International Conference on Computational Science, Ho Chi Minh City, Vietnam, 24–27 June 2013; Volume 18, pp. 2278–2287.

35. Cencerrado, A.; Cortés, A.; Margalef, T. On the Way of Applying Urgent Computing Solutions to Forest Fire Propagation Prediction. In Proceedings of the 12th International Conference on Computational Science, Salvador, Bahia, Brazil, 18–21 June 2012; Elsevier: Amsterdam, The Netherlands, 2012; Volume 9, pp. 1657–1666.
36. Cencerrado, A.; Artés, T.; Cortés, A.; Margalef, T. Relieving Uncertainty in Forest Fire Spread Prediction by Exploiting Multicore Architectures. In Proceedings of the 15th International Conference on Computational Science, Banff, AB, Canada, 22–25 June 2015; Volume 51, pp. 1752–1761.
37. Artés, T.; Cencerrado, A.; Cortés, A.; Margalef, T. Time aware genetic algorithm for forest fire propagation prediction: Exploiting multi-core platforms. *Concurr. Comput. Pract. Exp.* **2016**, *29*, 1–18. [CrossRef]
38. Ascoli, D.; Lonati, M.; Marzano, R.; Bovio, G.; Cavallero, A.; Lombardi, G. Prescribed burning and browsing to control tree encroachment in southern European heathlands. *For. Ecol. Manag.* **2013**, *289*, 69–77. [CrossRef]
39. Vacchiano, G.; Motta, R.; Bovio, G.; Ascoli, D. Calibrating and Testing the Forest Vegetation Simulator to Simulate Tree Encroachment and Control Measures for Heathland Restoration in Southern Europe. *For. Sci.* **2014**, *60*, 241–252. [CrossRef]
40. Mendes, J.; Souza, F.; Araújo, R.; Gonçalves, N. Genetic Fuzzy System for Data-Driven Soft Sensors Design. *Appl. Soft Comput.* **2012**, *12*, 3237–3245. [CrossRef]
41. Holland, J.H. *Adaptation in Natural and Artificial Systems*; University of Michigan Press: Ann Arbor, MI, USA, 1975.
42. Anderson, H. *Aids to Determining Fuel Models for Estimating Fire Behavior*; Technical Report; US Department of Agriculture, Forest Service, Intermountain Forest and Range: Fort Collins, CO, USA, 1982.
43. Lopes, S.; Viegas, D.X.; de Lemos, L.T.; Viegas, M.T. Equilibrium moisture content and timelag of dead Pinus pinaster needles. *Int. J. Wildland Fire* **2014**, *23*, 721–732. [CrossRef]
44. Rossa, C.G. A generic fuel moisture content attenuation factor for fire spread rate empirical models. *For. Syst.* **2018**, *27*, 1–8. [CrossRef]
45. Viegas, D.X.F.C.; Raposo, J.R.N.; Ribeiro, C.F.M.; Reis, L.C.D.; Abouali, A.; Viegas, C.X.P. On the non-monotonic behaviour of fire spread. *Int. J. Wildland Fire* **2021**, *30*, 702–719. [CrossRef]
46. Fernandes, P.A.M. Fire spread prediction in shrub fuels in Portugal. *For. Ecol. Manag.* **2001**, *144*, 67–74. [CrossRef]
47. Sivanandam, S.; Deepa, S.N. *Introduction to Genetic Algorithms*; Springer: Berlin/Heidelberg, Germany, 2008.
48. Gaspar-Cunha, A.; Takahashi, R.; Antunes, C.H. *Manual de Computação Evolutiva e Metaheurística*; Imprensa da Universidade de Coimbra, Coimbra University Press: Coimbra, Portugal, 2012.

Appendix B

Paper accepted on the IECON2022 conference

Wildfire Spread Prediction Model Calibration Using Metaheuristic Algorithms

Jorge Pereira*, Jérôme Mendes*, Jorge S. S. Júnior*, Carlos Viegas† and João Ruivo Paulo*
* University of Coimbra, Institute of Systems and Robotics,
Department of Electrical and Computer Engineering, Pólo II, PT-3030-290 Coimbra, Portugal
Emails: {jorge.pereira, jermendes, jorge.silveira, jpaulo}@isr.uc.pt
† Association for the Development of Industrial Aerodynamics,
University of Coimbra, PT-3030-289 Coimbra, Portugal. Email: carlos.viegas@uc.pt

