Habitat selection of *Oryx dammah* in the Ouadi Rimé-Ouadi Achim Faunal Reserve in Chad

Julien Court

University of Neuchâtel

03 August 2025

Table of content

1. Introduction 2

1.1. About the large herbivores 2

1.2. *Oryx dammah* presentation 2

1.2.1. Physical features 2

1.2.2. Ecology and behavior of *Oryx dammah* 3

1.2.3. Conservation status and reintroduction programs 7

1.3. Conservation challenges 8

1.4. Context and aims of the study 9

1.5. Research question 10

1.6. Hypotheses about seasonal variations in the habitat selection of *Oryx dammah* 10

2. Method 11

2.1. Theory of the integrated step-selection function (ISSF) 11

2.2. Data collection and Timeframe 12

2.3. Study site 13

2.3.1. Climate and ecosystem 14

2.3.2. Demographics and culture 14

2.4. Factors analyzed 15

2.4.1. Vegetation factors 15

2.4.2. Topography 20

2.4.3. Human presence 21

2.5. Summary of oryx movements data 24

2.6. Selection of oryxes movements data 25

2.6.1. Preparation of data 25

2.6.2. Organization and filter of the data in a summary table 25
2.6.3. Filtering of the original dataset 29
2.7. Extraction of movement statistics 30
2.8. Creation of the available steps 32
2.9. Extraction of the environmental factors 35
2.9.1. Simple extraction on the non-time-dependent factors 35
2.9.2. Dynamic extraction of time-dependent indicators 36
2.10. Statistical analysis 36
2.10.1. Collinearity pair-to-pair assessment 36
2.10.2. Presentation of the models 37
2.10.3. Standardization of the data 39
2.10.4. ISSF models calculations 39
2.10.5. Performance assessment of the models 39
2.10.6. Production of Relative Selection Strength figures 40
3. Results 41
3.1. Best model 41
3.2. Movement statistics 42
3.3. Vegetation cover 44
3.4. Woody vegetation density 45
3.5. Wildfires factor 46
3.6. Topography factor 47
3.7. Human activity factor 48
4. Discussion 50
4.1. Movement statistics 50
4.2. Vegetation 51

4.1. Topography 53
4.2. Human activity 54
4.3. Methodological limitations 55
4.3.1. Irregularities between seasons in the amount of data 55
4.3.2. Udapte of turning angle 55
4.3.3. Accuracy of the indicators 56
4.3.4. Interaction assumption 56
5. Conclusion 57
6. Acknowledgements 58
7. Bibliography 59
A. Appendix 65
A.1. Extraction of binary layers of wildfires 65
A.2. Creation of the raster: sum of fires 2004 to 2023 66
A.2.1. Extraction of the yearly sum of wildfires 66
A.2.2. Sum of the yearly layers of wildfires 68
A.3. Topographic Position Index calculation code 69
A.4. Topographic Convergence Index calculation code 70
A.5. Creation of the layer of human activity 71
A.5.1. R code used for the creation of the human activity layer 71
A.5.2. Processing workflow for the transhumant paths layer 75
A.6. First operations on the dataset 76
A.7. Creation of the summary table 77
A.7.1. Addition of the date gaps 78
A.7.2. Filtering of the incompatible time intervals 81
A.7.3. Addition of seasons 81

A.7.4. Seasonal table creation 84
A.8. Filtering of the original dataset 85
A.8.1. Filtering of irrelevant bursts 85
A.8.2. Filtering of incomplete seasons 86
A.8.3. Resampling of the dataset 89
A.9. Extraction of movement statistics 90
A.10. Creation of available steps 91
A.10.1. Creation of parametric curves for the relative turning angles 91
A.10.2. Extraction of the available steps 92
A.11. Extraction on the non-time-dependent indicators 94
A.12. Dynamic extraction of time-dependent indicators 97
A.12.1. Extraction of MSAVI Landsat 97
A.12.2. Extraction of MSAVI (MODIS) 99
A.12.3. Extraction of D-MSAVI 102
A.12.4. Merging of the data by season 104
A.13. Pair-to-pair collinearity assessment 106
A.13.1. Code used for the creation of the collinearity assessment 106
A.13.2. Results of the pair-to-pair collinearity assessment 108
A.14. Standardization of the environmental variables 110
A.15. Code for the ISSFs 111
A.15.1. ISSF models with only linear factors 111
A.15.2. ISSF models with quadratic topographic factors 128
A.15.3. ISSF models with quadratic vegetation density factors 144
A.15.4. ISSF models with topographic and vegetation factors both quadratic 161
A.16. AICc analyses 178

A.16.1. Code for the AICc analyses 178
A.16.2. AICc for hot dry season models 180
A.16.3. AICc for rainy season models 182
A.16.4. AICc for cool dry season 184
A.17. Code to save the best ISSF model in CSV format 187
A.18. Code for RSS creation 187

Abstract

The aim of this study was to determine the main factors influencing the habitat selection of *Oryx dammah* in the Ouadi Rimé-Ouadi Achim Faunal Reserve in Chad. More precisely, we used the spatial positions and movements of *Oryx dammah* acquired by GPS collars between 2016 and 2024 and sought correlations with some environmental factors in the observed habitat selection, using integrated step-selection functions. The final goal of this operation is to contribute to the conservation of this threatened species, by extending our knowledge of its ecology and increasing the probability of success of the current and potential future reintroduction projects.

We used step-selection functions to determine how vegetation density, wildfires, topography and human presence affect the habitat selection of *Oryx dammah*. We conducted this study in a seasonal framework, examining the particular features of each season.

Our work shows that vegetation density is the main driver of *Oryx dammah*'s habitat selection. An indicator of woody vegetation density, in addition to overall vegetation density, increases the explanatory performance of a model. *Oryx dammah* is particularly sensitive to changes in vegetation during the hot dry season. Its presence is also positively correlated with places that suffered wildfires during previous cool dry seasons. Finally, the species is generally attracted by areas with a low Topographic Convergence Index (TCI) and a low human presence.

1. Introduction

1.1. About the large herbivores

Large herbivores are generally considered to have great importance in their ecosystems. Their selective consumption of certain plant species allows them to influence the competition between various plant species, thereby affecting the structure of the plant community (Owen-Smith, 2021). This in turn impacts biomass levels and plant demographics (Pringle et al., 2023). They can also affect wildfire distribution, enhance nutrient cycling through their excretions, and help with seed dispersal (Pringle et al., 2023; Ripple et al., 2014).

Large herbivores occupy ecological niches distinct from those of small herbivores, and together, these groups complement each other within the ecosystem (Pringle et al., 2023). Large herbivores can even help maintain favorable environmental conditions for mesoherbivores and regulate their densities (Ripple et al., 2014).

Due to their considerable size, they effectively increase the heterogeneity of the environment through activities such as trampling and others (Owen-Smith, 2021). This can result in a variety of microhabitats that can increase species richness.

In addition to all these effects that can be observed in most ecosystems, the relative density of grazing versus browsing herbivores shapes the landscape by influencing the balance between grass and trees in savannas (Owen-Smith, 2021; Staver et al., 2021).

Despite their ecological and socio-economic importance, large herbivores are a particularly threatened group (Pringle et al., 2023). Habitat destruction and overhunting are two major causes of their decline (Ripple et al., 2014). Their decline can lead to a range of severe and detrimental consequences, making their conservation a critical priority.

1.2. *Oryx dammah* presentation

1.2.1. Physical features

Oryx dammah (Figure 1) is an antelope with a white coat and a robust body, allowing it to weigh approximately 140 kg with a head-body length of 1.9-2.2 m and a height at the withers of 1.1-1.25 m (Kingdon, 2015). Males are on average slightly larger than females and have shorter but thicker horns. Their lifespan is around 17 years (Kingdon, 2015).

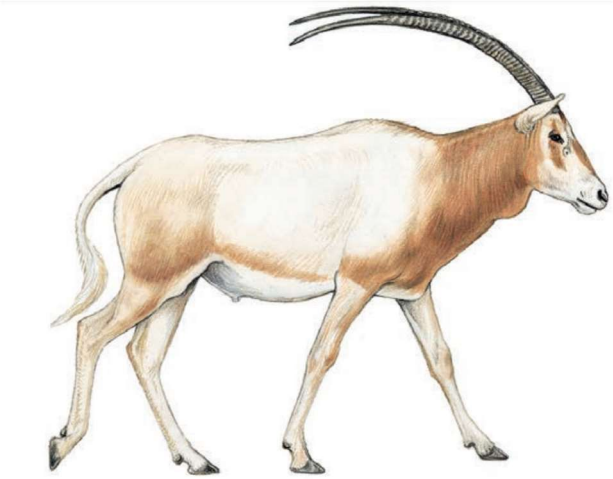

Figure 1: A drawing of Oryx dammah *(Kingdon & Hoffmann, 2013)*

1.2.2. Ecology and behavior of *Oryx dammah*

1.2.2.1. Social behavior

They live in herds of around 10-15 individuals (Kingdon & Hoffmann, 2013). They are mixed-sex, usually with at least one old male and several young and juvenile individuals (Kingdon & Hoffmann, 2013).

1.2.2.2. Range and distribution

Oryx dammah was, before its rapid decline and extinction in the wild, widespread in countries such as Egypt, Algeria, Morocco, Mauritania, Mali, Chad, Niger, and Sudan (Kingdon & Hoffmann, 2013). Today, it mainly lives in captivity in private collections and also in semi-captive conditions in reserves in Morocco, Tunisia, and Senegal (IUCN SSC Antelope Specialist Group, 2023). The only truly wild population is found in the Ouadi-Rimé Ouadi Achim Faunal Reserve (OROAFR) in Chad (Figure 2) (IUCN SSC Antelope Specialist Group, 2023).

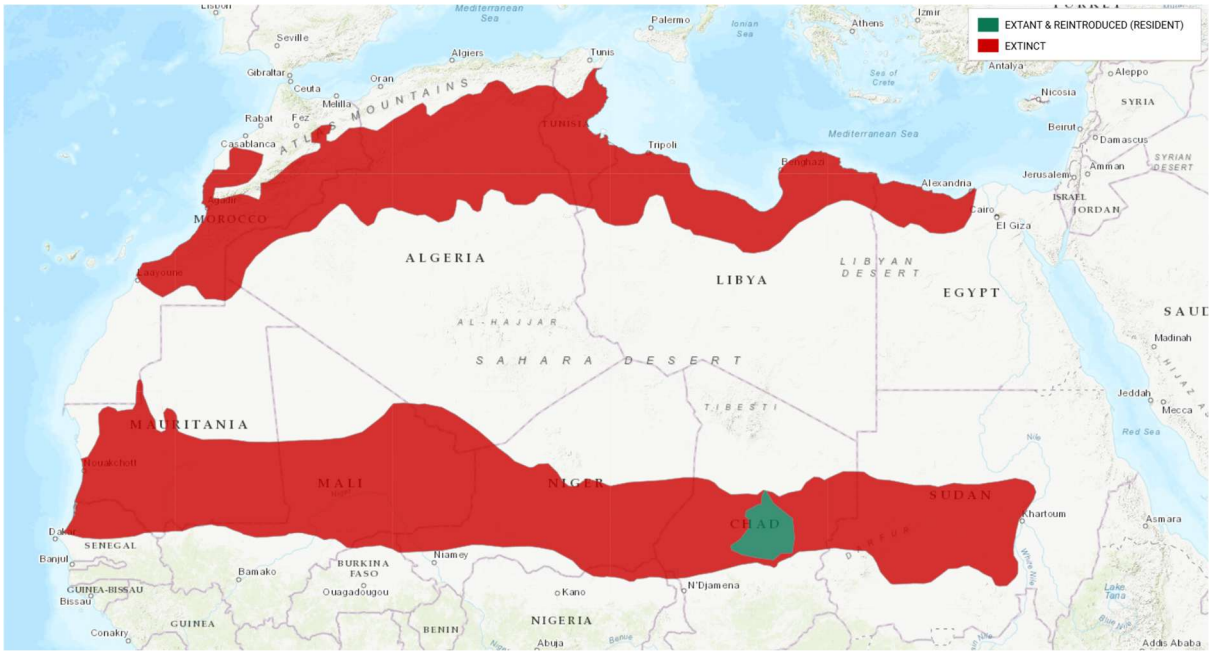

Figure 2: The historical range of Oryx dammah *in red, with its current distribution in green (Cooke, 2023).*

1.2.2.2.1. Seasonal migrations

Traditionally, it was assumed that there was just a wet season during the July-September period, with the rest of the year being a long dry season (Wacher et al., 2022). However, a recent study shows that the OROAFR experiences three climatic seasons and *Oryx dammah* adapts its behavior to the seasonal variations (Whyle et al., in prep). The "rainy" season takes place from July 11 to October 1, the "cool dry" season spans from October 2 to March 12 and the "hot dry" season lasts from March 13 to July 10 (Whyle et al., in prep). The rainy season is the season when most precipitation occurs. The rain stops during the cool dry season, but the temperature is still relatively low. The hot dry season comes after several dry months and is characterized by very high temperatures (Whyle et al., in prep).

Oryx dammah is strongly influenced by this seasonal cycle that dictates its long-distance migrations (Kingdon & Hoffmann, 2013). Its great stamina allows it to travel up to 300 km between locations used in the hot season and those used in the rainy season (Kingdon, 2015). In Chad, it has been observed to live in the south during the dry season, where more vegetation and lower temperatures can be found, while it returns to the north during the rainy season, when the temperatures decrease and the arid soil becomes more fertile (Majaliwa et al., 2022). The topographic characteristics of the habitat selected by *Oryx dammah* also change between dry and rainy seasons (see section 1.2.2.3.3 for more details).

1.2.2.3. Habitat and diet

Oryx dammah lives in Sahelian-Saharan semi-desert grassland (Kingdon & Hoffmann, 2013). Vegetation is a key feature of the habitat of this species. The vegetation cover represents a source of food, mainly through grasses (Kingdon & Hoffmann, 2013). Indeed, the diet of *Oryx dammah* consists of perennial grasses and sometimes pods, herbs, and shrubs such as *Panicum*

turgidum, *Aristida mutabilis*, *Acacia tortilis* or other herbs (Kingdon & Hoffmann, 2013).

1.2.2.3.1. Access to shade

Another significant role of vegetation, through large plants such as trees, is the creation of shaded areas, which are important in such regions with high heat stress (Lox, 2024). In particular, the woody wadis and isolated trees are identified as important shade providers for *Oryx dammah*, particularly during the dry season, when they spend most of the hottest hours under this vegetation (Figure 3) (Gillet, 1965; Majaliwa et al., 2022).

Figure 3: Oryx dammah *using a* Balanites *as their source of shade (Lox, 2024).*

1.2.2.3.2. Wildfires impact

Wildfires can be a factor influencing the environment by changing the plant species present on the ground (Figure 4) (Lox, 2024). This might affect *Oryx dammah* in a way hypothesized by Lox as follows:

> "As a result of the changes to the vegetation in the burnt sites, the scimitar-horned oryx could favor sites burnt frequently and at an early stage at the beginning of the rainy season, to graze on the abundant young shoots of *Schoenefeldia gracilis*." (Lox, 2024).

Figure 4: Oryx dammah *on a field that has suffered from wildfires. The vegetation has been removed, which will affect its plant composition until it recovers (Lox, 2024)*

1.2.2.3.3. Topographic preferences

Topography is also an important identified factor. In the Bouhedma National Park (Tunisia), Beyouli & Nefati (2016) note: "It turns out that Scimitar horned oryx population tends to use the plain of the park covering more than 2000 hectares and avoids the areas with high and medium elevation despite the presence of palatable plant, like *Stipa tenacissima* (especially in drought periods) (Caron, 2001), and *Helianthemum sp*." (Beyouli & Neffati, 2016). Quantitative observations highlighted that in the OROAFR, *Oryx dammah* selects higher areas, with low roughness, during the dry season, and lower and rougher areas during the rainy season (Majaliwa et al., 2022). We also have sources indicating that *Oryx dammah* goes more to interdunal depressions during the dry seasons to benefit from more shade (Kingdon & Hoffmann, 2013). These observations are not totally consistent, which could justify further research on the topic.

1.2.2.3.4. Cohabitation with humans

Human presence is usually a strong factor impacting the habitat selection of most animals. *Oryx dammah* is no exception. The main documented way in which humans impact the activities of *Oryx dammah* is through livestock presence (Wacher et al., 2022). Indeed, the movements and the size of the herds are affected by the presence of humans and livestock (Kingdon & Hoffmann, 2013). This is a recognized cause of the extinction of this species at that time (IUCN SSC Antelope Specialist Group, 2023).

The presence of livestock is particularly strong near water points. Some of these places do not

provide water year-round, such as hafirs, which are artificial ponds (Worgue & Yamtibaye, 2023). This reduces the presence of livestock near some of these sources during the dry months, while their presence increases during the rainy season (Wacher et al., 2022).

Pastoralists are accustomed to giving natron to their animals (Collelo, 1990). Indeed, a high salt intake is known to help herbivores reduce water stress (Razali et al., 2025). Casual field observations indicate that during the hot season, *Oryx dammah* is attracted by this artificial source of minerals (Ngaba, Caleb, 2022, pers. obs.). Even if there is a lack of evidence for the significance of this behavior, it could represent another plausible human-wildlife interaction influencing habitat selection.

1.2.2.4. Predators

Large carnivores are rarely found in the habitat of *Oryx dammah*, but the list includes a few species of hyaenas (*Crocuta crocuta* and *Hyaena hyaena*), the painted dog (*Lycaon pictus*), the golden jackal (*Canis aureus*) and cheetahs (*Acinonyx jubatus*). They mainly target young or weak individuals (Kingdon & Hoffmann, 2013).

1.2.3. Conservation status and reintroduction programs

The populations of *Oryx dammah* declined severely during the 19th and 20th centuries, due to overhunting and competition with livestock (Kingdon & Hoffmann, 2013). The species was officially declared extinct in the wild in 1988 (IUCN SSC Antelope Specialist Group, 2023).

However, *Oryx dammah* survived in captivity, due to large private collections that are estimated to account for around 15,000 individuals in 2023 (IUCN SSC Antelope Specialist Group, 2023).

Several populations have benefited from reintroduction programs over the years, but none of these meet the criteria to be considered wild populations, due to the fencing that confines them to small areas (IUCN SSC Antelope Specialist Group, 2023; Woodfine & Gilbert, 2016).

The first reintroduction program took place in 1985 in Tunisia's Bou-Hedma National Park in cooperation with British and French organizations and included 10 individuals from British zoos (Gordon & Gill, 1993; Woodfine & Gilbert, 2016).

This was followed by three other reintroductions of European-owned individuals in Tunisia, in 1999. One took place in Sidi Toui National Park and involved ten individuals. Another was in the Oued Dekouk National Reserve with 3 animals, while the last one was in Bou Hedma (Woodfine & Gilbert, 2016). In 2007, seven additional individuals were introduced in Bou Hedma, and a fifth site of reintroduction was created in Dghoumes National Park (Woodfine & Gilbert, 2016). The total population in 2023 was estimated at around 210 individuals. However, it was not expected to grow any further because it is approaching the carrying capacity of the available habitat (Petretto & Gilbert, 2024).

In Morocco, two reintroductions of *Oryx dammah* took place in 1995 and 1997, in the Arrouais

Reserve, within Souss-Massa National Park, involving 25 animals (Woodfine & Gilbert, 2016). The total population is estimated at around 300 individuals in 2023 (IUCN SSC Antelope Specialist Group, 2023). These animals are not intended to remain there permanently, but only until they can be reintroduced into a more suitable environment (Whitemore & Gilbert, 2024).

Two reintroductions also took place in Senegal. The first occurred in 1999, in the Guembeul Fauna Reserve, with eight individuals coming from the Hai Bar Reserve in Israel, and two additional individuals from the Paris Zoo added in 2001 (Gilbert, 2004; Woodfine & Gilbert, 2016). The second one was in 2003, in the Ferlo National Park and Biosphere Reserve, transferring 8 individuals from the first reserve (Gilbert, 2004; Woodfine & Gilbert, 2016).

1.2.3.1. Reintroduction program in Chad

In March 2016, for the first time since its extinction in the wild, 25 individuals were reintroduced into the wild in their natural habitat, in the Ouadi-Rimé Ouadi Achim Faunal Reserve (OROAFR) in Chad. By November 2023, more than 600 individuals were living freely in the wild, thanks to additional waves of reintroduction and reproduction (Chardonnet et al., 2024). The success of Chad's Scimitar Oryx Reintroduction Project was made possible through the collaboration and efforts of the Chadian government and many conservation organizations, including the Environmental Agency of Abu Dhabi, the Smithsonian Conservation Biology Institution, and the Sahara Conservation Fund (Meyer, 2016).

1.3. Conservation challenges

Even though *Oryx dammah* has been successfully reintroduced, many challenges still directly threaten the long-term survival of the wild population. Habitat degradation is a major concern. Desertification, a historical cause of decline, has been exacerbated by human activity (Gilbert & Woodfine, 2004). In addition to this factor, the loss of food resources and habitat due to competition with large livestock populations in the reserve is a serious human-induced problem (IUCN SSC Antelope Specialist Group, 2023). The rapid expansion of agriculture in the OROAFR may cause soil degradation and habitat fragmentation, and the use of certain agricultural practices and motorized equipment raises concerns (Worgue & Yamtibaye, 2023). Overhunting has historically contributed to their decline, and droughts have been identified as causing significant declines in *Oryx dammah*'s populations in the past (Majaliwa et al., 2022; Petretto & Gilbert, 2024). Political instability is another danger, as wars and other prolonged military conflicts are a historical cause of decline (Majaliwa et al., 2022).

The small size of its population leaves *Oryx dammah* vulnerable to inbreeding and stochastic effects, as has been observed with other threatened antelopes living in similar environments in Senegal (Moreno et al., 2012).

Generally, a reintroduction is always a high-risk conservation strategy that must be implemented with great care. The success rate is known to be relatively low, and rigorous post-release research and monitoring are essential to maximize the chances of a successful outcome (Mertes et al., 2019).

1.4. Context and aims of the study

The aim of this study is to expand our understanding of *Oryx dammah*'s relationship with its environment in the OROAFR. Indeed, most of the documentation about this species is general and not tailored to the specific conditions in this region. Moreover, quantitative data on the subject are lacking. As stated by Leimgruber et al. (2023): "The species disappeared from the wild before being systematically studied and there are only a few very limited observations that give information about the ecology and behavior of species in the wild. Thus, little is known about its habitat requirements." (Leimgruber et al., 2023).

Some studies were conducted to fill this gap, mainly in Tunisia (Beyouli & Neffati, 2016; Louhichi et al., 2024). However, the populations studied were smaller and not truly wild, as their mobility was restricted by fencing.

A study conducted in the strategic location of the OROAFR, used the powerful tool of integrated step-selection functions to perform for the first time a large-scale and systematic analysis of *Oryx dammah*'s habitat selection in the wild (Majaliwa et al., 2022).

Although the work of Majaliwa (2022) is a valuable resource, it merely opens a path in a field that demands further exploration, especially the search for alternative and potentially more effective environmental indicators to explain *Oryx dammah*'s habitat selection.

Recent observations suggest that *Oryx dammah*'s ecology may follow the rhythm of three seasons, contrary to the two traditionally considered (Whyle et al., in prep). However, the implications of this discovery remain mostly unexamined.

All these elements – considered in the context of a highly threatened species – require a rapid development of scientific knowledge regarding the behavior of *Oryx dammah*, particularly in the OROAFR, where the most important reintroduction program is taking place. In this context, our aim with this study is to explore and deepen our knowledge about *Oryx dammah*'s habitat selection, in order to improve the likelihood of success for conservation efforts targeting this species.

1.5. Research question

We aim to use the newly defined seasonal framework to determine whether it reveals novel trends in the habitat selection of *Oryx dammah* in relation to several major environmental factors that were not identified by Majaliwa (2022) and other previous studies. In addition, we seek to evaluate the relevance of additional environmental factors, in order to identify a combination of ecological indicators that can accurately describe the habitat selection of *Oryx dammah*. The question that characterizes our research is:

"How do vegetation, topography, and human presence influence the habitat selection of *Oryx dammah* through the seasonal cycle in the Ouadi Rimé-Ouadi Achim Faunal Reserve?"

1.6. Hypotheses about seasonal variations in the habitat selection of *Oryx dammah*

To be more precise about how we aim to answer our research question, there are several hypotheses that we want to test as goals for this study:

1. *Oryx dammah* selects places with higher vegetation productivity regardless of the season, as food resources are among the primary drivers of habitat selection in herbivores in general (Kingdon & Hoffmann, 2013).

2. The hot dry season is the one during which *Oryx dammah* selects places with the highest tree and shrub density, as they seek shade to endure the high temperatures (Gillet, 1965; Majaliwa et al., 2022).

3. *Oryx dammah* presence will be positively correlated with the areas that suffered wildfire previously, as vegetation that regenerates in such places corresponds to plants that are appreciated as food by *Oryx dammah* (Lox, 2024).

4. The rainy season is the one during which *Oryx dammah* selects the habitat with the highest terrain irregularity, while the hot dry season is associated with areas with lower terrain irregularity, confirming the previous observations made in that environment (Majaliwa et al., 2022).

5. *Oryx dammah*, in general, tends to avoid human presence. However, it may select habitats closer to places associated with intense human activity during the hot dry season compared to other seasons, as they seek the salt commonly deposited around these places or because of lower human activity during this period (Wacher et al., 2022).

2. Method

2.1. Theory of the integrated step-selection function (ISSF)

To study the factors influencing the habitat selection of *Oryx dammah*, we use the same analytical tool as in Majaliwa (2022): the integrated step selection function (ISSF) (Majaliwa et al., 2022). It is an analytical approach that uses small-scale movement data from a population to identify its ecological preferences (Florko et al., 2025).

A step is the line that can be drawn by connecting two consecutive GPS records (Hofmann et al., 2024). This is the data analyzed by an ISSF. To do so, it uses two major properties of the steps: the step lengths and the relative turning angles (Fieberg et al., 2021). The step length is the distance between two consecutive GPS records. The relative turning angle is the change in orientation between a step and the preceding one (Hofmann et al., 2024).

These two properties describe the movement behavior of the studied subject. It is known as the selection-free movement kernel or resource-independent movement kernel (Fieberg et al., 2021; Forester et al., 2009). Both the step length and relative turning angle present specific distributions of their empirical values derived from the dataset. Based on these distributions, for each step, we can randomly pick values from each of these two statistics that respect their empirical distributions (Thurfjell et al., 2014). These values are then used to create alternative locations that could likely have been chosen by the individual, based solely on its movement behavior. Those theoretical alternative steps are called "available steps" (Avgar et al., 2017) (or "random steps" (Hofmann et al., 2024)). They contrast with the actual next step that the animal actually chose, which is called "observed step" (Hofmann et al., 2024) (or also "used step" (Avgar et al., 2017)). An observed step and its associated available – generated for the same time point – form a "stratum" together (Signer et al., 2017).

Once the available steps are created, environmental data can be gathered from both available and observed steps (Thurfjell et al., 2014).

After defining the selection-free movement kernel and generating the available steps, and once environmental values have been extracted, the ISSF intervenes. For each observed step, the ISSF compares its environmental characteristics with those of the available steps belonging to the same stratum. By repeating this process over numerous steps, the ISSF can determine whether observed steps consistently exhibit environmental features that distinguish them from available alternatives (Thurfjell et al., 2014). If so, this indicates a correlation between specific environmental conditions and the likelihood of selection by the animals.

It is important to note that unlike the Step-Selection Function (SSF), the Integrated Step Selection Function (ISSF) simultaneously accounts for both environmental covariates and movement statistics, thereby allowing for the consideration of cross-correlation and auto-

correlation effects (Fieberg et al., 2021).

Once the influence of environmental factors is quantified by the ISSF, we can refine the estimate of the selection-free movement kernel. Indeed, its first calculation was based on empirical observations that were influenced by the habitat selection of the studied animal (Forester et al., 2009). Consequently, it did not represent a 'pure' selection-free movement kernel but rather an approximation. When this habitat selection is quantified, we become able to isolate it from the movements, to obtain, this time, the real selection free movement kernel, and not an approximation anymore (Fieberg et al., 2021).

As a result, the ISSF creates a model of conditional logistic regression, indicating for each environmental variable the change in selection associated with a one-unit increase in the variable (Thurfjell et al., 2014). This makes it possible to efficiently highlight and quantify the influence of various environmental factors on an individual/population habitat selection (Thurfjell et al., 2014). In doing so, the ISSF improves our understanding of the ecological niche of the studied population and helps better identify strategic areas for its conservation. Therefore, it is a powerful tool that is totally relevant for the specific case of the conservation of the threatened *Oryx dammah*, being potentially able to fill some of the knowledge gaps mentioned earlier.

2.2. Data collection and Timeframe

The data in this study consist of *Oryx dammah* movements in the OROAFR collected by GPS collars covering the period from August 13, 2016 to June 22, 2024 (Figure 5). We use this whole period in this study except for the 10 weeks following the release of reintroduced individuals, to avoid a post-release effect (more details in section 2.6.2.). Reintroductions and data collection were funded by the Environment Agency – Abu Dhabi (EAD) and conducted in cooperation with Sahara Conservation and the Smithsonian Conservation Biology Institution, which provided us with the essential data.

Figure 5: A reintroduced animal fitted with a GPS collar (Ngaba Caleb, RFOROA, June 2025).

2.3. Study site

The study site is centered around the Ouadi Rimé–Ouadi Achim Faunal Reserve (OROAFR) in Chad (Figure 6). In addition to the reserve, the study site encompasses the surrounding region, for a total area of approximately 270,000 km^2.

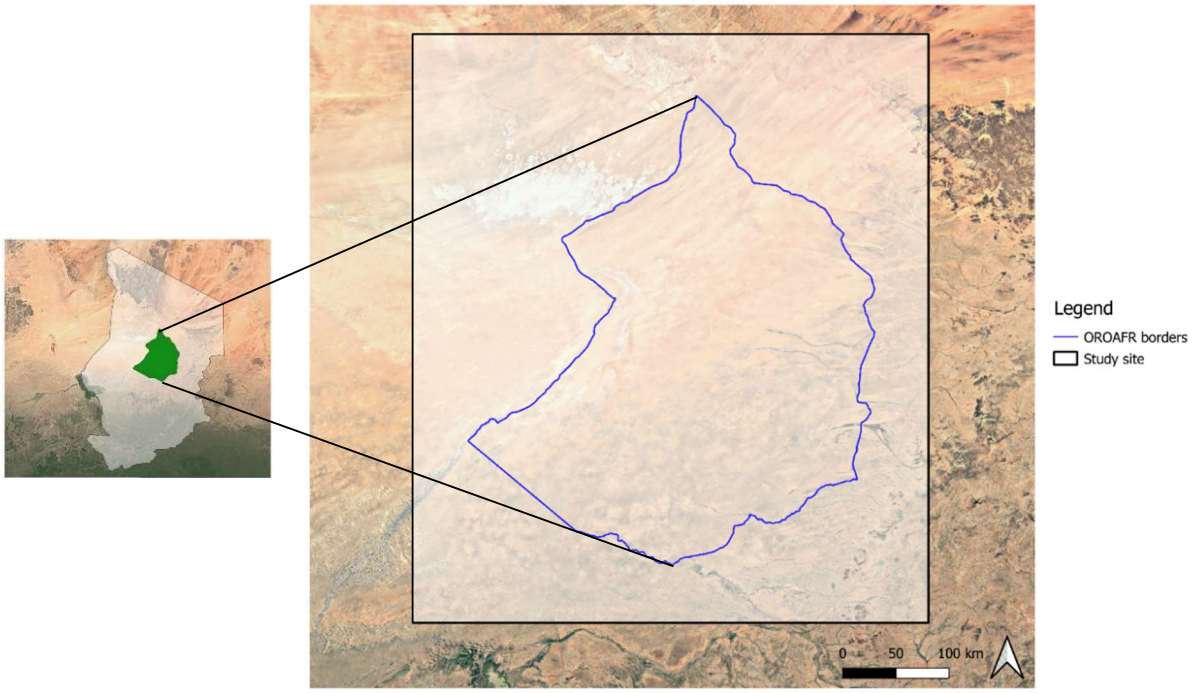

Figure 6: Location of the Ouadi Rimé-Ouadi Achim Faunal Reserve (OROAFR) in Chad, and the geographical extent of the study site (white rectangle).

2.3.1. Climate and ecosystem

The reserve is mainly composed of Sahelian grasslands, with a Saharan climate to the north (Worgue & Yamtibaye, 2023). Temperatures vary from 15 °C during the rainy season to 47 °C during the hot dry season, with an average of 29 °C. Annual precipitation varies from under 100 mm in the northern part to 200–500 mm in the southern part (Worgue & Yamtibaye, 2023).

During the rainy season, precipitation creates wadis, i.e., temporary rivers (Wacher et al., 2022). The two main wadis – Wadi Rimé and Wadi Achim – gave their names to the reserve.

The area is generally flat, with dunes as the main source of relief (Worgue & Yamtibaye, 2023).

2.3.2. Demographics and culture

The area is shared between several Chadian provinces: Batha, Wadi Fira, Bahr El Gazel, Borkou, and Ennedi Ouest (Worgue & Yamtibaye, 2023). The population density is low, with an average of 3.9 inhabitants/km^2 across the five provinces in 2009. Updated data are lacking, but the rapid growth of the population in the country led Worgue & Yamtibaye (2023) to estimate that it had risen to approximately 5.7 inhabitants/km^2 in 2020, causing rapid changes in human impact in the reserve (Worgue & Yamtibaye, 2023). Most of the population lives in the south and is composed of pastoralists, with also a significant agricultural presence (Worgue & Yamtibaye, 2023).

2.4. Factors analyzed

To use the ISSF, we need to define and create indicators for each of the environmental factors that we aim to test.

We sometimes create multiple alternative indicators for a given environmental factor, to test which one is more informative and provides more relevant insight about the best way to study the behavior of *Oryx dammah*. A summary of the different indicators used can be found in Table 1, and more details about them are available in the following sections.

Table 1: A list of all environmental factors studied, and the different indicators used to alternatively represent each factor.

Environmental factor	Environmental indicator
Vegetation cover density	MSAVI – Landsat (resolution: 30 m)
	MSAVI – MODIS (resolution: 250 m)
	DMSAVI (resolution: 250 m)
Woody vegetation density	Sum of MSAVI 2021-2023 (resolution: 30 m)
Wildfires	Presence of fires the previous year (resolution: 1000 m)
	Sum of fires over the period from 2004 to 2023 (resolution: 1000 m)
Topography	Topographic Position Index – Neighborhood 90 × 90 m (resolution: 30 m)
	Topographic Position Index – Neighborhood 210 × 210 m (resolution: 30 m)
	Topographic Convergence Index – Neighborhood 90 × 90 m (resolution: 30 m)
	Topographic Convergence Index – Neighborhood 210 × 210 m (resolution: 30 m)
Human activity	Probability of human presence (resolution: 30 m)

2.4.1. Vegetation factors

2.4.1.1. Vegetation cover density

We assume vegetation cover density to be a reliable indicator of food availability, since most of the vegetation consists of grasslands that contain the plants *Oryx dammah* eats (Kingdon & Hoffmann, 2013; Worgue & Yamtibaye, 2023).

We choose to calculate this factor with the Modified Soil Adjusted Vegetation Index (MSAVI) instead of the classical NDVI because MSAVI has an auto-correction that reduces the inaccuracies created by extensive bare soil surfaces, making it more efficient in arid regions such as the OROAFR (Qi et al., 1994). The MSAVI is calculated with the expression in Equation

(1):

$$MSAVI = \frac{\sqrt{2\times NIR+1-(2\times NIR+\ )^2-8\times(NIR\qquad)}}{2} \quad (1)$$

Where NIR represents the value of the near infrared band of the satellite image, and RED corresponds to the value of the red band of the satellite image (Qi et al., 1994).

Due to the high temporal variation of this factor, we need a temporally dynamic extraction. We perform it using Google Earth Engine. We have several criteria that guide the choice of the appropriate satellite image collection:

- It must be open access
- It must contain the near infrared and red spectral bands, necessary for the MSAVI calculation
- It must cover the 2017 to 2023 period with regular data
- The higher the spatial resolution, the better
- The higher the temporal resolution, the better

We have two image collections that appropriately fit these criteria (Table 2).

Table 2: A summary of the satellite collections that best fit our criteria (LAADS, n.d.-b, n.d.-a; Landsat Acquisition Tool, *n.d.)*

Satellite collection	Spatial resolution	Revisitation time
USGS Landsat 8 Level 2, Collection 2, Tier 1	30 m	16 days
MODIS/Terra Surface Reflectance Daily L2G Global 250m SIN Grid **+** **MODIS/Aqua Surface Reflectance Daily L2G Global 250m SIN Grid**	250 m	< 1 day

Consequently, we have Landsat with a very good spatial resolution and an average temporal resolution, and MODIS with a very good temporal resolution, but an average spatial resolution. We must choose between these two options that focus on different properties. We do not know which – good spatial resolution or good temporal resolution – is the most important to obtain reliable results. Therefore, we will test the MSAVI with both models separately to see which one produces the best results.

2.4.1.1.1. Vegetation productivity

Another indicator we want to test to see if it is a good predictor of movements is the difference in MSAVI (D-MSAVI). It quantifies the variation in MSAVI over a period preceding the GPS record. This indicator is identical to the D-NDVI used in other studies, except that it is based on MSAVI rather than NDVI (Gu et al., 2007). It highlights areas that produce a lot of vegetation. Testing this factor can help determine which – the density of the vegetation cover or the vegetation productivity – is the strongest factor influencing the habitat selection of *Oryx dammah*.

The interval between MSAVI measurements was fixed at 10 days, in accordance with the standard used in other studies (Gu et al., 2007). Extraction with Landsat is not suitable for this indicator, because its larger temporal offset would create significant inconsistencies in the duration of the 10-day time intervals. Therefore, we perform the extraction using MODIS Terra & Aqua, for a maximum temporal resolution.

2.4.1.2. Woody vegetation density

To assess the effect of shade-providing vegetation, we use a composite image previously created by Katherine Mertes of the Smithsonian Conservation Biology Institution and made available via Google Earth Engine (Figure 7). It is a cumulative sum of MSAVI values from the period 2021 to 2023. The fact that it is a sum allows us to emphasize the variations in MSAVI, highlighting the places that are green most of the time due to perennial vegetation, such as trees or shrubs. We assume this layer to be a good indicator of woody plants.