***Abstract*—Every year, wildfires cause significant losses and destruction around the globe. In order to attempt to reduce their damages, resources have been put into developing fire propagation prediction systems. In a real wildfire event, these systems provide the authorities with information about the fire propagation in the near future, thus allowing them to make better decisions. Wildfire spread prediction systems are based on fire propagation models, from which the most used and accepted model is the Rothermel model. However, given the complexity of the wildfire phenomena and the uncertainty of some of its input parameter values, the Rothermel model can produce misleading results of fire propagation. This paper uses 3 metaheuristic algorithms, genetic algorithm (GA), differential evolution (DE) and simulated annealing (SA), for calibration of input parameters from the Rothermel model. These algorithms were validated using 37 datasets containing data from controlled experimental fires. Results have shown that these algorithms provide a precise wildfire spread prediction accounting for the uncertainties in the model's selected parameters.**

***Keywords*—wildfire spread prediction; model calibration; genetic algorithm; differential evolution; simulated annealing.**

I. Introduction

Wildfires are well-known phenomena with great environmental, economic, and societal impacts. Some of their consequences include the destruction of ecosystems and wildlife, loss of human lives, degradation of air and water quality, and the destruction of human property. According to the 2020 European Commission's annual report on wildfires, fires over 30 [ha] were observed in 39 countries throughout Europe, Middle East, and North Africa, adding up to a total burnt area of $1,075,145$ [ha] [1]. This area is approximately 35% larger than the records from 2019. Furthermore, a 2022 report published by the United Nations Environment Programme estimates that, by the end of the century, the likelihood of catastrophic wildfire events will increase by a factor of up to 1.57 [2]. These reports reveal the urgent need for allocating resources to wildfire research. There is the need for more intelligent wildfire management systems, part of which must be devoted to wildfire spread prediction, allowing to identify the areas that will be affected in advance [3].

This work was carried out under the project IMFire - Intelligent Management of Wildfires, ref. PCIF/SSI/0151/2018, fully funded by national funds through the Ministry of Science, Technology and Higher Education, Portugal.

Models developed to predict fire spread can be classified into three categories according to their nature [4]: (i) theoretical, derived from the laws of fluid mechanics and heat transfer; (ii) semiempirical, developed initially from theoretical principles and completed with experimental data; and (iii) empirical, developed based on experimental or historical fire data. The most used model for fire spread prediction is the Rothermel model [4], [5], a semiempirical model that serves as the foundation of some fire simulators [6]. Despite being widely used and effectively implemented in most fire simulators, the Rothermel model can produce inaccurate and unreliable results [7]. According to [8], there are three main sources of discrepancy between fire model spread predictions and real fire propagation: the model's lack of applicability to the scenario at hand, the model's intrinsic lack of prediction quality, and the inaccuracy in the estimation of the input parameters' values. Regardless of the fire spread model adopted, to obtain reliable predictions, one should ensure the accuracy and validity of the input parameters. This statement is particularly critical if such predictions are used by authorities in real-time during a fire occurrence to aid in their decision-making process, potentially leading to catastrophic consequences. Due to this fact, several authors stated that fire spread predictions based solely on estimations for all input parameters of the Rothermel model should not be taken confidently [7], [9], [10]. Some Rothermel model's input parameters can be easily and accurately obtained through measurement or based on existing records, such as the terrain slope. On the other hand, some parameters (e.g., fuel type, characteristics and distribution, wind speed, and direction) cannot be obtained with sufficient resolution or accuracy due to the scale of the domain and the available information sources [6].

Strategies based on Evolutionary Algorithms (EAs) have been proposed in the literature to deal with the uncertainty in the input parameter values and obtain accurate fire spread predictions [6], [7], [10], [11]. In [12], genetic algorithms (GAs) are referred to as one of the most used methods in wildfire science, with the main focus on the optimization of input parameters from fire simulators. In [11], a two-stage framework was introduced, which became a cornerstone for wildfire spread prediction calibration. This framework establishes two stages: (i) calibration of the fire propagation model

using wildfire data; and (ii) prediction of wildfire propagation using the model with the calibrated input parameters. This framework is mentioned in a large number of works in this field, using GAs as the calibration method. Some examples are [9], [13], [14]. The authors in [9] incorporate a numerical weather prediction model with the GA for dealing with the uncertainty in the model's wind parameters. In [13], the two-stage framework with the GA and Wildfire Analyst is used for model calibration. In [14], parallel implemented versions of the GA and the FARSITE fire simulator are used for improving the calibration and prediction times.

This paper explores/studies the feasibility of using three metaheuristic algorithms, genetic algorithm (GA), differential evolution (DE), and simulated annealing (SA), for the calibration of the Rothermel model. The calibrated parameters are surface-area-to-volume ratio, fuel bed depth, fuel moisture, and midflame wind speed. The main contribution of this paper is to validate two metaheuristic algorithms (DE, SA) for the calibration of the Rothermel model, in comparison with the already well-established genetic algorithms, in the subject of wildfire spread prediction calibration. The results show the potential for using differential evolution (DE) as a population-based alternative metaheuristic to genetic algorithms.