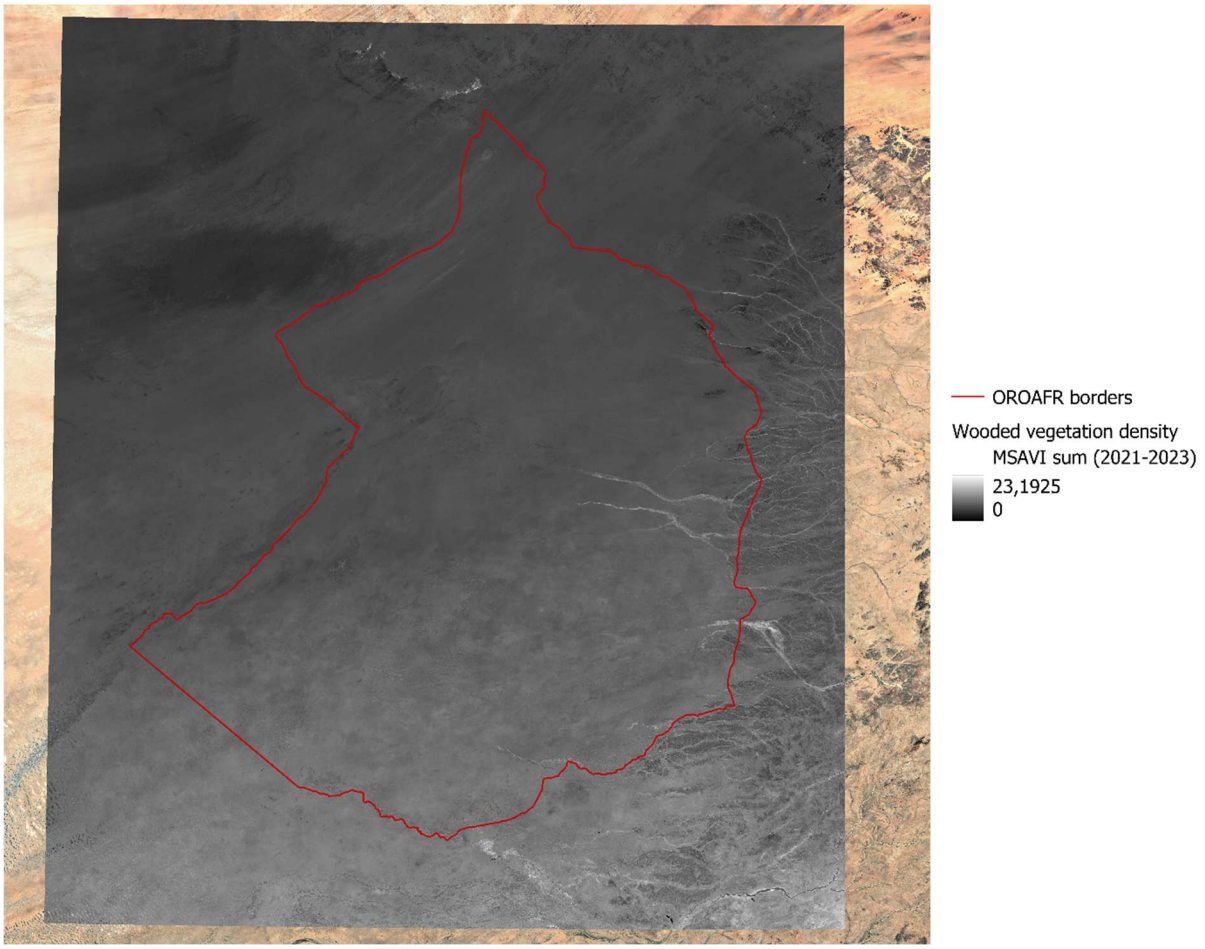

Figure 7: The layer containing the indicator of woody vegetation density, under the form of a sum of MSAVI records from 2021 to 2023.

We assume that despite the layer covering only the period from 2021 to 2023, it is representative of the situation for the whole 2016–2023 period, because this kind of perennial and woody vegetation can be stable over several years.

Even though the initial resolution is 5 m, we download it at a resolution of 30 m to avoid excessively heavy files and to keep consistency with most of the other layers.

Due to the proximity of this indicator with the normal MSAVI that we also use, we test two versions of each model, one with this woody vegetation density and one without it.

2.4.1.3. Wildfires

Fires can be identified by satellite collections freely available on Google Earth Engine. We use the collections `MOD14A1` and `MYD14A` from the MODIS Terra and Aqua satellites. They contain 8-day composite images of burned areas with a 1-km spatial resolution (Giglio & Justice, 2015a, 2015b).

2.4.1.3.1. Binary indicator of wildfires previous year

The first indicator we want to test is a binary variable. It indicates whether a fire was recorded during the year preceding the GPS data year. To do that, we need to extract a binary layer for each year. This allows us to detect effects in the relatively short term.

The code used to extract the images can be found in section A.1.

2.4.1.3.2. Long term sum of fires

Our second indicator is a long-term indicator, using a composite over 20 years, for the period from 2004 to 2023. It is intended to verify that the short-term impact of fires on the habitat selection of *Oryx dammah* is stronger. If this indicator produces better results than the yearly one, it will probably mean that the original explanation for a positive correlation between burned areas and *Oryx dammah* presence is wrong. Indeed, *Schoenefeldia gracilis* is an annual species and we do not expect it to be the cause of a correlation between the presence of fire several years ago and the presence of *Oryx dammah* (Clayton, 1972).

We create a raster for each year from 2004 to 2023 using Google Earth Engine (Figure 8). We download them and then we sum them up to form a single raster counting the number of fires for the whole period.

The code used to generate the yearly images can be found in section A.2.1., and the one used to add up all yearly images is available in section A.2.2.

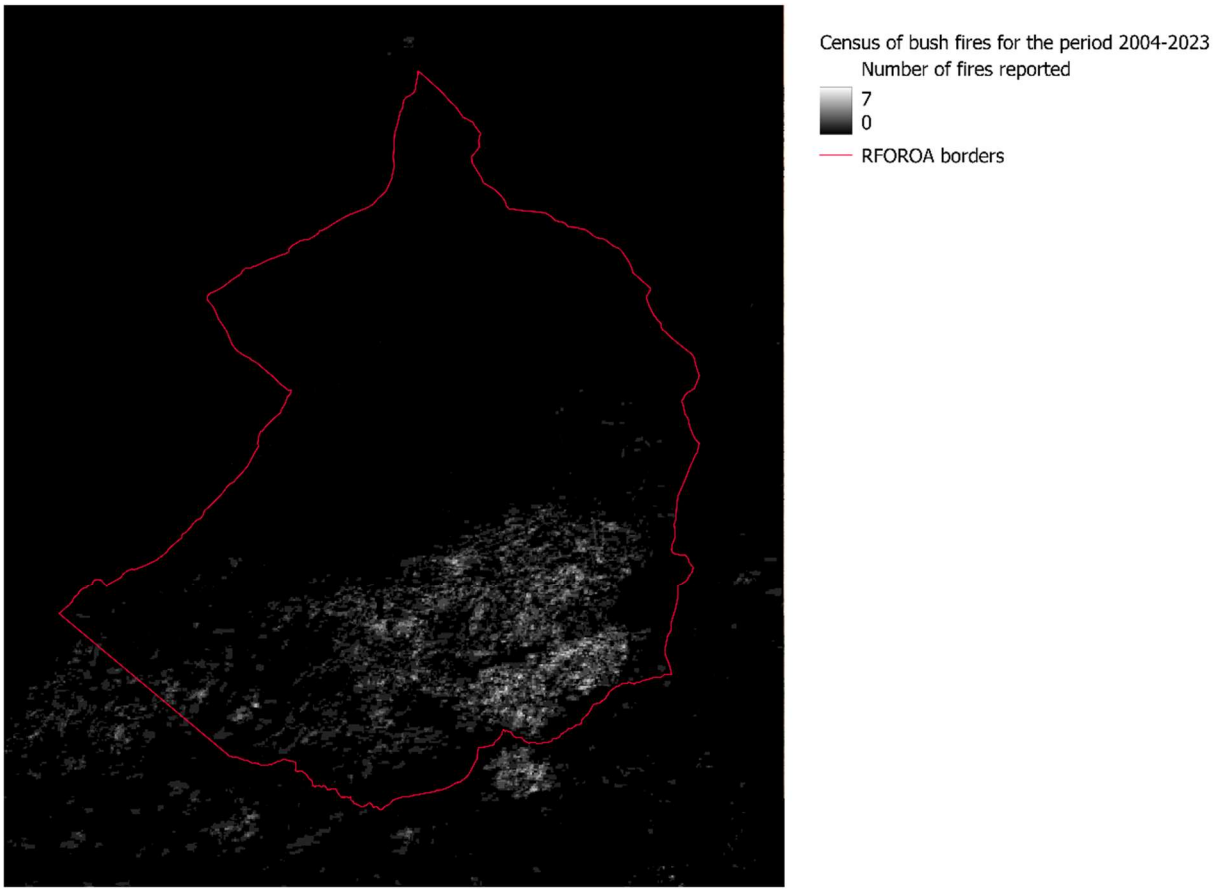

Figure 8: Image of the raster used to extract the number of wildfires for the period from 2004 to 2023.

2.4.2. Topography

We use a Digital Elevation Model (with a resolution of 30 meters) from the Shuttle Radar Topography Mission (SRTM), provided by the Scimitar-horned Oryx Reintroduction Project of the Sahara Conservation NGO (Earth Engine Data Catalog, n.d.). Since elevation and topographic roughness have already been studied, we aim to explore the effects of other topographic indicators.

2.4.2.1. Topographic Position Index

The Topographic Position Index (TPI) is an indicator of local variations in elevation. It calculates the difference in elevation between each pixel of a Digital Elevation Model and its direct environment, called "neighborhood" (Deenick, 2021; Weiss, 2001). The size of the neighborhood can be adapted depending on the scale of the observations we aim to make.

For our study, we want to test two sizes of neighborhood. A small neighborhood grid of 3×3 pixels, which accounts for small variations in topography. This corresponds to an actual size of 90×90 meters and it is the smallest grid we can use. A larger neighborhood grid of 7×7 pixels

can account for a larger scale, especially big dunes, with a neighborhood size of 210×210 meters (Wahed et al., 2023).

Testing these two indicators alternatively can help to give an idea of which scale of relief has the strongest impact on the habitat selection of *Oryx dammah*.

To create the two TPI layers, we use the R function `wbt_diff_from_mean_elev()` from the package `whitebox`. The code is available in section A.3.

2.4.2.2. Topographic Convergence Index

The Topographic Convergence Index (TCI) is another topographic indicator that represents the capacity of a surface to accumulate water from the surrounding area (Amatulli et al., 2020). Such as the TPI, the TCI is calculated in relation to the neighboring pixels. The TCI ranges from −90° to 90°. A cell with a −90° value means that all the water present in the neighborhood flows into this cell. In contrast, a cell with a 90° value means that the cell cannot retain any water because it can flow into all the neighbor cells. A 0° value means that the neighborhood forms a regular slope (Liu et al., 2015).

The size of the neighborhood can again vary, but to keep consistency with the TPI comparison, we choose to keep the same neighborhood sizes: 90 × 90 and 210 × 210 meters.

If this indicator returns better results than the TPI, it will allow us to identify that the main impact of the topography is indirect, through the way it affects the spread of water.

To create the two TCI layers, we use the `CI()` function, from the `StarsExtra` package. The code is available in section A.4.

2.4.3. Human presence

We have access to several GIS layers that we use to define areas with high human presence, grouped into three types:

- Places where human settlements have been reported,
- Transhumance corridors and locations reported to have high concentrations of transhumants,
- Natural and artificial water points, which we assume to be indicators of livestock presence, due to the documented high frequency of such places by herds of pastoralists (Wacher et al., 2022; Worgue & Yamtibaye, 2023).

To account for our diverse indicators of human presence, we use a total of eight vector GIS layers. These layers are complementary, thereby mitigating the problem of non-exhaustive and opportunistic monitoring of each individual indicator:

- A layer containing wells (209 points), created by Katherine Mertes during an incomplete survey in the field,
- A layer containing wadi wells (84 points), complementary to the wells layer from Katherine Mertes, also based on an incomplete survey in the field,
- A layer reporting hafirs (41 points), created by Tim Wacher in 2020 during field missions,
- A layer containing human settlements (1,049 points), created by Katherine Mertes during an incomplete survey in the field,
- A layer containing named settlements (70 points), complementary to the layer of Katherine Mertes, created by Tim Wacher in 2017, during field missions,
- A layer about water points (249 points), complementary to the wells and wadi wells, created by a field survey in 2015, in the framework of the PREPAS project,
- A layer containing the areas with intensive transhumance (71 points), created by a field survey in 2021, in the framework of the PREPAS project,
- A layer containing the main transhumance routes (28 points), created by a field survey in 2021, in the framework of the PREPAS project.

All these layers are merged, and we create a 5-kilometer buffer zone around each point or line. We assume that outside of this area, human presence becomes a negligible factor influencing the habitat selection of *Oryx dammah*. Each pixel contains the value of the distance to the nearest buffer center.

Inside the buffer zones, we assume the centers (points and lines) to have the strongest human presence. Then, the intensity of this factor decreases as we move closer to the borders of the buffers. We assume that the density of human activity increases exponentially from the border of the buffer to the center, due to the center being the focus of interest. To represent that, we convert the distance values of our raster using the expression detailed in Equation (2):

$$y = \left(1 - \frac{x}{5000}\right)^2 \quad (2)$$

In this equation, x is the distance to the center of the buffer, in meters, and y the probability of human presence. With this function, the pixels at the border of the buffer zones will contain a value of 0 and the values increase exponentially until they reach 1 in the center of the buffer zones.

The code used to perform all the operations described in this section can be found in section A.5.1.

The transhumant paths layer cannot be processed in the same way, because it consists of lines rather than points. This significantly increases the computational power required, and our technical resources cannot meet this need. Consequently, we use an alternative method with QGIS to obtain a similar result. The operations in QGIS are described in detail in section A.5.2.

The use of this alternative method in QGIS introduces two small inaccuracies, one of which is visible in the upper part of Figure 9. Combined, they cover an area of less than 60 km^2, which is a very small part of our whole analysis. Furthermore, they are located far away from the main area of activity. Our empirical observations show that no GPS record was ever made in those areas. Consequently, we assume those two inaccurate areas to have no significant effect on the quality of our analysis.

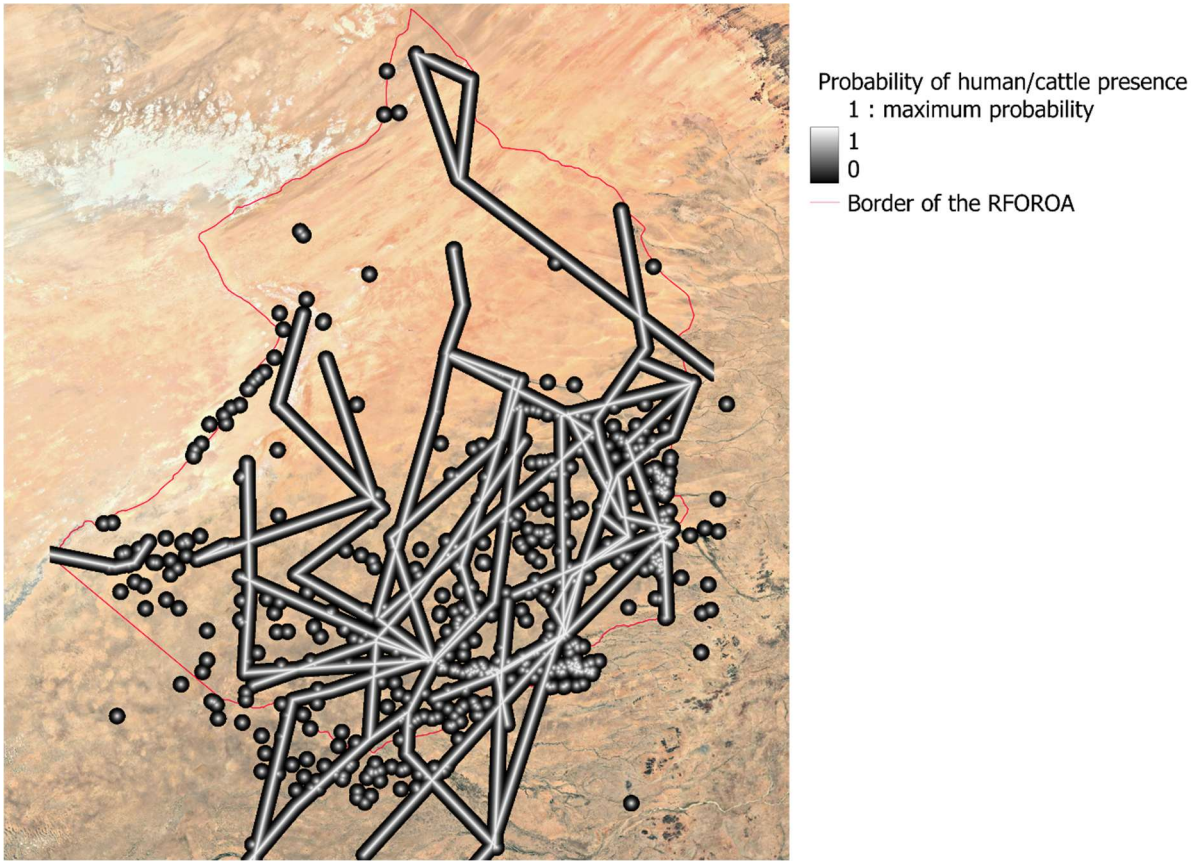

Figure 9: The raster we use represents the probability of encountering human activity. A 5 km buffer zone is applied around each point and line feature, and the probability increases exponentially (rate x^2) as one moves closer to the center. These highest points contained within the buffer zone present a small irregularity.

2.5. Summary of oryx movements data

The movement dataset of *Oryx dammah* we use to conduct the study is provided by the Scimitar Oryx Reintroduction Project. It contains GPS records of the movements of 262 individuals reintroduced in the OROAFR, covering the 2016–2024 period. The data have been partially cleaned, and some useful variables are already included.

The data are structured in a CSV file containing 21 columns and 1,569,425 rows. The details about the dataset are provided in Table 3. We remove columns containing technical information about GPS collars to keep only the data that are ecologically relevant for analyzing the positions and movements of animals.

Table 3: A summary of the information contained in the columns of our GPS dataset

Column name	Description
`collar_ID`	Identifies individual
`collar`	Identifies collars
`burst_ID`	An identification number attributed to each burst. A burst is an uninterrupted series of data
`WAT`	A column including the date and the time when a data was recorded in the format "year-month-day hour:minute:second". The time zone in West African Time (WAT), which corresponds to UTC +1.
`utm_e`	A description of the position where the data was recorded, east of the central meridian. It is on the UTM projection, in meters.
`utm_n`	A description of the position where the data was recorded, north of the equator. It is on the UTM projection, in meters.
`dist`	An indication of the distance between the position of a data record and the position of the previous one, in meters.
`dt`	An indication of the duration since the previous data record, in seconds.
`dx`	An indication of the distance on the X axis between a data record and the previous one.
`dy`	An indication of the distance on the Y axis between a data record and the previous one.
`abs.angle`	The value of the angle formed by the vertical axis and the line obtained by linking the position of this data record to the previous one, calculated in a clockwise direction.
`rel.angle`	The value of the angle formed between the extension of the previous step and the one formed by linking this data record with the previous one.
`long`	A value of longitude, in the geographic coordinate system, in degrees.
`lat`	A value of latitude, in the geographic coordinate system, in degrees.
`height`	A value of the altitude where the data was recorded.
`temp_c`	The temperature in Celsius degrees.
`DOP`	The dilution of precision.