This paper proceeds in the following structure. Section II briefly describes the Rothermel model to be calibrated. Section III presents the proposed methodologies based on GA, DE and SA algorithms for the Rothermel model calibration. In Section IV, the datasets and the calibration results are described and analyzed. Concluding remarks are given in Section V.

II. Wildfire spread prediction: Rothermel model

This section presents the fire spread model to be calibrated, the Rothermel model [5]. The Rothermel model consists of a set of equations that lead to a final equation (1), which determines the fire Rate of Spread R:

$$R = \frac{I_R \xi (1 + \phi_w + \phi_s)}{\rho_b \varepsilon Q_{ig}}, \quad (1)$$

which represents the linear velocity of propagation ([m/s]) of a fire front, in a particular direction. The terms of (1), $I_R(\rho_p, \sigma,\ \delta, w_0, S_T, h, M_x, M_f, S_e), \xi(\sigma,\ \rho_p, w_0, \delta)$, $\phi_w(\rho_p, w_0, \delta, \sigma, U),\ \phi_s(\rho_p, w_0, \delta, tan\phi),\ \rho_b(w_0, \delta), \varepsilon(\sigma)$, and $Q_{ig}(M_f)$, depend on several input parameters represented in Table I. For more information on the Rothermel model, the reader is invited to read [5], [6].

III. Wildfire spread calibration

In this section, the proposed methodology for the calibration of the Rothermel model is presented. In Section III-A, the fitness function used to evaluate the solutions generated by the metaheuristic algorithms is described. Following Sections III-B, III-C and III-D present, respectively, the genetic algorithm, differential evolution algorithm and simulated annealing algorithm used for Rothermel model calibration.

The calibration performed in this work follows the two-stage framework [11]. Figure 1 shows the framework for the calibration of the Rothermel model's input parameters. Using the real fire data R_{obs} obtained in t_1, the algorithm calibrates the four input parameters σ, δ, M_f and U. Next, assuming that the fire conditions don't suffer meaningful variations from the moment that the model calibration ends until t_2, the set of calibrated parameters is used as input for the Rothermel model to predict the real fire rate of spread in t_2.

TABLE I: Rothermel model's input parameters [5].

Category	Parameter name	Units
Fuel properties	Heat content (h)	$[kJ/kg]$
	Total mineral content (S_T)	-
	Effective mineral content (S_e)	-
	Oven-dry particle density (ρ_p)	$[kg/m^3]$
	Oven-dry fuel load (w_0)	$[kg/m^2]$
	Surface-area-to-volume ratio (SAV, σ)	$[cm^{-1}]$
	Fuel bed depth (δ)	[m]
	Dead fuel moisture of extinction (M_x)	[%]
	Fuel moisture (M_f)	[%]
Topography	Slope steepness ($tan\phi$)	-
Wind properties	Midflame wind speed (U)	$[m/s]$

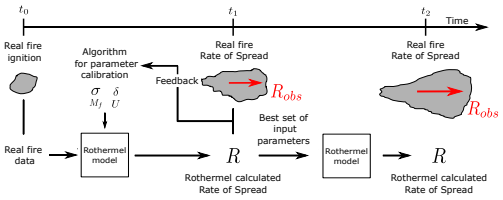


Fig. 1: Framework for the Rothermel model calibration.

A. Fitness Function

For the fire spread calibration problem described in this work, a candidate solution is represented by a set of four different parameters corresponding to the four input parameters to be calibrated: surface-area-to-volume ratio (σ), fuel bed depth (δ), fuel moisture (M_f), and midflame wind speed (U). In other words, the i-th solution (S_i) is represented by $[\sigma^i, \delta^i, M_f^i, U^i]$. The choice for calibrating these four parameters is justified as follows [6]: fuel parameters values (σ and δ) are commonly based on standard fuel models, which are not accurate in depicting the characteristics of the real fuel; additionally, the fire itself induces local changes in the fire environment near the fire front (i.e., as fuel moisture decreases, the wind speed tends to increase), meaning that the field-average wind speed value U and the fuel moisture M_f should be calibrated.

In this work, the quality of the solutions generated is evaluated by the relative error between a real observed value of rate of spread R_{obs} and the rate of spread from the Rothermel model when fed with the four input parameters values of the solution:

$$R^i_{Error}(S_i) = \frac{|R(\sigma^i, \delta^i, M_f^i, U^i) - R_{obs}|}{R_{obs}}. \quad (2)$$

B. Calibration using Genetic Algorithms

Genetic algorithms (GAs) are population-based methods that apply the principles of natural evolution to optimization problems [15]. In GA terminology, a potential solution is named individual or chromosome, and the set of potential solutions is named population.

The proposed methodology based on GA for wildfire spread calibration is represented in Algorithm 1 and is based on the GA calibration performed in [6]. In the first generation ($g = 1$), an initial population $P(g)$ of N individuals is randomly generated. Afterward, based on the evaluation by the fitness function (2), the selection operator is applied to the current population to obtain a pair of parent chromosomes. The selection operator used is the Roulette Wheel Selection, where individuals are selected from a population according to a probability that is proportional to the individual's fitness [15], [16]. Then, the crossover and mutation operators are applied to the parent pair to obtain a new pair of offspring. The crossover operator, with a probability of occurrence $cross_{prob}$, acts by separating the parent chromosomes at a corresponding crossover point (single-point crossover operator) [15]. Then, the genes after the crossover point are exchanged between chromosomes, generating two new individuals. Mutation occurs for an individual according to a probability mut_{prob} and consists of altering the value of one of the individual's genes. If the mutation operation is accepted for a given individual, one of its genes is randomly selected to change its corresponding value within a search space, performing uniform mutation [17]. This sub-process (selection, crossover and mutation) is repeated until achieving a new population of N individuals, $P(g+1)$. The elitism operator is then applied, which consists of choosing at random a small fraction ($elitism$) of the new population to be replaced with the same number of the best individuals from the previous population [17]. The new population is evaluated, and the process is repeated up to the maximum number of generations (g_{max}). The resulting final solution is the fittest individual from the last population, i.e., the chromosome with the lowest R_{Error} value.

C. Calibration using Differential Evolution

Differential evolution (DE) was first introduced in 1995 by Rainer Storn and Kenneth Price [18]. DE is similar to a GA in working by evolving a population of candidate solutions for a given problem. However, DE's search mechanism (differential mutation) is not based on any natural process.

DE initiates at iteration $t = 1$ by generating randomly an initial population with N individuals, each one containing n parameters (for this problem, $n = 4$). After this, the algorithm's main loop begins. First, a new mutant population is generated: j-th gene of the individual in the new population's position i is obtained using the differential mutation operator:

$$P'(t,i,j) = P(t,r_1,j) + F(P(t,r_2,j) - P(t,r_3,j)), \quad (3)$$

if $\gamma < C \vee j = \alpha_i$, otherwise $P'(t,i,j) = P(t,i,j)$, where $r_1, r_2, r_3 \in \{1, ..., N\}$ are random integers, F is a user-defined scale factor which "controls the rate at which the population evolves" [19], $\gamma \in [0,1]$ is a random uniform scalar, and $C \in [0,1]$ is a user-defined number that controls the fraction of parameter values copied to the new mutant solution. The differential mutation operator is applied to a given gene if $\gamma < C$, which means that the chance of applying the operator to more genes increases if C is closer to 1, with fewer parameter values copied to the new mutant solution. $\alpha_i \in \{1, ..., n\}$ is a random uniform integer to guarantee that at least one solution parameter is altered in the mutant solution.

Afterward, the current and the new populations are compared: if the i-th individual from the new mutant population $P'(t,i)$ is less fit than the corresponding individual from the current population $P(t,i)$, then the new individual is replaced by the current population's i individual. Finally, the main loop's stopping criterion is verified: if there isn't an improvement in the best fitness from the current population, $R^b_{Error}(P(t,b))$, to the new population $R^b_{Error}(P(t+1,b))$, then the variable $count$ is increased by one unity. The main loop stops if $count = count_{max}$ or t reaches the maximum number of iterations. As in the GA, the final solution from the DE is the fittest individual from the last iteration's population. Algorithm 2 describes the proposed methodology for wildfire spread calibration based on DE.

Algorithm 1 Wildfire spread calibration based on GA.

Input:

Limits of the input parameters to be calibrated: σ_{min} and σ_{max}, δ_{min} and δ_{max}, $M_{f_{min}}$ and $M_{f_{max}}$, U_{min} and U_{max}; Experimental dataset, i.e. Rothermel input parameters values and R_{obs}.

GA's parameters: N, g_{max}, $cross_{prob}$, mut_{prob}, and $elitism$;

Output: Calibrated Rothermel model.

```
g ← 1
Randomly generate the initial population P(g).
while g ≤ g_max do
    Evaluate all individuals using R^n_Error (2).
    repeat
        Select a pair of parents using Roulette Wheel Selection operator.
        Generate a pair of offspring by applying Crossover operator (single-point crossover).
        Obtain the mutated offspring pair by applying Mutation operator (uniform mutation).
    until Obtain new population P(g+1) of N individuals
    Perform Elitism on P(g + 1).
    g ← g + 1.
end while
```

D. Calibration using Simulated Annealing

Simulated annealing (SA) is a metaheuristic introduced in 1983 by Scott Kirkpatrick [20] and it is based on annealing, i.e., the process of heating a material and then slowly cooling it to obtain minimal energy states. As opposed to GA and DE, simulated annealing is not population-based.