In addition to this main dataset, we have access to another CSV file containing numerous columns with various information about each individual. It is called `sho_master_info_Aug2024`. We only use two columns of this file: `gap_begin` and `gap_end`, which indicate and situate in time any data gaps for a given collar. Our empirical observations show that the smallest time gaps have incomplete monitoring, but we assume that the table is trustworthy regarding the most significant time gaps (lasting a few weeks or months).

2.6. Selection of oryxes movements data

2.6.1. Preparation of data

First, we prepare the data by sorting it by burst and by time (`WAT` column) within each burst. Then, we remove duplicate records and create a new column `id` to uniquely identify each GPS record. In this cleaned version of the dataset, we find 1,569,415 GPS records.

The code used to do this part is available in section A.6.

2.6.2. Organization and filter of the data in a summary table

After that, we must remove all unusable data and make sure we have a standardized and reliable dataset. To do so, we must define criteria for what we consider meeting those principles. We choose the following:

1. All the data must have a consistent time interval to be comparable and have a uniform weight in the analysis.
2. The data must exclude the acclimatization period of animals just after their release. Their unfamiliarity with the area creates irregularities in their behavior during the first weeks (Mertes et al., 2019). To avoid this, we exclude all data from the initial weeks following each animal's arrival in the reserve. Although Mertes et al. (2019) identified post-release effects lasting up to 27 weeks, the most severe behavioral alterations typically dissipate within the first 10 weeks (Mertes et al., 2019). Based on this and on personal communication with Katherine Mertes, we use a 10-week threshold to define the exclusion period.
3. The data from a single individual must fully cover at least one entire season.

2.6.2.1. Creation of a summary table

In the first step, we create a summary table that allows us to better understand the data structure and perform the first filtering. The table uses the data of each individual animal and defines five characteristics (Table 4).

Table 4: A list of the features indicated by the summary table

Column name	Description
burst_ID	A report of all different values presents in the original column `burst_ID`. Those bursts are only approximations of the true bursts, but we can sometimes find gaps within them.
collar_ID	Indicates to which collar ID the burst ID belongs. The `collar_ID` values identify the individuals.
Date_Start	Gives us the earliest record for each `collar_ID`
Date_Start_valid	Gives us the earliest data record after the end of the acclimatization period (defined as 70 days) for each value in `burst_ID`
Date_End	Gives us the last record for each value in `collar_ID`

We also have 261 unique values in `collar_ID` that correspond to 261 animals monitored during the period. The result is a table of 285×5 dimensions, meaning 285 unique values in `burst_ID`, indicating 285 data series. The difference between these two values comes from the fact that the same collar may have been used for multiple animals at different periods. The `burst_ID` variable best represents a single animal's data.

The code used to create the summary table is available in section A.7.

2.6.2.2. Add date gaps

This summary table has a limitation. Some significant data gaps are not reflected by changes in the `burst_ID` column. Their duration varies from several hours to multiple years.

We can spot the most significant ones because they are indicated in the gap columns of the additional file `sho_master_info_Aug2024`.

For each collar ID, we extract the corresponding gaps in `sho_master_Aug2024` if there are any. We put the start and end dates of the gap in two new columns `gap_begin` and `gap_end` in the summary table. Then, when we have a gap for one of our `collar_ID`, we split the affected data segment into two separate data entries: one before the gap and one after. This results in some burst_IDs being split, increasing the number of entries in the summary table from 285 to 304.

The R code used to do these operations is available in section A.7.1.

2.6.2.3. Filtering of incomparable time intervals

Now, we want to calculate the average time interval between two consecutive records within each burst. First, we create a new column `points` that indicates the number of GPS records for each burst. Then, we calculate the average time interval using Equation (3).

$$average\ time\ interval\ for\ a\ burst = \frac{Date\_end - Date\_start}{number\ of\ GPS\ record} \quad (3)$$

Once this is done, we take the obtained values, round them, and add them to a column called `interval_category`.

We create a figure showing the distribution of the bursts within the different categories of time interval (Figure 10). This allows us to observe that most of the data are split between 1-hour interval bursts and 4-hour interval bursts.

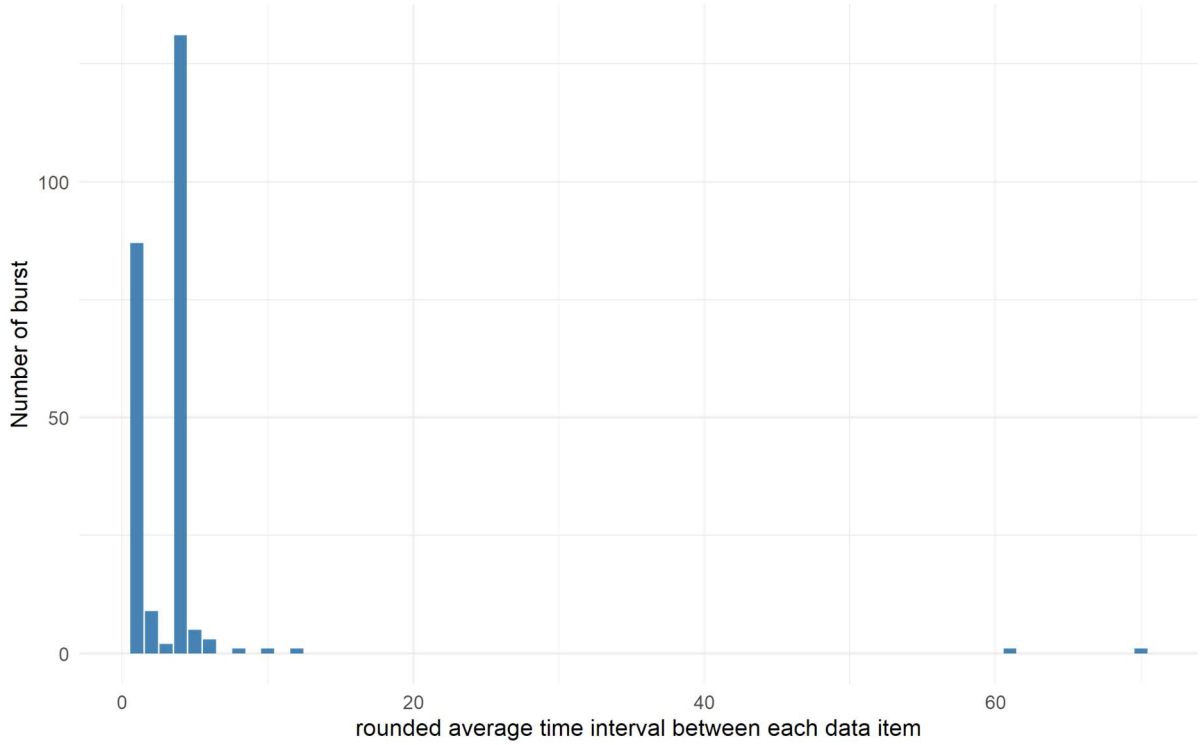

Figure 10: Distribution of the bursts by their rounded average time interval between two consecutive GPS records

A valid ISSF analysis requires a constant time interval between all data points (Avgar et al., 2017; Fieberg et al., 2021). Consequently, to include the highest amount of relevant data, we must resample some data. We choose to resample the whole dataset to a 4-hour interval, which allows us to include the two main time intervals (1-hour and 4-hour) as well as the third most common: the 2-hour interval.

However, we remove the bursts with a 3-hour interval and those with intervals greater than 4 hours, due to their low number and because resampling them would require significantly increasing the time interval, which would reduce accuracy. After this filtering step, our dataset is reduced from 304 to 227 valid bursts.

The code used to perform these operations can be found in section A.7.2.

2.6.2.4. Determination of the seasonal cover

As already mentioned, we want to analyze the data within a seasonal framework. This means we need to determine which data can be used to analyze which season. To do so, we add a new column for each of the three seasons from 2016 to 2024 to our summary table. This represents an addition of 27 new columns. We then compare the dates of each season with the dates of each burst in the columns `Date_Start_valid` and `Date_End`. If a season fits entirely within the time frame of a burst, we add a 1 in the corresponding season's column for this burst. If a season does not fit entirely in a burst, we add a 0 we add a 0 in the corresponding column for this burst.

This allows us to see which bursts cover a sufficient time range and which ones are too short. We remove all bursts that do not cover at least one season fully.

Excluding the bursts that do not cover a season fully allows us to make sure that each part of a season is represented by the same number of bursts, thereby preventing some irregularities that would introduce inaccuracies in the analysis. In addition, our dataset is very large and would eventually exceed the capacity of our computing resources if not reduced. This operation thus has both a methodological and practical purpose. As a result, the number of bursts in our summary table is reduced from 227 to 161.

We can also create a table showing the number of bursts for each season for information purposes (Table 5).

Table 5: A list of every season within the time frame, with the number of bursts fully covering each one

Season	Number of bursts
hot_dry_2017	17
rainy_2017	25
cool_dry_2017	23
hot_dry_2018	27
rainy_2018	26
cool_dry_2018	19
hot_dry_2019	29
rainy_2019	32
cool_dry_2019	27
hot_dry_2020	37

rainy_2020	31
cool_dry_2020	14
hot_dry_2021	12
rainy_2021	5
cool_dry_2021	5
hot_dry_2022	38
rainy_2022	39
cool_dry_2022	36
hot_dry_2023	40
rainy_2023	40
cool_dry_2023	38

This analysis reveals an uneven distribution of data across seasons. Some seasons are represented by up to 40 bursts, while others have fewer than 15 bursts. Notably, the year 2021 has a particularly low number of bursts, with only 5 bursts each for the rainy and cool dry seasons.
However, we assume that this low data volume in 2021 will not significantly affect the overall analysis, since all seasons are combined during the modeling process.
The code used to perform these operations is available in section A.7.3.

2.6.3. Filtering of the original dataset

2.6.3.1. Filtering of the bursts

Now that we have defined which `burst_ID` is relevant for which date, we can remove all data that do not belong to a valid burst kept in the summary table. This filtering reduces the dataset from 1,569,425 to 1,330,099 records.

The code we used to do these operations is available in section A.8.1.

2.6.3.2. Filtering of the incomplete seasonal data

Next, we create a new column, `season_year`, in the dataset to indicate the season to which each GPS record belongs. Using the summary table, we then remove data points that, although part of a valid burst, occur during a non-valid period for that burst (i.e., a season not fully

covered).
As a result of these operations, the dataset is reduced from 1,330,099 to 802,148 GPS records. This step improves the regularity of the data within each season. Although it significantly reduces the number of observations, it is necessary to avoid handling a dataset too large for our computational resources. Thus, this operation serves both methodological and practical purposes.
Simultaneously, we add a new column, `season`, containing the full name and year of the season corresponding to each record.

The code used for these operations is available in section A.8.2.

2.6.3.3. Resampling of the dataset

Earlier, we decided to resample the dataset to a 4-hour interval. To achieve this, we convert the dataset in R to a `track_xyt` object using the `amt` package. This conversion enables us to apply several specialized functions designed for datasets such as ours. In particular, we use the `track_resample()` function, which efficiently resamples the data to a 4-hour interval with a specified tolerance. The tolerance parameter allows the function to split the data into bursts whenever it detects a temporal gap larger than the tolerance between two consecutive records. We set the tolerance to 10 minutes. After resampling, the dataset is reduced to 394,105 records.

The code used for this procedure is available in section A.8.3.

2.7. Extraction of movement statistics

In this section, we calculate two movement statistics from our GPS points: step length (Figure 11) and relative turning angle (Figure 12) (Hofmann et al., 2024). To do this, we use the `steps_by_burst()` function from the `amt` package, which automatically computes these statistics while organizing the data into the appropriate format for subsequent analysis. Data points that do not belong to any burst are also automatically removed, reducing the dataset from 394,105 to 390,713 records.

The code used for this process is available in section A.9.

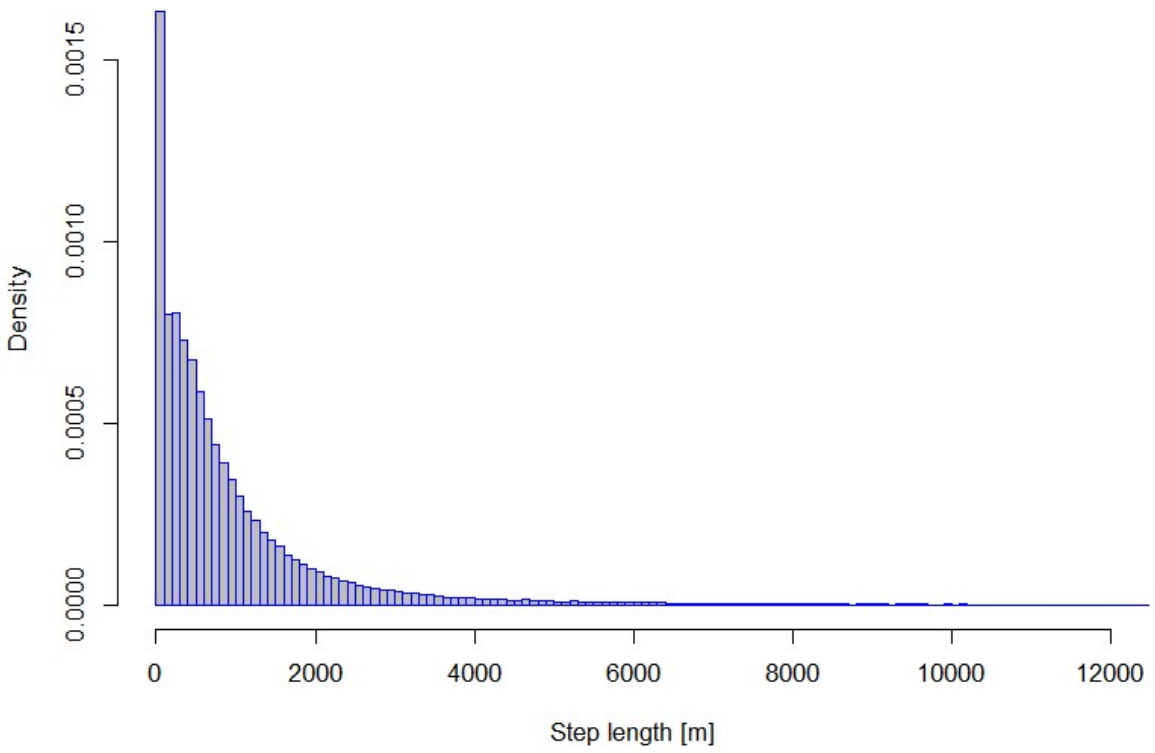

Figure 11: Distribution of the step lengths

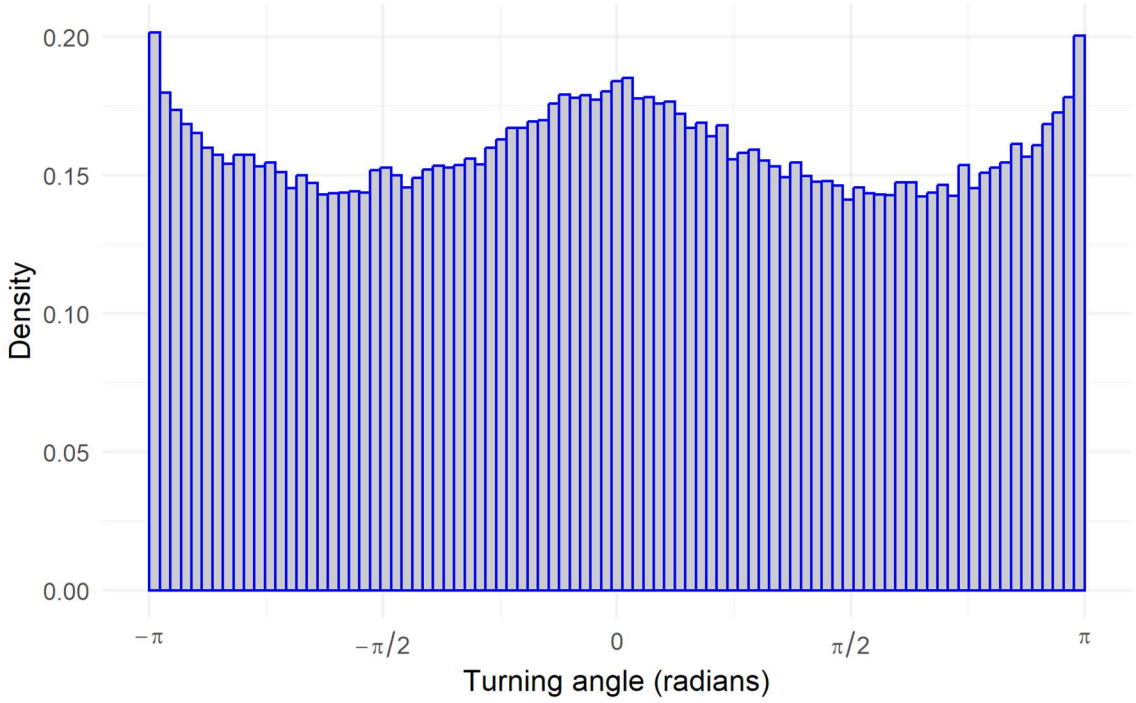

Figure 12: The spread of the relative turning angles; 0 indicates no change in direction, π and -π meet each other and express a change in direction of 180°. Positive values indicate a clockwise change in direction, while negative values indicate a counterclockwise change in direction.

2.8. Creation of the available steps

Now that we have a clean dataset representing the observed steps of *Oryx dammah*, we proceed to generate the available steps. These available steps are created based on the distribution of our observed movement statistics (Thurfjell et al., 2014). To account for potential seasonal variation in movement behavior, we divide the dataset into three subsets, each corresponding to a different season.

We use the `random_steps()` function from the `amt` package to generate available steps. This function estimates step lengths by fitting a parametric Gamma distribution to the empirical data (Signer, Smith, et al., 2024), while turning angles are modeled using a parametric von Mises distribution (Signer, Smith, et al., 2024). Using parametric distributions helps avoid bias in these estimations (Fieberg et al., 2021; Forester et al., 2009). Our step length distribution poses no issue, as it follows a common shape well approximated by the fitted Gamma distribution (Figure 13). However, the turning angle distribution is bimodal and thus cannot be adequately described by the simple von Mises or uniform distributions available in `random_steps()` (Figure 14).