Algorithm 2 Wildfire spread calibration based on DE.

Input:

Limits of the input parameters to be calibrated: σ_{min} and σ_{max}, δ_{min} and δ_{max}, $M_{f_{min}}$ and $M_{f_{max}}$, U_{min} and U_{max}; Experimental dataset, i.e. Rothermel input parameters values and R_{obs}.

DE's parameters: N, n, C, F, t_{max}, and $count_{max}$.

Output: Calibrated Rothermel model.

$t \leftarrow 1$, $count \leftarrow 0$

Randomly generate initial population $P(t)$.

while $t < t_{max}$ **and** $count < count_{max}$ **do**

  for $i = 1, \ldots, N$ **do**

    Randomly generate $r_1, r_2, r_3 \in \{1, \ldots, N\}$.

    Randomly generate $\alpha_i \in \{1, \ldots, n\}$.

    for $j = 1, \ldots, n$ **do**

      Generate uniform random number $\gamma \in [0, 1]$.

      if $\gamma < C$ **or** $j = \alpha_i$ **then**

        Obtain new gene in the position j, $P'(t, i, j)$, through differential mutation (3).

      else

        $P'(t, i, j) = P(t, i, j)$

      end if

    end for

  end for

  for $i = 1, \ldots, N$ **do**

    Using (2) obtain the fitness values of $P(t, i)$, $R^i_{Error}(P(t, i))$, and $P'(t, i)$, $R^i_{Error}(P'(t, i))$.

    if $R^i_{Error}(P'(t, i)) \leq R^i_{Error}(P(t, i))$ **then**

      $P(t + 1, i) \leftarrow P'(t, i)$

    else

      $P(t + 1, i) \leftarrow P(t, i)$

    end if

  end for

  if $R^b_{Error}(P(t + 1, b)) \geq R^b_{Error}(P(t, b))$ **then**

    $count \leftarrow count + 1$

  else

    $count = 0$

  end if

  $t \leftarrow t + 1$

end while

The algorithm initiates by generating an initial solution, S_i. Then, S_i is evaluated using the defined fitness function (2) $R_{Error}(S_i)$ and set to the current solution, S_c. Furthermore, the temperature T is set to an initial value T_i, starting the main loop that lasts until the temperature reaches a final value T_f. For each value of T, the following process is repeated t_{max} times: n_s neighboring solutions are generated from the current solution S_c by randomly selecting one of its elements and replacing its value by a new random value that fits the respective parameter range. Afterward, the n_s neighboring solutions are evaluated. The best of these new n_s solutions (with the lowest R_{Error}) is selected and set to S_{new}. The process of generating neighboring solutions and selecting the fittest is based on greedy search [21]. If this new solution S_{new} is fitter than S_c or if a randomly chosen uniform number $\epsilon_{[0,1)}$ is smaller than the probability of acceptance $exp((R_{Error}(S_c) - R_{Error}(S_{new})/T)$, then S_c is replaced by S_{new}. In this paper, the temperature T is updated by being multiplied by the cooling factor c_f: $T \leftarrow T \times c_f$. When the condition $T \leq T_f$ is verified, the algorithm is ceased and the current solution S_c is considered to be the best and final solution. Algorithm 3 describes the proposed methodology for wildfire spread calibration based on SA.

Algorithm 3 Wildfire spread calibration based on SA.

Input:

Limits of the input parameters to be calibrated: σ_{min} and σ_{max}, δ_{min} and δ_{max}, $M_{f_{min}}$ and $M_{f_{max}}$, U_{min} and U_{max}; Experimental dataset, i.e. Rothermel input parameters values and R_{obs}.

SA's parameters: T_i, T_f, c_f, t_{max}, and n_s.

Output: Calibrated Rothermel model.

Randomly generate initial solution S_i.

Using (2), evaluate initial solution $R_{Error}(S_i)$ and set current solution to the initial solution: $S_c \leftarrow S_i$.

$T \leftarrow T_i$.

while $T > T_f$ **do**

  for $t = 1, \ldots, t_{max}$ **do**

    Generate n_s solutions by disturbing the current solution.

    Using (2), evaluate the n_s neighboring solutions and select the one best one, assigning it as S_{new}.

    if $[R_{Error}(S_{new}) < R_{Error}(S_c)]$ **or** $[\epsilon_{[0,1)} < exp(\frac{R_{Error}(S_c) - R_{Error}(S_{new})}{T})]$ **then**

      $S_c \leftarrow S_{new}$

    end if

  end for

  $T \leftarrow T \times c_f$.

end while

IV. Results

In this section, the results of the proposed methodologies in Algorithms 1–3 for the calibration of the input parameters of the Rothermel model are presented and discussed. Section IV-A describes the datasets used for calibration. Section IV-B presents and discusses the results of the calibration.

A. Datasets

For this work, 37 datasets were used. Each dataset contains information from a different prescribed fire, which occurred in the center region of Portugal in the last five years, under various locations, different fuels, and weather conditions. Each dataset contains values for the Rothermel model's input parameters (1) according to the type of fuel burned, observed conditions (w_0, ρ_p, S_T, M_f, M_x, S_e, h, U, ϕ), measured values for w_0 ($w_{0_{obs}}$), δ (δ_{obs}), and fire rate of spread R_{obs}.

As previously stated, the only Rothermel input parameters to be calibrated are σ (surface-area-to-volume ratio), δ (fuel

TABLE II: Parameter settings for GA, DE, and SA algorithms.

GA	DE	SA
$N = 300$	$N = 300$	$T_i = 1000$
$g_{max} = 150$	$n = 4$	$T_f = 0.001$
$cross_{prob} = 0.7$	$C = 0.5$	$c_f = 0.99$
$mut_{prob} = 0.3$	$F = 0.5$	$t_{max} = 2$
$elitism = 0.05$	$t_{max} = 500$	$n_s = 20$
	$count_{max} = 20$	

bed depth), M_f (fuel moisture), and U (midflame wind speed). The M_f and U values from each dataset, obtained from initial fire conditions, are defined as M'_f and U'. Despite being given, M'_f and U' are still calibrated since they may have an elevated associated error. According to the experts [6], the intervals of variation of each parameter to be calibrated are: $\sigma \in [43, 80]$ [cm^{-1}]; $\delta \in [0.25, 1.2]$ [m]; $M_f \in [0.8 \times M'_f, 1.2 \times M'_f]$ [%]; and $U \in [0.75 \times U', 1.25 \times U']$ [m/s].

B. Results Analysis

Since Algorithms 1–3 are stochastic optimization methods, they were executed 30 times for each dataset. Furthermore, the machine used for this work consisted of an AMD Ryzen 7 3700X 8-Core Processor, 3.59 GHz, with 32.0 GB RAM, running Windows 10 Pro version. The parameters of each algorithm were fixed by trial and error according to the values shown in Table II. To evaluate the algorithms on each dataset, it is defined R^{Final}_{Error} (4), which is the average of the best fitness values over 30 trials:

$$R^{Final}_{Error} = \frac{1}{30}\sum_{t=1}^{30} R^{t}_{Error}, \tag{4}$$

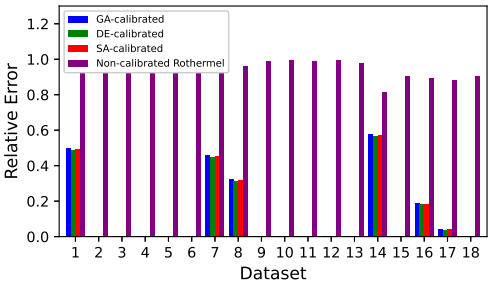
where R^{t}_{Error} is the fitness of the best solution given by (2) and provided by the t-th trial of the algorithm.

Figure 2 contains the prediction error (4) of the Rothermel model calibrated by each proposed algorithm for all datasets, and the relative error R^{NC}_{Error} (5) between the non-calibrated rate of spread R_{NC} and the observed rate of spread R_{obs}:

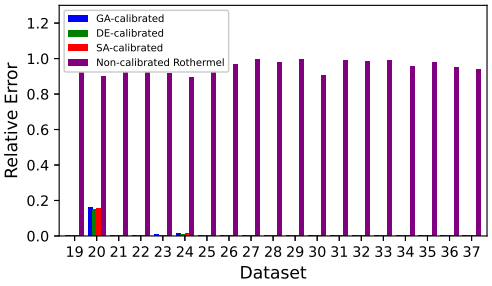
$$R^{NC}_{Error} = \frac{|R_{NC}(\sigma', \delta_{obs}, M'_f, U') - R_{obs}|}{R_{obs}}, \tag{5}$$

where R_{NC} is calculated using the observed values provided in each dataset for σ, δ, M_f and U, i.e., σ', δ_{obs}, M'_f and U'.