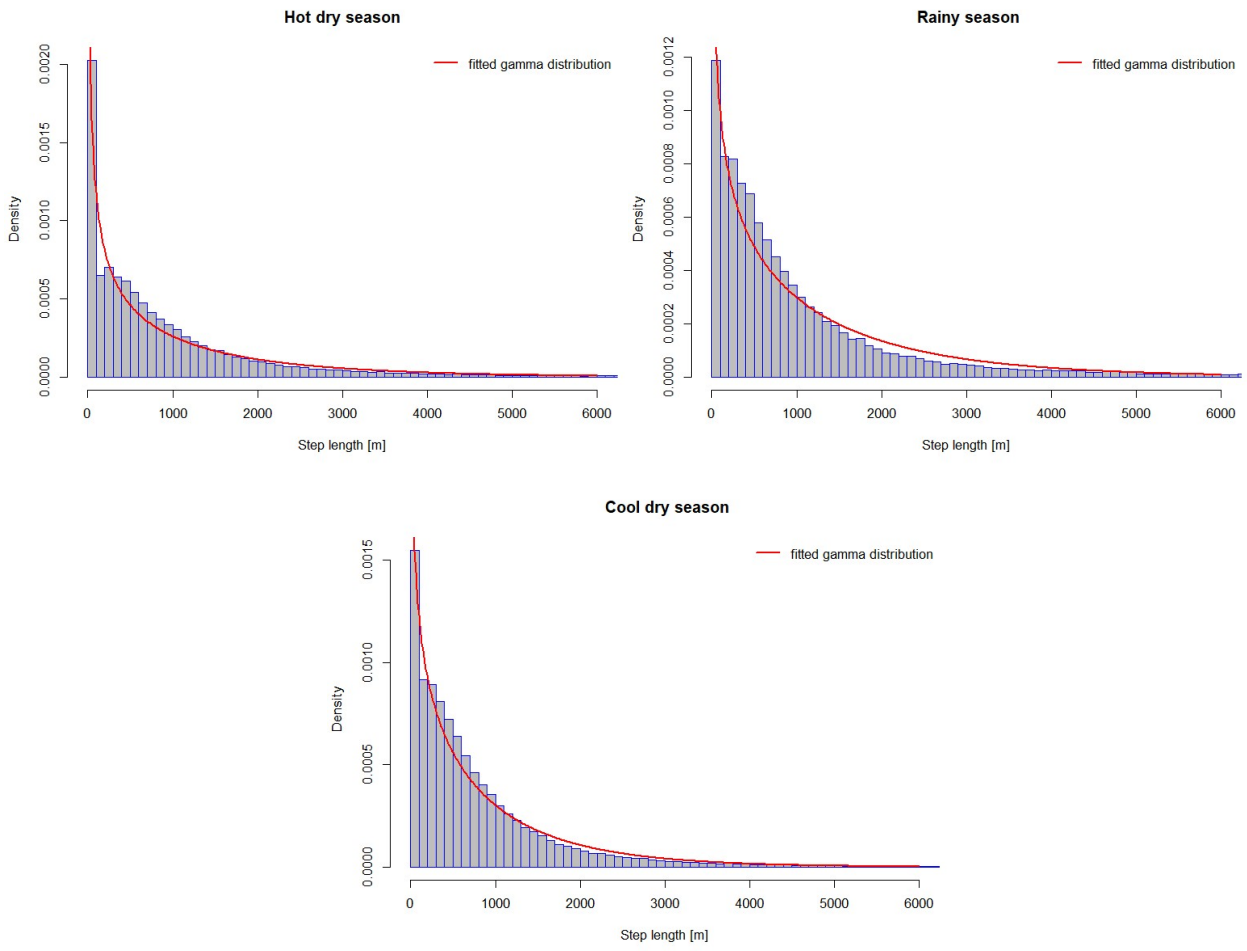

Figure 13: The empirical step length distribution of each season compared to their fitted Gamma distribution of

`random_steps()` in red; the Gamma function is an acceptable approximation of the empirical observations.

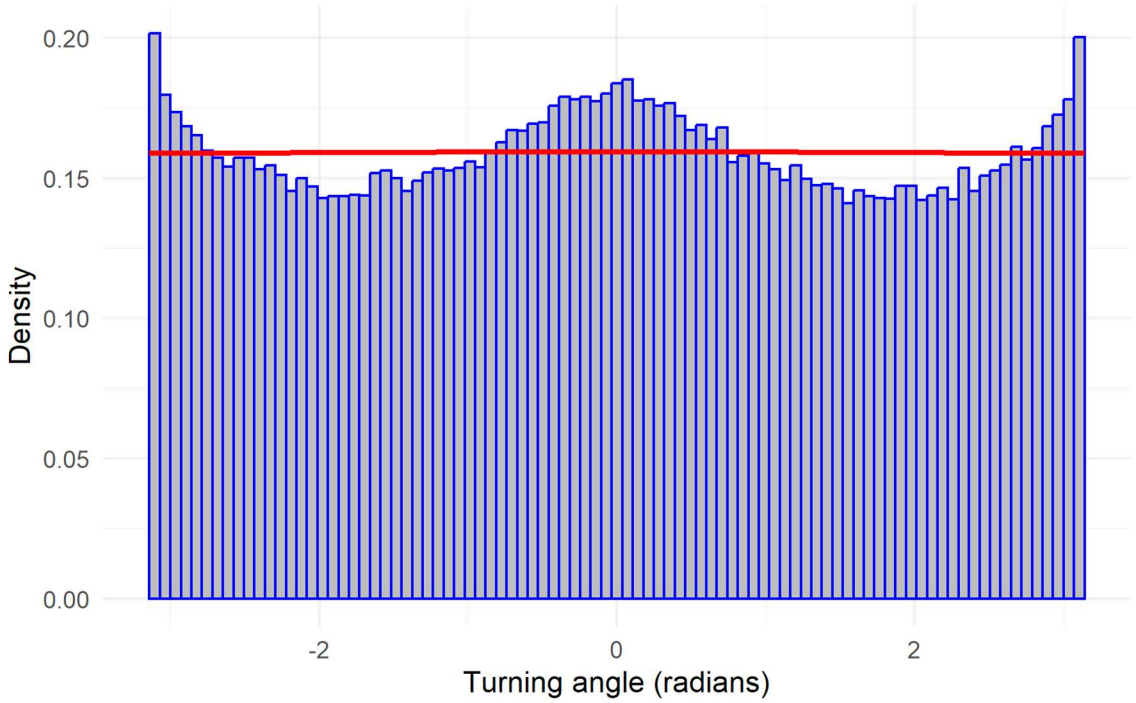

Figure 14: The empirical distribution of the relative turning angle (for the whole dataset) compared to the fitted von Mises distribution from the `random_steps()` in red; due to the second local maximum near π/-π, the von Mises function cannot correctly approximate the empirical observations, making it appear extremely flat.

To address this issue, we assume that modeling the turning angle distribution as a mixture of two von Mises distributions – forming a bimodal von Mises distribution – provides a suitable approximation. This approach is a classic method for analyzing circular data with two local maxima (Jiang, 2009) and has previously been applied to animal turning angles (Hillen et al., 2017). We use specialized functions to fit this bimodal von Mises distribution (Figure 15) and generate a vector of turning angle values consistent with the fitted distribution. This vector is then provided as an argument to the `random_steps()` function, enabling it to generate available steps that respect this more complex distribution, which it cannot handle natively. We apply this procedure separately to each of our three seasonal subsets.

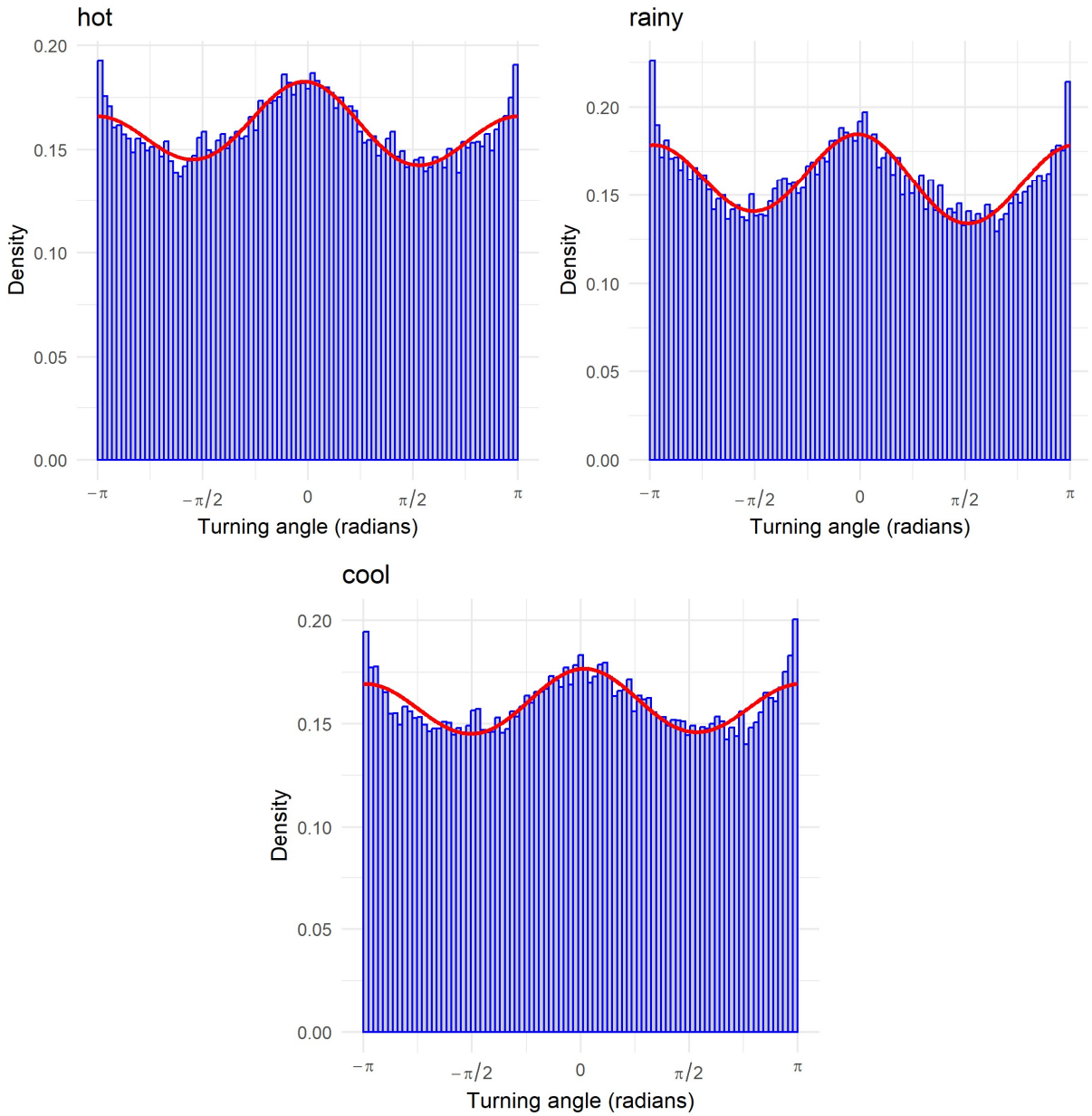

Figure 15: The empirical distribution of relative turning angles for each season (hot dry, rainy, and cool dry respectively); in red, the approximation of these empirical distributions by a parametric bimodal von Mises distribution.

Even though our Gamma and bimodal von Mises distributions do not perfectly describe the movement statistics, this is not problematic. Indeed, these serve only as "proposal distributions" (to borrow the term used by Pohle et al., 2024), which are expected to approximate the true distributions reasonably well.

Regarding the number of available steps per stratum, this depends on the dataset size and can range from 2 to 200, with 10 or 20 being standard choices (Thurfjell et al., 2014). Given that our dataset is large and includes many individuals, 10 available steps per observed step suffices to produce reliable results while avoiding technical limitations that may arise from generating too many random steps.

The `random_steps()` function automatically creates a `step_ID` column that links each alternative step to its corresponding observed step, using the original dataset's `id` column as reference. It also generates a `case_` column with binary values distinguishing available steps (0) from observed steps (1).

This process increases the size of our seasonal subsets to 1,549,207 records for the hot dry season, 1,049,020 for the rainy season, and 1,666,324 for the cool dry season, totaling 4,292,896 records.
The code to create the parametric curves for the relative turning angle is available in section A.10.1, and the code for generating the available steps can be found in section A.10.2.

2.9. Extraction of the environmental factors

In this section, we assign environmental parameter values to each record in our three seasonal subsets. This requires using two types of data extraction methods: classical extraction and dynamic extraction.

Classical extraction is used for static variables that do not vary over time, whereas dynamic extraction is employed for time-dependent variables. This distinction ensures that each environmental indicator is integrated appropriately, depending on its temporal characteristics.

2.9.1. Simple extraction on the non-time-dependent factors

The classical extraction method is applied to most of our indicators which are represented as raster layers. We want to extract the value of the pixel corresponding to the GPS coordinates in each raster for each record. To ensure data consistency, we resample our rasters to harmonize their spatial extent, resolution, and Coordinate Reference System (CRS) using the package `terra` from R. We then combine the harmonized rasters to perform a grouped extraction. The environmental indicators extracted through this method include:

- The total number of fires for the period from 2004 to 2023
- The two Topographic Position Index layers (with 90×90 m and 210×210 m neighborhood sizes)
- The two Topographic Convergence Index layers (with 90×90 m and 210×210 m neighborhood sizes)
- The group of eight annual rasters containing the number of fires for each year between 2016 and 2023
- The sum of MSAVI values from 2021 to 2023: used as an indicator of woody vegetation density

- The probability of encountering human activity

The code we use to perform this is available in section A.11.

2.9.2. Dynamic extraction of time-dependent indicators

For time-dependent factors, a dynamic extraction using Google Earth Engine (GEE) is necessary. Using raster layers containing long-term median values would not accurately reflect the conditions at the specific moments of interest in our seasonal subsets. The factors that are concerned by this procedure are the three indicators of vegetation cover:

- MSAVI – Landsat version
- MSAVI – MODIS version
- D-MSAVI – MODIS version

For each of these indicators, we design a script that iterates through each seasonal subset and uses the timestamps and geographical coordinates to find the most appropriate image in the chosen satellite collection. Then, we extract the necessary values and calculate the MSAVI before storing it in a new column within the seasonal subsets. To avoid too much NA data, our code extracts the values from the temporally nearest image containing valid data, within a limited time interval. For Landsat, we set it at 7 days, and for MODIS, at 3 days. We also extract, in a distinct column, the time offset between each GPS record and the satellite image used to associate a MSAVI value to it. This allows us to quickly check if anything went wrong.

Once this is done, we can download the results of the dynamic extraction and merge together the files of the same season. Because the seasonal subsets were modified with new columns and a different order of rows and columns, the easiest way to obtain complete and sorted seasonal subsets is to append the newly created columns to each subset that we created just before the export.

The codes used to do these operations are available in section A.12.1 for the MSAVI Landsat, A.12.2. for the MSAVI MODIS, A.12.3. for the D-MSAVI and the code to merge the data from a same season is in section A.12.4.

2.10. Statistical analysis

2.10.1. Collinearity pair-to-pair assessment

To help us design meaningful models, we want to make sure that they will not include any variables that present collinearity. We use the package `corrplot` to estimate the collinearity between each pair of environmental indicators (Figure 16). This allows us to easily identify

whether there is a significant collinearity between any variable. There are various ideas about which threshold should indicate a significant correlation between two variables. Depending on the standard we use, a correlation greater than 0.7 or even 0.4 can be considered problematic (Dormann et al., 2013). In our case, the only variables that reach such thresholds are those that are alternative indicators of the same environmental factor and are not intended to be used together anyway. The highest correlation between two variables of different environmental covariates reaches |0.22|, which shows that we have no restrictions on the models we can create.

The code used to perform these operations is available in section A.13.1. and the results for each season can be found in section A.13.2.

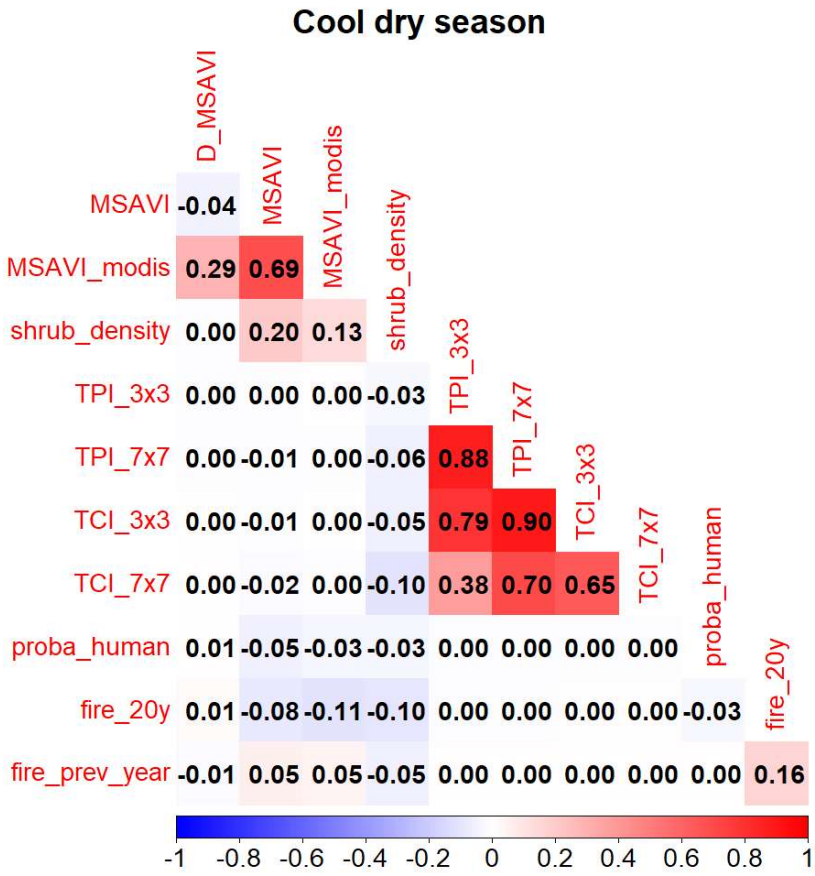

Figure 16: Example of the correlation matrix of our environmental covariates, shown here for the cool dry season; values greater than > 0.4 are considered problematic.

2.10.2. Presentation of the models

We want to test multiple alternative models to assess which combination of variables is the most efficient at explaining the movements of *Oryx dammah*. We are also interested in testing, for some indicators, whether they have a linear or a quadratic effect, by testing both.

- Vegetation:

- Vegetation cover: we have three alternative indicators (MSAVI-Landsat, MSAVI-MODIS, D-MSAVI) and we will use just one at a time
    - All three indicators are tested in both linear and quadratic specifications for a total of six variants of this indicator.
    - Woody vegetation density: the single indicator will be tested in both linear and quadratic specifications. However, because it is derived from the MSAVI, similarly to the vegetation cover, we expect both to have the same specifications. We will test their linear/quadratic behavior together.
- Wildfires: we have two alternative indicators (fire previous year and sum of fires during the period from 2004 to 2023)
    - The first one is categorical, so it does not require specification testing. For the second one, due to its low range (only seven units), we assume a linear effect.

- Topography: we have four alternative indicators (TPI 3 × 3, TPI 7 × 7, TCI 3 × 3, TCI 7 × 7) and we will use only one at a time.
    - All four indicators will be tested in both linear and quadratic specifications, for a total of eight variants of this environmental factor.
- Human activity: the single indicator is kept in every model. We assume it to be linear because the layer used has its values already expressed in a quadratic form.

This leads us to 96 different models to test (Table 6).

Table 6: A summary table of the different versions tested for each factor

	Topography	Vegetation cover	Woody vegetation	Wildfires	Human density	Total number of combinations
Number of indicators	4 ×	3 ×	1 ×	2 ×	1 ×	= 24
Number of specifications tested	2 ×	2 ×		1 ×	1 ×	= 4
Number of versions of each factor	8 ×	6 ×		2 ×	1 ×	= 96

All these 96 models will be tested for each of the three seasons.

We are interested in ISSF and not simple SSF, due to their higher accuracy (Fieberg et al., 2021). As a result, we add to the environmental indicators of each model the following movement statistics: $step\ length$, $log(step\ length)$ and $cos(turning\ angle)$. Indeed, it has been demonstrated that adding $log(step\ length)$ to the analyses, and using $cos(turning\ angle)$ as the sole indicator of the turning angle reduces bias (Avgar et al., 2016; Duchesne et al., 2015).

For all our models, we assume the movement statistics are constant and that there is no variation in their distribution depending on the environment. This allows us to clearly define the scope of our analysis.