In Figure 2, in some datasets, only the non-calibrated model error bar is noticeable, since the proposed algorithms obtained, approximately, null relative error. Also, Figure 2 shows the significant difference between the prediction errors of the calibrated and non-calibrated models. Table III presents the average of R^{Final}_{Error} (4) of all datasets, R^{all}_{Error}, and the best fitness result from all datasets. Table III shows that the three algorithms achieved similar calibration performances. Furthermore, GA and DE had the same best fitness results, despite DE performing slightly better than GA in R^{all}_{Error}. Additionally, for some datasets (1-st, 7-th, 8-th, 14-th, 16-th, 17-th and 20-th), the three algorithms could not obtain a near-zero relative error, despite obtaining similar results. This may be due to the fact that only four input parameters are being



(a) Datasets 1 to 18.



(b) Datasets 19 to 37.

Fig. 2: Comparison of the calibration results for the three algorithms with the non-calibrated Rothermel model.

TABLE III: Calibration results of the proposed algorithms (average of all datasets).

Algorithms	R^{all}_{Error}	Best fitness	First occurrence	
			Iteration	Time (s)
GA	0.250	3.23×10^{-4}	37	3.166
DE	0.244	3.23×10^{-4}	3	0.037
SA	0.248	6.33×10^{-4}	25	0.412

calibrated, a bad suitability of the considered fuel model to the real fuel burned in those fire experiments or, simply, the model's intrinsic incapacity of accurately replicating the real fire behavior in those specific conditions.

Another important aspect in wildfire spread prediction is the calibration time [6]. The calibration of the model should be performed on time to obtain usable fire spread predictions. To evaluate the time performance of the algorithms, we consider the time and number of iterations that led to the first occurrence of the best fitness value provided by the algorithms, as shown in Table III. Figure 3 contains the iterations of the first occurrences of the best fitness values for each algorithm and each dataset. From Figure 3, it can be observed a clear pattern for the differential evolution, which takes a small number of iterations to obtain a first value of the

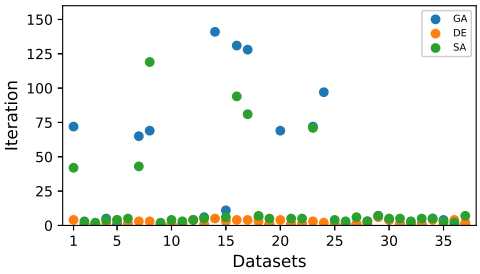


Fig. 3: Iteration of first occurrence of the best fitness value, for each algorithm and dataset.

best fitness. The number of iterations is more dispersed for the genetic algorithm and simulated annealing. Additionally, it is important to refer that for three datasets (14-th, 20-th, and 24-th), the simulated annealing algorithm ran for more than 150 iterations until the first occurrence of the best fitness value (314, 961 and 432 iterations, respectively). Consequently, these points are not shown in Figure 3 to ensure a more consistent and accurate viewing. From Table III, we verify that the differential evolution is the fastest algorithm, with an average duration of 3 iterations until the first occurrence of the best final fitness value, in comparison with 37 iterations from the GA and 25 iterations from the SA. Regarding the effective computation time required for the study, the average run times ($\bar{t}$) of each algorithm iteration were the following: $\bar{t}_{GA} = 4.75\,s$, $\bar{t}_{DE} = 0.491\,s$ and $\bar{t}_{SA} = 82.7\,s$. The overall study time was approximately $97627.61\,s$.

By summarizing the results of this work, we conclude that the three algorithms (based on GA, DE, and SA) had similar behavior in terms of calibration quality, despite the DE being better in terms of time performance.

V. Conclusions

As stated in Section I, the wildfire spread prediction area has been dominated by the use of genetic algorithms as the main tool for the calibration of the Rothermel model. However, the results obtained in this work show that differential evolution is also a very suitable algorithm for the calibration of the Rothermel model, mainly due to its time performance, which is critical in wildfire spread prediction.

As described in [6], the use of evolutionary algorithms as calibration tools of fire simulators may result in a large number of simulations, which can be time-consuming. Thus, several works perform parallel implementations of genetic algorithms and fire simulators in order to reduce computational times and obtain faster fire spread predictions [14]. Since DE can be used as a calibration algorithm for the Rothermel model, it should also be considered for parallel implementation in future works.

References

[1] J. San-Miguel-Ayanz, T. Durrant, R. Boca, P. Maianti, G. Libertà, T. Artés-Vivancos, D. Oom, A. Branco, D. de Rigo, D. Ferrari, H. Pfeiffer, R. Grecchi, D. Nuijten, M. Onida, and P. Löffler, "Forest Fires in Europe, Middle East and North Africa 2020," Tech. Rep., 2021.

[2] UNEP and GRID-Arendal, "Spreading like Wildfire: The Rising Threat of Extraordinary Landscape Fires," United Nations Environment Programme, Tech. Rep., 2022.

[3] J. S. Júnior, J. R. Paulo, J. Mendes, D. Alves, L. M. Ribeiro, and C. Viegas, "Automatic forest fire danger rating calibration: Exploring clustering techniques for regionally customizable fire danger classification," *Expert Systems with Applications*, p. 116380, 2022.

[4] E. Pastor, L. Zárate, E. Planas, and J. Arnaldos, "Mathematical models and calculation systems for the study of wildland fire behaviour," *Progress in Energy and Combustion Science*, vol. 29, no. 2, pp. 139–153, December 2003.

[5] R. C. Rothermel, *A mathematical model for predicting fire spread in wildland fuels*. Forest Service, U. S. Department of Agriculture, 1972.

[6] J. Pereira, J. Mendes, J. S. S. Júnior, C. Viegas, and J. R. Paulo, "A Review of Genetic Algorithm Approaches for Wildfire Spread Prediction Calibration," *Mathematics*, vol. 10, no. 3, 2022.

[7] B. Abdalhaq, A. Cortés, T. Margalef, E. Luque, and D. X. Viegas, "Optimization of Parameters in Forest Fire Propagation Models," *Forest Fire Research & Wildland Fire Safety*, pp. 1–13, January 2002.

[8] F. A. Albini, "Estimating wildfire behavior and effects," USDA Forest Service, Intermountain Forest and Range Experiment Station, Tech. Rep., 1976.

[9] C. Brun, T. Artés, T. Margalef, and A. Cortés, "Coupling Wind Dynamics into a DDDAS Forest Fire Propagation Prediction System," in *Proc. 12th International Conference on Computational Science*, June 2012, pp. 1110–1118.

[10] M. Denham, K. Wendt, G. Bianchini, A. Cortés, and T. Margalef, "Dynamic Data-Driven Genetic Algorithm for forest fire spread prediction," *Journal of Computational Science*, vol. 3, no. 5, pp. 398–404, September 2012.

[11] B. Abdalhaq, A. Cortés, T. Margalef, and E. Luque, "Enhancing wildland fire prediction on cluster systems applying evolutionary optimization techniques," *Future Generation Computer Systems*, vol. 21, no. 1, pp. 61–67, January 2005.

[12] P. Jain, S. Coogan, S. G. Subramanian, M. Crowley, S. Taylor, and M. Flannigan, "A review of machine learning applications in wildfire science and management," *Environmental Reviews*, vol. 28, no. 4, pp. 478–505, August 2020.

[13] T. Artès, A. Cardil, A. Cortés, T. Margalef, D. Molina, L. Pelegrín, and J. Ramírez, "Forest Fire Propagation Prediction Based on Overlapping DDDAS Forecasts," in *Proc. 15th International Conference on Computational Science*, June 2015, pp. 1623–1632.

[14] T. Artés, A. Cencerrado, A. Cortés, and T. Margalef, "Relieving the Effects of Uncertainty in Forest Fire Spread Prediction by Hybrid MPI-OpenMP Parallel Strategies," in *Proc. 13th International Conference on Computational Science*, June 2013, pp. 2278–2287.

[15] S. Sivanandam and S. N. Deepa, *Introduction to Genetic Algorithms*. Springer-Verlag Berlin Heidelberg, 2008.

[16] J. Mendes, R. Araújo, T. Matias, R. Seco, and C. Belchior, "Evolutionary Learning of a Fuzzy Controller for Industrial Processes," in *Proc. of The 40th Annual Conference of the IEEE Industrial Electronics Society (IECON 2014)*, October 29 - November 1 2014, pp. 139–145.

[17] A. Gaspar-Cunha, R. Takahashi, and C. H. Antunes, *Manual de computação evolutiva e metaheurística*. Coimbra University Press, June 2012.

[18] R. Storn and K. Price, "Differential Evolution: A Simple and Efficient Adaptive Scheme for Global Optimisation Over Continuous Spaces," *Journal of Global Optimization*, vol. 23, 1995.

[19] K. Price, R. Storn, and J. Lampinen, *Differential Evolution-A Practical Approach to Global Optimization*. Springer, 2005.

[20] S. Kirkpatrick, C. D. Gelatt, and M. Vecchi, "Optimization by Simulated Annealing," *Science*, vol. 220, pp. 671–680, 1983.

[21] S. Lee and S. B. Kim, "Parallel Simulated Annealing with a Greedy Algorithm for Bayesian Network Structure Learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 6, pp. 1157–1166, 2020.