2.10.3. Standardization of the data

We will not use directly the raw environmental values of our indicators. Instead, we will standardize them to ensure a better convergence of the models, as is common practice when using ISSFs (mean = 0, standard deviation = 1) (Chan et al., 2024; Fieberg et al., 2021).

We perform the standardization over the entire dataset, rather than the seasonal subsets. This allows for better comparison of the data afterwards. This method is reported to create an asymptotic bias in case of extreme variation of values between the subgroups in multilevel analyses (Wang et al., 2019). Despite this risk, this method is still commonly used in statistics, and none of the studies we could find about linear regressions and ISSFs mentioned that it poses the same problem in this context (Wang et al., 2019). For these reasons, we consider any inaccuracy caused by the standardization to be negligible in the results of our study.

The code used to do this is available in section A.14.

2.10.4. ISSF models calculations

Once we have defined all the models we want to test, we can calculate the ISSF of each one for all seasons, for a total of 288 ISSF models. To do this, we use a function from the `amt` package: `fit_issf()`.

The code used to do this is available in section A.15.

2.10.5. Performance assessment of the models

Once we have all the ISSFs for all our models, we are interested in comparing them, to determine which one is the most efficient at explaining the movements. To do so, we use the corrected Akaike Information Criterion (AICc), similarly to several similar studies before us (Majaliwa et al., 2022; Naawa et al., 2025).

The code used to perform this comparison is available in section A.16.1.

The full results of the AICc tests are available in section A.16.2. for the hot dry season, A.16.3.

for the rainy season, and A.16.4. for the cool dry season.

We can then save the ISSF models with the best AICc score for each season. The code to do this can be found in section A.17.

2.10.6. Production of Relative Selection Strength figures

The relative selection strength (RSS) is a simulation where we fix all environmental parameters except one, which we vary to observe its individual effect on habitat selection (Avgar et al., 2017). We can display it in a figure, to have an easy and very informative view about the individual impact of the variation of an environmental factor on the behavior of the study subject.

For visual purposes, Avgar et al. (2017) use the log(RSS) in their methods. However, using RSS directly allows for a more intuitive understanding of the real impact of the variable. This is why we use it whenever it does not reduce the clarity of the representation (Majaliwa et al., 2022). Consequently, we use RSS most of the time, switching to log(RSS) only when the data requires it.

We express the results between the 1^{st} and 99^{th} percentile, to exclude the most extreme values and to restrict the range of observations to conditions that are representative of what can be met in the study site. We also express the values of the environmental factors on their natural scale rather than the standardized one to ease interpretation and enable more concrete reasoning.

We choose a confidence interval of 95%, calculated with the "standard errors" method. This method is slightly less accurate than the bootstrap, but can still be used because the difference is usually slight (Fieberg et al., 2021). However, we face technical limitations for using the bootstrap method, due to its high computational demands. As we only aim for visualization, we assume that a small inaccuracy in the confidence interval does not pose a major problem.

The code used to create the RSS is available in section A.18.

3. Results

3.1. Best model

The results of the AICc analysis show that the best model is the one that uses the following indicators:

- Topography: TCI (210 × 210 m grid size) with a quadratic effect,
- Wildfires: sum over the period from 2004 to 2023,
- Vegetation cover: MSAVI (based on Landsat images) with a quadratic effect,
- Woody vegetation density, with a quadratic effect.

This is true for all seasons (Table 7 for hot dry season, Table 8 for rainy season, Table 9 for cool dry season). Consequently, we only study the conclusions that this model provides.

Table 7: Results of the best ISSF model for hot dry season; "TCI7_scaled" is the standardized version of the TCI with a 210 × 210 m neighborhood grid. "I(TCI7_scaled^2)" is the quadratic effect of this variable. "proba_human_scaled" is the standardized probability of human presence in a buffer of 5 km around the points reported to have significant human presence. "MSAVI_landsat_scaled" is the standardized MSAVI values calculated with Landsat and the following indicator "I(MSAVI_landsat_scaled^2)" is its quadratic effect. "fire_20y_scaled" is the standardized sum of fires over the 2004 to 2023 period. "shrub_density_scaled" is the standardized woody vegetation density indicator, followed by its quadratic effect "I(shrub_density_scaled^2)". "sl"_ is the step length, followed by its logarithmic values "log_sl_". "cos_ta" is the cosine of the turning angle. "coef" represents the coefficients in a logarithmic scale, while "exp_coef" is its expression on a natural scale. "se_coef" indicates the standard error of the coefficients. "z" is the z-statistic and "p" is the p-value (significance of the results).

variable	coef	exp_coef	se_coef	z	p
TCI7_scaled	-0,098	0,9066	0,0034	-28,6306	2,797E-180
I(TCI7_scaled^2)	0,0049	1,0049	0,0026	1,9132	0,055721918
proba_human_scaled	-0,0375	0,9632	0,0058	-6,423	1,3365E-10
MSAVI_landsat_scaled	0,9536	2,5951	0,0149	64,184	< 2e-16
I(MSAVI_landsat_scaled^2)	0,0896	1,0938	0,0095	9,4715	2,75869E-21
fire_20y_scaled	-0,0541	0,9474	0,0053	-10,2332	1,40748E-24
shrub_density_scaled	0,6867	1,9871	0,0066	104,5052	< 2e-16
I(shrub_density_scaled^2)	0,0187	1,0189	0,0025	7,5684	3,77783E-14
sl_	0,0001	1,0001	0	15,3663	2,75513E-53
log_sl_	0,0127	1,0128	0,0024	5,3618	8,24149E-08
cos_ta_	0,0246	1,0249	0,0046	5,3602	8,3123E-08

Table 8: Results of the best ISSF model for the rainy season; for the meaning of the variables and the other columns, see Table 7.

variable	coef	exp_coef	se_coef	z	p
TCI7_scaled	-0,0314	0,9691	0,004	-7,9087	2,59994E-15
I(TCI7_scaled^2)	0,0112	1,0112	0,0031	3,6297	0,000283719
proba_human_scaled	-0,0409	0,9599	0,0066	-6,1555	7,48262E-10
MSAVI_landsat_scaled	0,2372	1,2677	0,0092	25,6682	2,6482E-145
I(MSAVI_landsat_scaled^2)	-0,0348	0,9658	0,002	-17,8073	6,19773E-71
fire_20y_scaled	-0,0481	0,953	0,006	-8,0713	6,95736E-16
shrub_density_scaled	0,3415	1,4071	0,0068	50,3511	< 2e-16
I(shrub_density_scaled^2)	-0,0076	0,9925	0,0031	-2,4392	0,01471943
sl_	0	1	0	4,2468	2,16836E-05
log_sl_	0,0933	1,0978	0,0034	27,6976	7,4584E-169
cos_ta_	-0,0299	0,9706	0,0054	-5,4853	4,12679E-08

Table 9: Results of the best ISSF model for the cool dry season; for the meaning of the variables and the other columns, see Table 7.

variable	coef	exp_coef	se_coef	z	p
TCI7_scaled	-0,0764	0,9264	0,0032	-23,5895	4,9352E-123
I(TCI7_scaled^2)	0,0005	1,0005	0,0025	0,191	0,848556392
proba_human_scaled	-0,0019	0,9981	0,0064	-0,2981	0,76563365
MSAVI_landsat_scaled	0,7378	2,0912	0,0107	69,1021	< 2e-16
I(MSAVI_landsat_scaled^2)	-0,1429	0,8668	0,0049	-29,203	1,7751E-187
fire_20y_scaled	0,0097	1,0098	0,0048	2,0269	0,042674823
shrub_density_scaled	0,409	1,5054	0,006	67,9344	< 2e-16
I(shrub_density_scaled^2)	0,0189	1,0191	0,0028	6,786	1,15261E-11
sl_	-0,0002	0,9998	0	-52,5595	< 2e-16
log_sl_	0,1295	1,1383	0,0028	45,8837	< 2e-16
cos_ta_	-0,0108	0,9893	0,0044	-2,4385	0,014748285

The full results of the AICc analyses are available in sections A.16.2. for hot dry season, A.16.3. for the rainy season, and A.16.4. for the cool dry season.

3.2. Movement statistics

We created a parametric distribution to fit the empirical observations (Figure 17, Figure 18). We used them to generate the available steps, but they also provide an interesting output on their own.

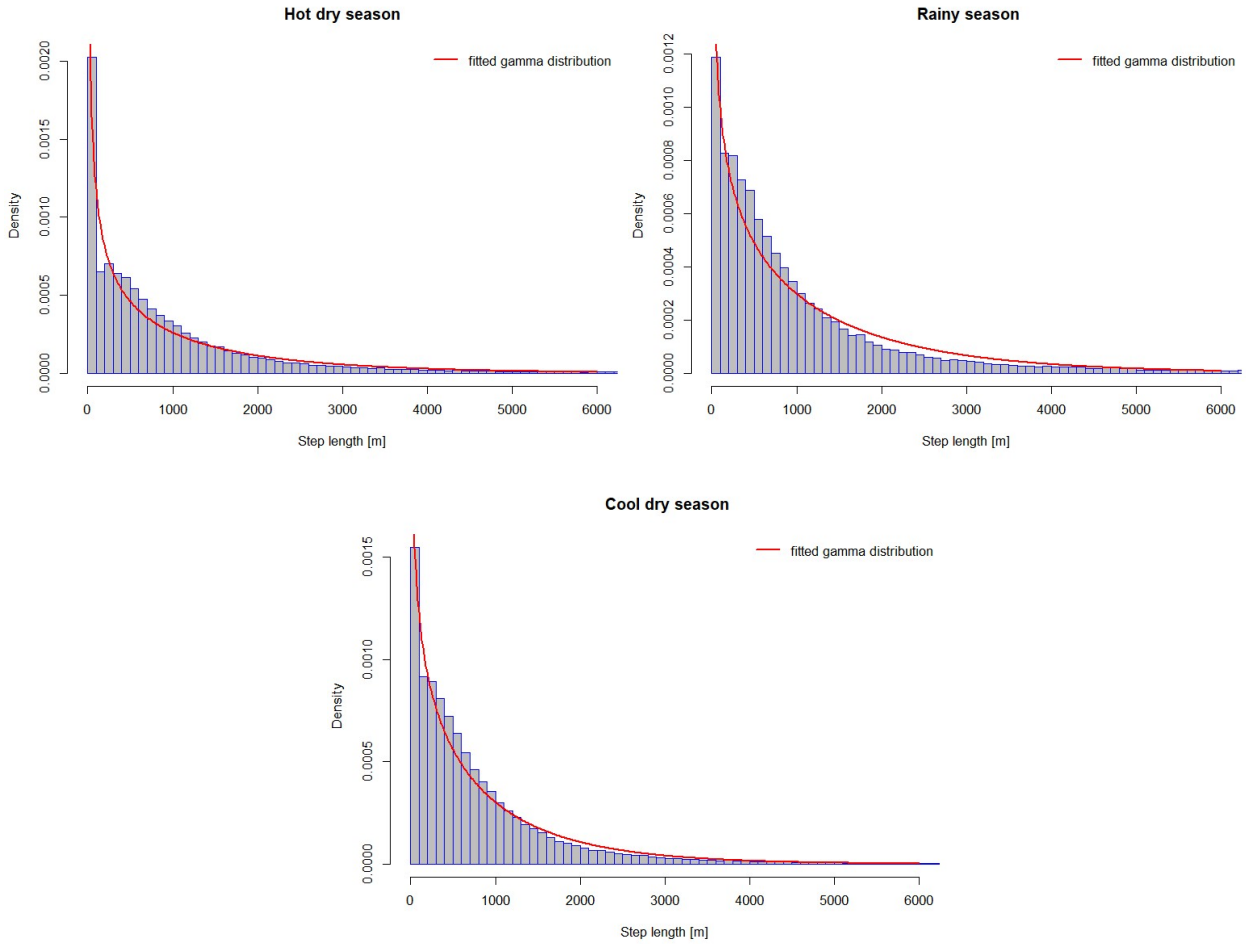

Figure 17: The empirical step length distribution of each season, overlaid by their parametric estimation shown by the red line.

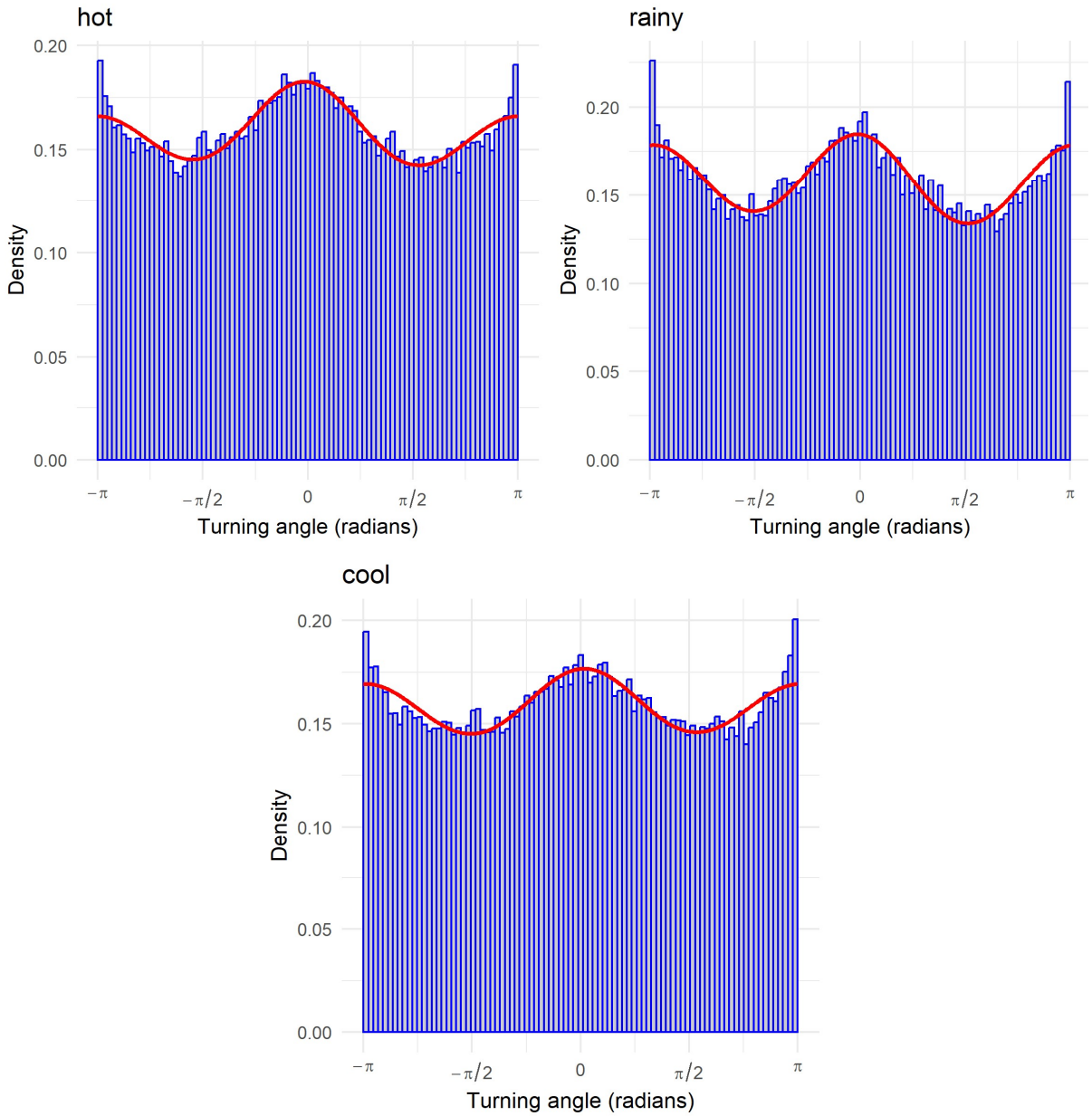

Figure 18: The empirical distribution of relative angles for each season (hot dry, rainy, and cool dry respectively); the red line shows the approximation of this empirical distribution by a parametric bimodal von Mises distribution.

3.3. Vegetation cover

Calculated with the MSAVI, based on the Landsat imagery.

The results of each season exceed the significance threshold. We define it as $p \leq 0.05$ (Majaliwa et al., 2022). The quadratic effects are also significant in every season.

Vegetation density has the weakest impact on *Oryx dammah* habitat selection during the rainy season (Figure 19). The impact becomes stronger during the cool dry season, but it reaches a

maximum and the selection becomes weaker for places with particularly dense vegetation. The hot dry season shows by far the strongest correlation. The effect is so strong during this season that we need to express this variable on a logarithmic scale to represent it correctly.

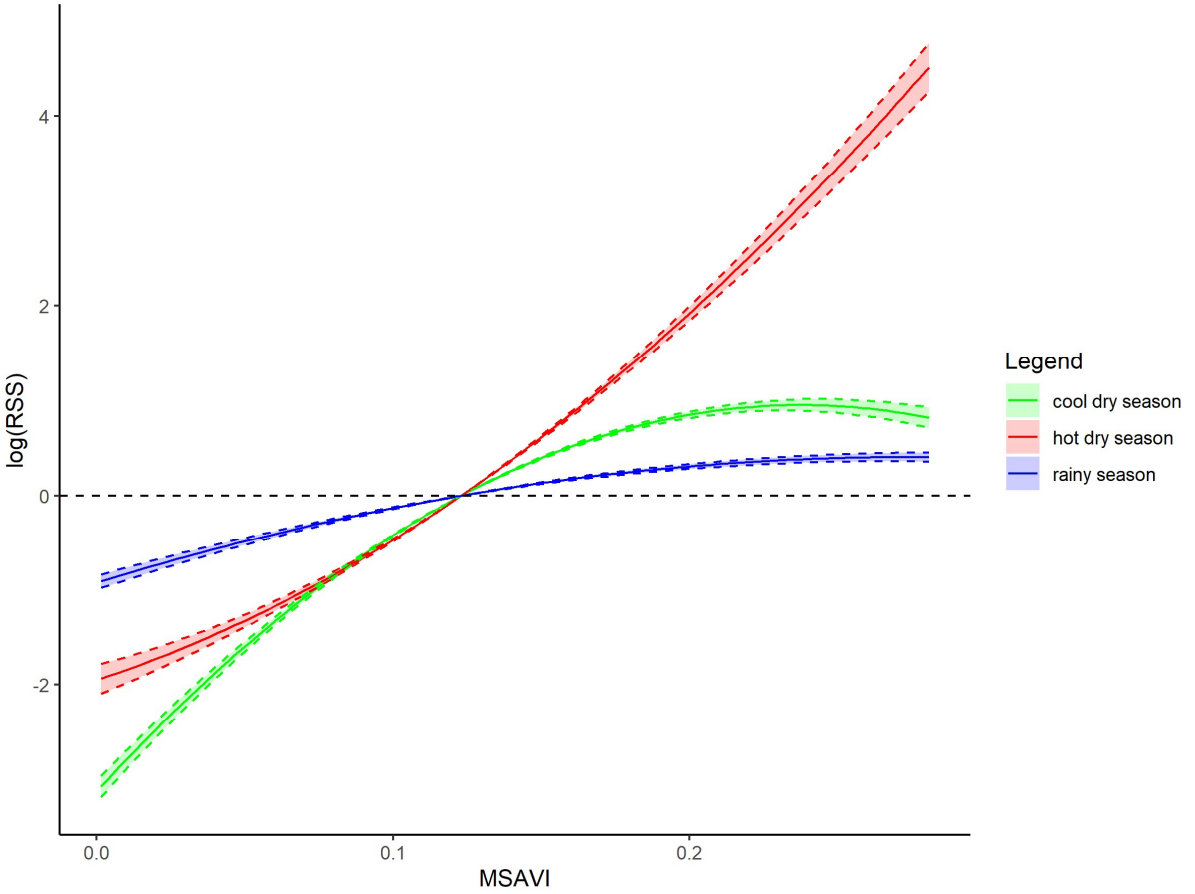

Figure 19: Indicates the variation in habitat selection by Oryx dammah *in relation to changes in vegetation cover intensity during the cool dry season (green), hot dry season (red), and the rainy season (blue). Shaded areas represent the 95% confidence intervals.*

3.4. Woody vegetation density

All three seasons reach the significance threshold, and their quadratic effects are also statistically significant.

In all cases, the presence of *Oryx dammah* is positively correlated with high values of the indicator (Figure 20). However, this trend is much stronger during the hot dry season than during the cool dry season or the rainy season, the latter being the one with the weakest effect.

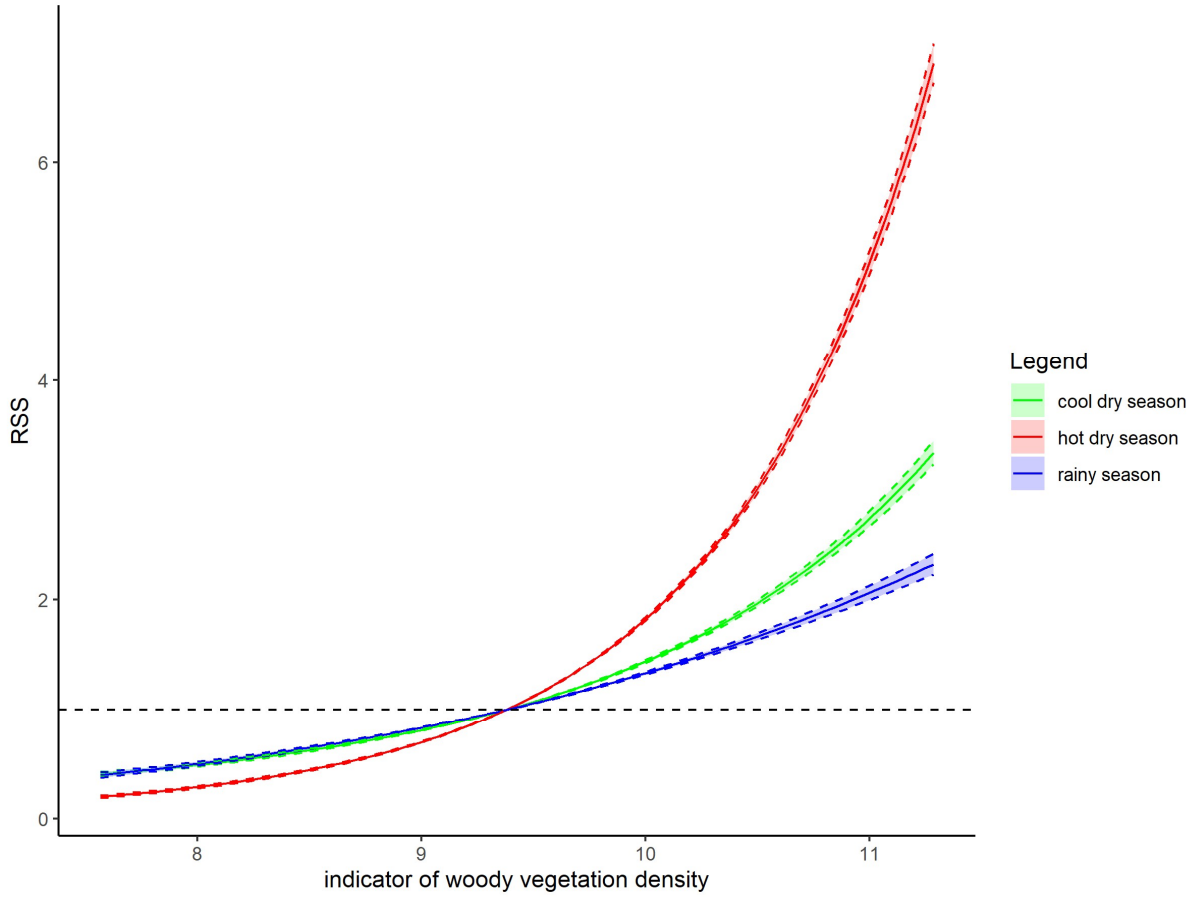

Figure 20: Indicates the variation in habitat selection by Oryx dammah *in relation to changes in woody vegetation cover intensity during the cool dry season (green), hot dry season (red), and the rainy season (blue). Shaded areas define the 95% confidence intervals.*

3.5. Wildfires factor

Based on the indicator of the sum of fires during the 2004 to 2023 period.

The data of all seasons are statistically significant, although this trend is weak for the cool dry season ($p = 0.427$).

The presence of *Oryx dammah* is negatively correlated with fire occurrence during both the hot dry and rainy seasons, but the correlation is positive during the cool dry season (Figure 21).

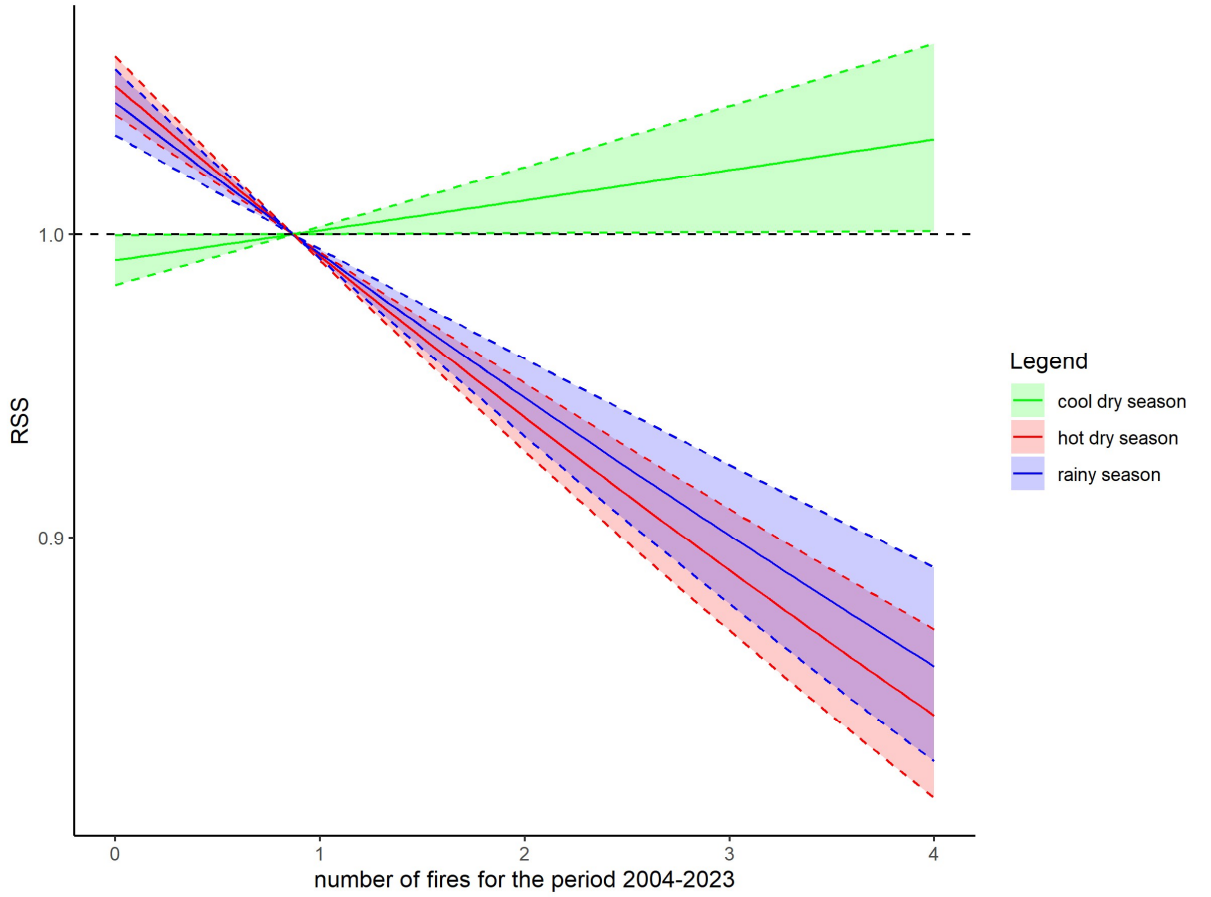

Figure 21: Indicates the variation in habitat selection by Oryx dammah *in relation with the number of fires reported over the period from 2004 to 2023 during the hot dry season (red) and the rainy season (blue). The cool dry season is not significant. Shaded areas define the 95% confidence intervals.*

3.6. Topography factor

It is based on the topographic convergence index (TCI) with a 210×210 meter grid.

All three seasons show statistically significant results. However, the quadratic nature of the hot dry and cool dry seasons is not significantly supported. In fact, the quadratic term of the hot dry season is $p = 0.0557$, and the one of the cool dry season is $p = 0.8486$.

The hot dry and cool dry seasons display a linear or nearly linear negative relationship with high TCI (Figure 22). The rainy season has a quadratic pattern, where very high values of TCI are preferred to medium-high values.

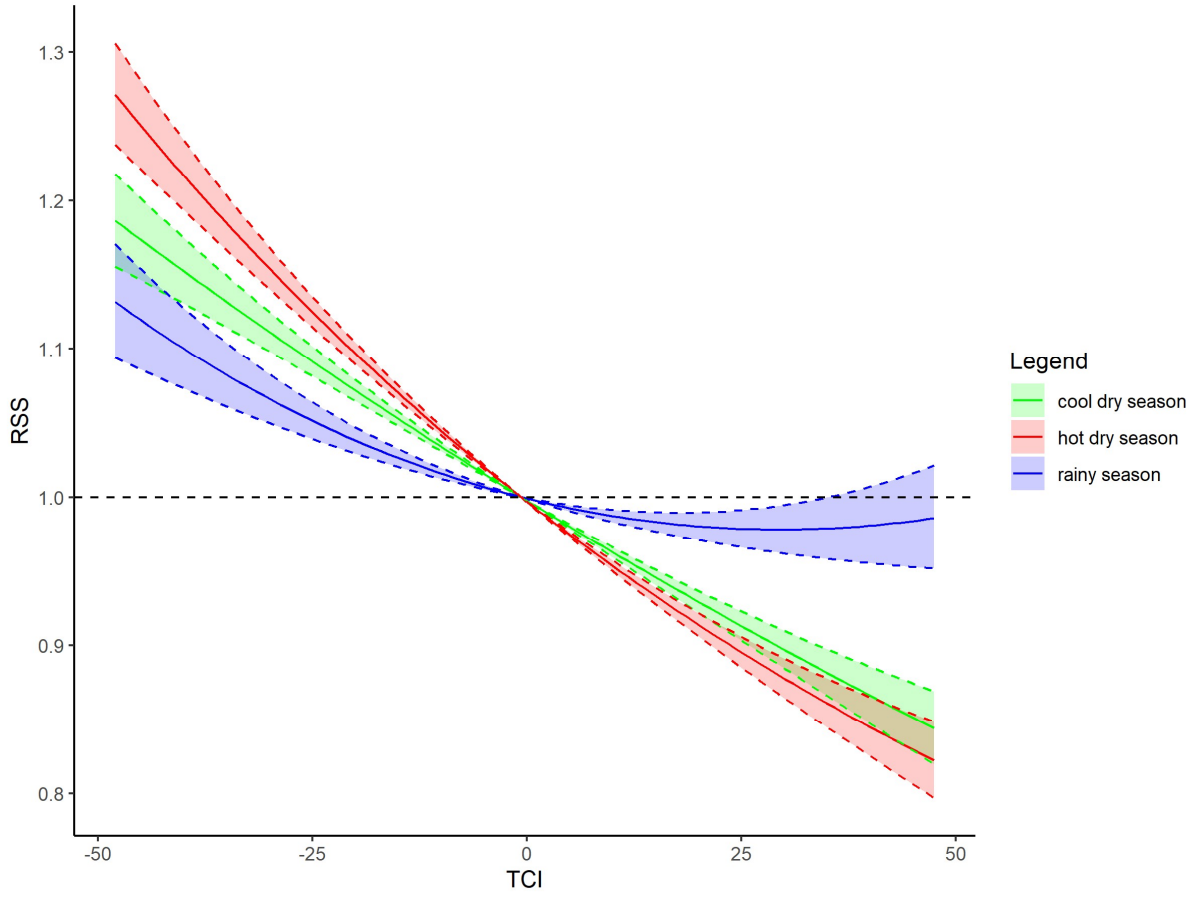

Figure 22: Indicates the variation in habitat selection by Oryx dammah *in relation with changes in TCI (210 × 210 m neighborhood) during the cool dry season (green), hot dry season (red), and the rainy season (blue). Shaded areas define the 95% confidence intervals.*

3.7. Human activity factor

The presence of *Oryx dammah* is negatively correlated with human activity during both hot dry and rainy seasons. The negative relationship is stronger in the rainy season than in the hot dry season (Figure 23). The cool dry season shows only a slight negative correlation, which does not reach the statistical significance threshold. The p-value for the cool dry season is far above the significance threshold, with $p = 0.7656$.

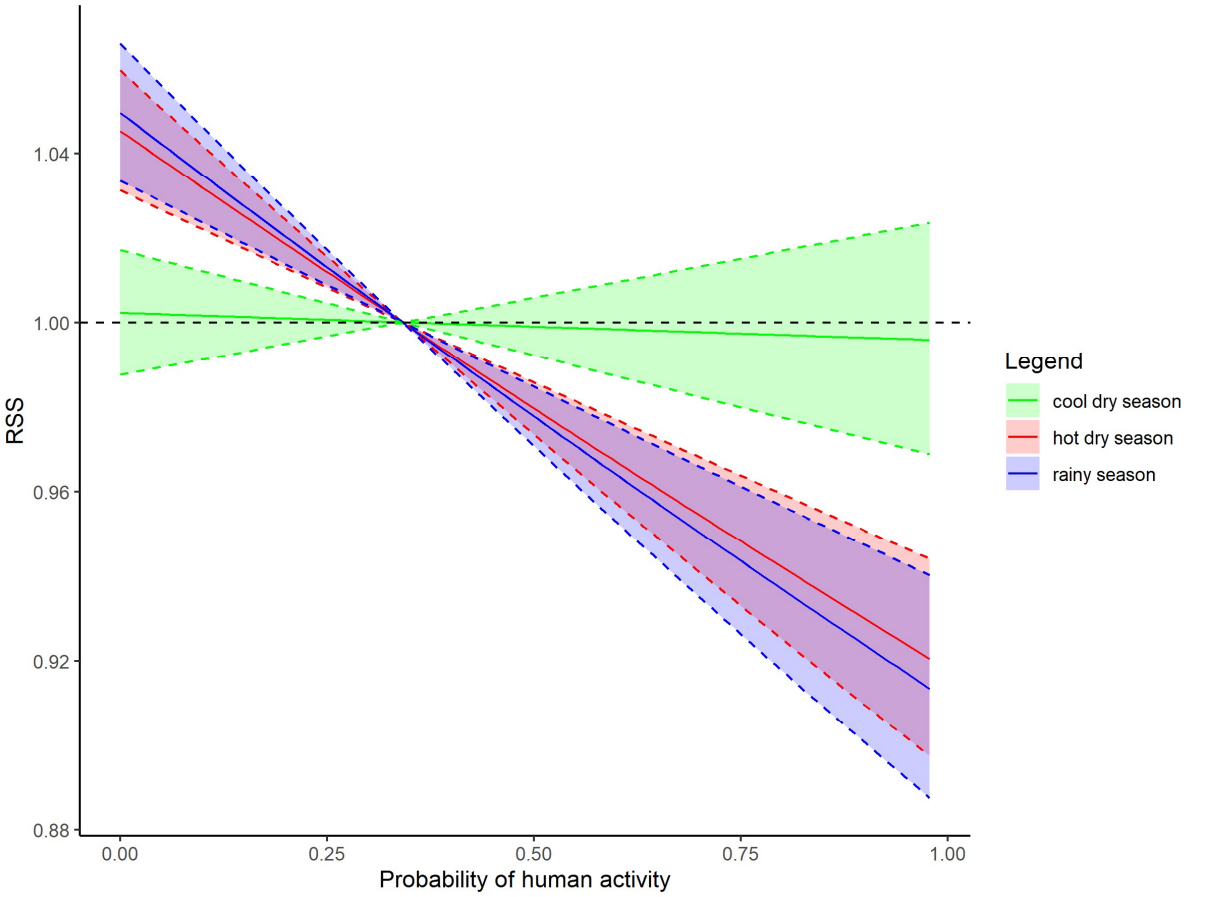

Figure 23: Indicates the variation in habitat selection by Oryx dammah *in relation with the presence of human activity during the hot dry season (red), and the rainy season (blue). The cool dry season is not significant. Shaded areas define the 95% confidence intervals.*

4. Discussion

4.1. Movement statistics

Although our study has no hypothesis about the movement statistics, they are secondary outputs that might require a few comments.

The step length seems to follow a Gamma distribution quite well, which is the expected form for this statistic (Signer, Fieberg, et al., 2024).

When we compare the parametric Gamma distribution across seasons, we only see very small seasonal variations (Figure 24). This contrasts with the empirical distributions, which show significant differences, especially in the smallest step length values. Indeed, they account for more than 2% of the step lengths during the hot dry season. This proportion is significantly smaller during the cool dry season and even more during the rainy season, reaching a minimum under 1.2% of all step lengths during this period. This feature is not capture by the parametric distributions. We must emphasize that the Gamma distributions are only approximations, and this is a clear case where inaccuracies are introduced by the conversion to parametric curves. They are useful to generate plausible available steps and give a broad idea of the behavior of the studied animals, but they seem unsuitable for drawing detailed conclusions.

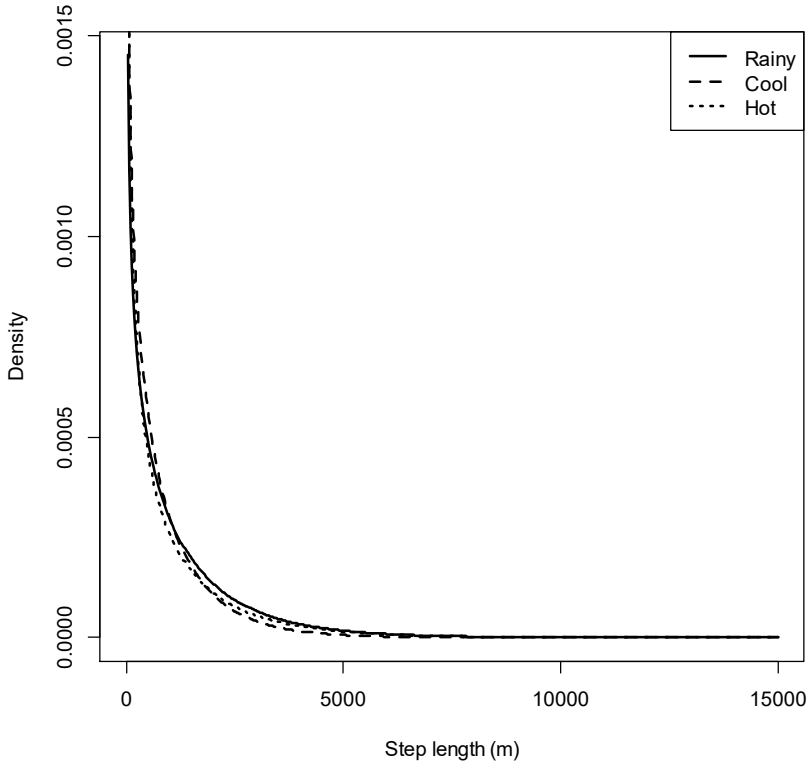

Figure 24: Comparison of the parametric estimation of the step length across seasons.

We encounter the same problem of approximation with the relative turning angle. This time,

the main characteristic that seems to be underrepresented by the parametric functions is the second local maximum located near π/-π. Indeed, the empirical distribution shows a few values of very high frequency in that area, that are considerably smoothed out in the parametric curve.

Despite the lack of accuracy when we investigate the details, our study highlights an important fact about the turning angle distribution. *Oryx dammah* does not exhibit the classical von Mises distribution used by default in ISSFs (Signer, Fieberg, et al., 2024). Instead, the species shows a bimodal distribution, caused by the second local maximum. This feature is notable, but not unique. For example, some populations of wolves or turtles have been identified as having such movement patterns (Hillen et al., 2017). In the case of the wolves, they are in northern Alberta and are limited in their movements by the presence of seismic lines that act as natural barriers. The turtles live around Ascension Island, and this location serves as an attractor, creating a bias in the movement statistics.

However, no significant ecological corridor or barrier is reported in the OROAFR that could justify such restrictions in movement. The only place that could represent a particular site for *Oryx dammah* is the release site. However, it does not play an ecological role similarly to the island does for the turtles. At this stage, those examples do not provide us with a plausible explanation for the specific movement pattern observed in our study. Further studies could investigate this question, but for the moment, the most important point to retain is that the turning angle distribution of *Oryx dammah* has a distinct bimodal pattern.

4.2. Vegetation

Our results confirm that vegetation is the main driver of habitat selection for *Oryx dammah*.

Our seasonal division adds nuance to the results of Majaliwa et al. (2022). Indeed, this study demonstrated that very high levels of vegetation density reduce habitat selection during the dry season (Majaliwa et al., 2022). However, our results show that this effect is only present during the early part of this period, namely the cool dry season, and is no longer observed during the hot dry season.

The intense search for areas with dense vegetation during the hot dry season can be explained by their rarity. Indeed, the study site mostly exhibits low vegetation densities during the dry months (Figure 25). In this context, rich patches of vegetation are important enough for *Oryx dammah* to strongly select them. Conversely, during the wet months, the range of MSAVI greatly widens, increasing access to areas with high vegetation density. The food stress is significantly reduced, which might cause a lower habitat selection for specific areas with high vegetation density.

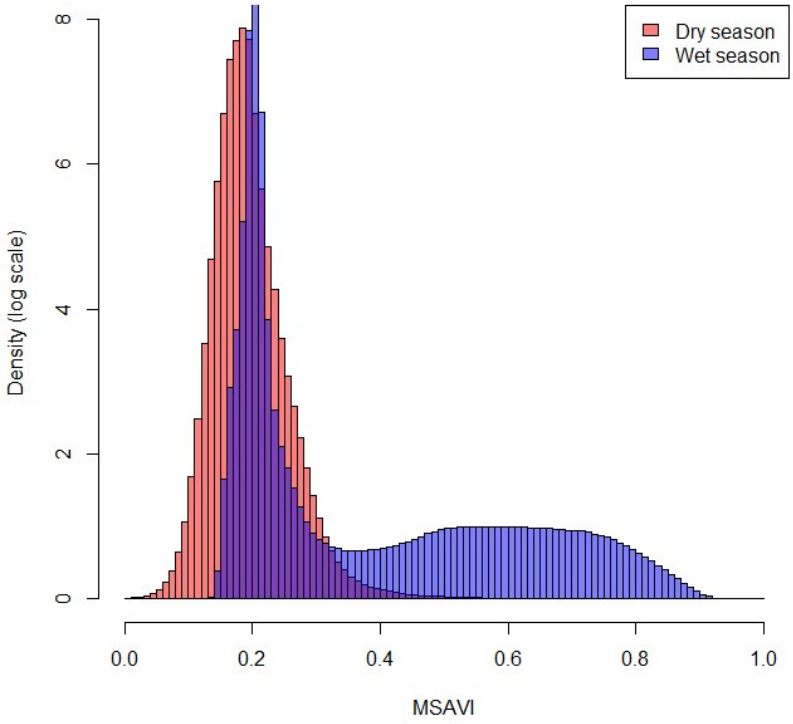

Figure 25: Distribution of the MSAVI values of the pixels of two composite images representing the mean MSAVI of the dry season (January – June) and the wet season (July – December) over the 2010-2019 period. These two raster layers, provided by the Scimitar Oryx Reintroduction Project, have a resolution of 30 m.

A higher woody vegetation density is also preferred in all seasons, but the intensity of the selection for this habitat is particularly strong during the hot dry season. This can readily be explained by the greater heat stress during this period, which causes a stronger selection for areas that provide shade – woody vegetation being an indicator of such areas.

The best indicator of wildfires is the cumulative sum over 20 years. This is unexpected, because we initially expected that the main impact of wildfires would be short term, better represented by fire occurrence during the previous year. However, the AICc results show that the difference in explanatory power between these two indicators is very small. We also suspect that technical limitations affected the quality of the "previous year" indicator, making it suboptimal (more details can be found in section 4.3.3). Consequently, we must remain cautious about the conclusions we can draw regarding which indicator is the best and why. Our recommendation is to create a fully functional "previous year" layer, as there is a significant probability that this improved version would outperform the 20-year sum layer in explanatory power.

The presence of *Oryx dammah* has a negative correlation with the presence of wildfires during the hot dry and rainy seasons, but this effect becomes slightly positive during the cool dry season. In fact, most fires occur during this season (Table 10). This indicates that in most cases, when an individual goes into a burned area during the cool dry season, this place was probably burned a few weeks or months ago. This is less likely for the hot dry and rainy seasons, as fires

generally do not occur during these periods. Consequently, we can assume that our results are consistent with the hypothesis of Lox (2024), which suggests that *Oryx dammah* is attracted to burned places for a period after the fire, to eat specific vegetation species. This fact would also be in line with similar observations made on other ungulates in Kenya (Odadi et al., 2017).

Table 10: Number of pixels burned per season over the period 2016-2022 in the study site. The third column shows the mean monthly number of burned pixels for each respective season. Data were obtained from the MODIS collection MODIS/006/MCD64A1.

	Number of burned pixels	mean burned pixels per day
Hot dry season	6628	55.23
rainy season	3634	44.32
Cool dry season	155227	952.31

However, the negative impact of fires on the habitat selection during the other two seasons has not been demonstrated until now and remains puzzling. We can imagine that after the destruction caused by fire, an ecological succession occurs, with the first plants growing on the burned soil being preferred by *Oryx dammah*, while the species that dominate in later stages are less preferred. However, we currently lack evidence to support this hypothesis. Further studies could test this idea and explore other possible causes of this negative relationship, which may in fact represent the primary way in which fires influence *Oryx dammah*'s habitat selection.

4.1. Topography

The fact that the 210 × 210 m neighborhood size indicators return better models than the 90 × 90 m indicators suggests that *Oryx dammah*'s behavior is more influenced by medium-scale topographic features than by fine-scale ones. However, further research is needed to determine the topographic resolution with the highest explanatory power for the habitat selection of *Oryx dammah*.

Our results show that the TCI has a higher explanatory power on the habitat selection of *Oryx dammah* than the TPI. This suggests that simple topographic irregularity is not the factor that impacts *Oryx dammah*'s habitat selection the most. The TCI is a measure of a surface's ability to accumulate water from surrounding areas. We could imagine that it would correlate with higher vegetation cover, but our correlation tests do not support this hypothesis (see section A.13.2.). If the TCI does not influence *Oryx dammah*'s habitat selection through either pure topographic structure or vegetation productivity, its ecological role remains unclear.

Since a minimal TCI value indicates a place from which water flows in every direction, we must consider it to be a local elevation maximum. The opposite is also true, a high TCI value represents a place that accumulates the water in its entire direct environment, which describes a topographic depression. However, this interpretation would mean that *Oryx dammah* avoids depressions, especially during the hottest months, which contradicts previous findings (Beyouli & Neffati, 2016; Kingdon & Hoffmann, 2013). We also notice that the results of Majaliwa (2022) indicate that high-elevation places are more strongly selected than low-lying areas during the dry season, while the opposite is observed during the wet season. This appears to be at odds with Kingdon & Hoffmann (2013) as well, but partially consistent with our own results. However, our findings for the rainy season are not aligned with those of Majaliwa (2022).

Further research is needed to know which indicator – or combination of indicators – is the most efficient to explain the habitat selection of *Oryx dammah*. If the TCI appears as a useful factor, more information on its ecological role should be gathered to better justify the observations made in this study.

4.2. Human activity

The general trend of the result is that *Oryx dammah* avoids places with greater human activity. Consequently, our hypothesis is only partially supported. *Oryx dammah* is indeed more likely to go to places where human activity take place during the hot dry season than during the rainy season. However, the difference between these two seasons, although predicted by our hypothesis, is relatively minor. This indicates that the causes under consideration (more salts and less human activity) probably only play a limited role, or that they are counterbalanced by opposing factors that were not identified in this study.

The absence of significant results for the cool dry season is also puzzling. This would mean that during this season, *Oryx dammah* is virtually unresponsive to human presence. However, this result is inconsistent with the theoretical framework presented in our introduction, particularly because of the competition with livestock, which is one of the most pressing conservation concerns. No clear explanation currently accounts for this unexpected relationship between *Oryx dammah* and humans and their livestock during this season despite the strongly supported conclusions found in scientific literature.

Because a sudden change in tolerance for human and livestock presence seems unlikely, we must explore alternative explanations. One of them is a seasonal reduction in human presence during the cool dry season. We have no quantitative data on seasonal variations in human presence in those places, although its existence has been acknowledged (Wacher et al., 2022). Consequently, our indicator fails to account for this aspect. However, even within the limited data available, indications suggest a peak in human presence during the rainy season, but no substantial difference between the hot dry and cool dry seasons (Wacher et al., 2022; Worgue

& Yamtibaye, 2023). This explanation does not seem to account for all the data, but information about the monthly distribution of human activity could reveal nuances in the coarse and incomplete data we have, which makes it the most promising hypothesis. A dedicated census quantifying the seasonal movements of pastoralists in the OROAFR could help fill this knowledge gap.

4.3. Methodological limitations

4.3.1. Irregularities between seasons in the amount of data

As revealed by the seasonal table (Table 5), our data do not evenly cover all seasons of the studied period. In the most extreme case, only five individuals were monitored during the rainy and cool dry seasons of 2019, whereas 40 were during the rainy season of 2023 and 38 during the cool dry season of 2023.

This means that the conditions observed in 2019 (e.g. specific climate features) and the ecological response of *Oryx dammah* will have less weight in the analysis compared to those from 2023. While most of the dataset is somewhat more balanced, a degree of imbalance remains.

This issue could be mitigated by introducing a weighting scheme to correct for seasonal sampling bias.

4.3.2. Udapte of turning angle

Usually, it is recommended to update the movement statistics parameters after running an ISSF (Fieberg et al., 2021). Indeed, the initial calculation of the parametric distributions does not exclude the influence of the habitat selection on distributions, making it only an approximation of a true selection-free movement kernel (Forester et al., 2009).

Due to the bimodal shape of our turning angle distribution, the classical tools of `amt` are not suitable to perform this operation (Signer, Smith, et al., 2024). Further research is needed to identify suitable functions for this type of update on bimodal von Mises distributions. However, it goes beyond the scope of this study. Indeed, this update does not affect the ISSF model itself (Signer, Smith, et al., 2024). It is essentially useful for additional analyses such as redistribution kernels (Signer, Fieberg, et al., 2024). Therefore, it is not required for studies similar in nature to ours (Majaliwa et al., 2022).

It is still important to note that our movement statistics have not been updated, and are therefore not suitable for further use in their current form. Additional processing would be required to bring them up to date.

4.3.3. Accuracy of the indicators

The validity of our results depends greatly on the accuracy of our indicators. However, these indicators are only approximations of real-world conditions, and they could have been improved in several ways. For the MSAVI, the Landsat satellite images often present a time lag of a few days compared to the GPS data, which can cause minor inaccuracies in the resulting values. The indicator of wildfires in the previous year is also imperfect: for data gathered in January, it reflects fires from the past 12 months, while for data in December, it covers fires from 11 to 23 months prior. This creates irregular temporal coverage that sometimes fails to produce significant results, which may explain why this indicator performed worse than the 20-year cumulative sum of wildfires.

The woody vegetation indicator is based on a sum of MSAVI values from 2021 to 2023. Even though this type of vegetation is relatively stable over time, we can reasonably assume that changes occurred between 2017 and 2021, making it an imperfect proxy.
The spatial resolution of most of our indicators is 30 m, which is generally insufficient to capture finer landscape details, especially for vegetation and topography. Wildfire data has a resolution of 1 km, while MSAVI from MODIS and D-MSAVI are at a 500 m resolution, which remains relatively coarse.

We defined the grid size of topographic indicators through an exploratory process, but the truly optimal scale is likely different.

The indicator of human presence is based on various features, ranging from natural water points to human settlements, but these are all assigned equal weight in their contribution to human presence. Additionally, the data are incomplete due to limitations in site monitoring. Moreover, this indicator does not account for seasonal variation.

4.3.4. Interaction assumption

To limit the number of models included in our analysis, we assumed that movement statistics do not interact with environmental factors outside of habitat selection. In other words, we assume that *Oryx dammah* exhibits the same movement behavior (in terms of step length and turning angle distributions) regardless of the environment in which it moves. However, this assumption is not supported by direct evidence. It is conceivable, for instance, that *Oryx dammah* walks shorter distances when traversing rugged terrain. In such a case, step length would be influenced by topographic conditions.

Other models of ISSF can account for such effects (Fieberg et al., 2021). This is a whole research avenue that still remains to explore to gain a more comprehensive understanding of *Oryx dammah*'s movement patterns.

5. Conclusion

In this study, we analyzed the habitat selection of *Oryx dammah* in the Ouadi Rimé–Ouadi Achim Faunal Reserve using integrated step-selection functions. In particular, we studied the effects of human presence, topography, and vegetation on the distribution of *Oryx dammah*. We used a new seasonal framework featuring three seasons to better account for variations in habitat selection and movement throughout the year.

We tested multiple indicators for each environmental factor to obtain a more accurate understanding of which combinations of variables are effective predictors of *Oryx dammah*'s habitat selection.

Our results show that *Oryx dammah* is less affected by microtopography than by medium-scale topographic variations, and that the Topographic Convergence Index is a promising indicator that warrants further research.

We also demonstrate that vegetation productivity is the main factor impacting the habitat selection of the species. The use of an indicator of woody vegetation is effective as a complement to vegetation productivity.

Finally, we highlight the impact of wildfires as a negative factor for habitat selection during the hot dry and rainy seasons while identifying a positive impact during the cool dry season.

This study represents a step towards further research, that could take many forms. It is possible to enrich the analysis with more environmental factors such as temperature or land cover. It is also possible to look for better indicators. For example, topography can be represented by various indicators or combinations of indicators: elevation, slope, roughness, TPI, and TCI with various neighborhood sizes, etc. Establishing which combination of factors is the most efficient for each of our environmental factors could help develop a more realistic model.

Using these data to produce a habitat suitability map over certain areas could help identify which areas are potential sites of reintroduction or highlight crucial zones for the species, contributing to its conservation.

In conclusion, this study explores a field of research that is rich in opportunities for additional research to help the conservation of *Oryx dammah*.

6. Acknowledgements

We want to warmly thank Clara Zemp for her supervision, guidance and advice, and sonPavla Hejcmanová for her constant advice and mentorship. We are also grateful to Caleb Ngaba Waye Taroum for his coaching and encouragement, as well as his advice and practical help on many occasions. We thank Gabriel Marcacci for his involvement in the project and his good advice. We are also very thankful to Katherine Mertes, who shared her expertise and advice several times, as well as her generous sharing of useful resources.

We also thank Sahara Conservation and the Smithsonian Institution for the precious data they shared with us, and the Czech University of Life Sciences for hosting us for a useful internship.

7. Bibliography

Amatulli, G., McInerney, D., Sethi, T., Strobl, P., & Domisch, S. (2020). Geomorpho90m, empirical evaluation and accuracy assessment of global high-resolution geomorphometric layers. *Scientific Data*, *7*, 162. https://doi.org/10.1038/s41597-020-0479-6

Avgar, T., Lele, S. R., Keim, J. L., & Boyce, M. S. (2017). Relative Selection Strength: Quantifying effect size in habitat- and step-selection inference. *Ecology and Evolution*, *7*(14), 5322–5330. https://doi.org/10.1002/ece3.3122

Avgar, T., Potts, J. R., Lewis, M. A., & Boyce, M. S. (2016). Integrated step selection analysis: Bridging the gap between resource selection and animal movement. *Methods in Ecology and Evolution*, *7*(5), 619–630. https://doi.org/10.1111/2041-210X.12528

Beyouli, H. chedli T., & Neffati, M. (2016). Vegetative habitat selection of Scimitar horned oryx (Oryx dammah) in Bouhedma National Park, Southern Tunisia. *Journal of King Saud University – Science*, *28*(3), 261–267. https://doi.org/10.1016/j.jksus.2016.04.003

Chan, A. N., Leimgruber, P., Williams, C., Shwe, N. M., Aung, S. S., Lwin, N., Oo, Z. M., Chit, A. M., & Wittemyer, G. (2024). Individual variation in habitat selection behavior of Asian elephants in a human-wildland interface. *Global Ecology and Conservation*, *53*, e03025. https://doi.org/10.1016/j.gecco.2024.e03025

Chardonnet, P., Newby, J., Whitemore, S., & Martin, J. (2024). Scimitar-horned Oryx: Back to the Wild. *Gnusletter*, *Special Issue 2*, 26.

Clayton, W. D. (1972). *Flora of West Tropical Africa*. Plants of the World Online. http://powo.science.kew.org/taxon/urn:lsid:ipni.org:names:421057-1

Collelo, T. (1990). *Chad: A country study* (2nd Edition). Library of Congress. Federal Research Division. https://www.loc.gov/item/89600373/

Cooke. (2023). IUCN Red List of Threatened Species: Oryx dammah. *IUCN Red List of Threatened Species*. https://www.iucnredlist.org/en

Deenick, K. (2021, February 16). *Topographic Position Index (TPI) – Eat.Sleep.Code.Ski*. https://blogs.ubc.ca/tdeenik/2021/02/16/topographic-position-index-tpi/

Dormann, C. F., Elith, J., Bacher, S., Buchmann, C., Carl, G., Carré, G., Marquéz, J. R. G., Gruber, B., Lafourcade, B., Leitão, P. J., Münkemüller, T., McClean, C., Osborne, P. E., Reineking, B., Schröder, B., Skidmore, A. K., Zurell, D., & Lautenbach, S. (2013). Collinearity: A review of methods to deal with it and a simulation study evaluating their performance. *Ecography*, *36*(1), 27–46. https://doi.org/10.1111/j.1600-0587.2012.07348.x

Duchesne, T., Fortin, D., & Rivest, L.-P. (2015). Equivalence between Step Selection Functions

and Biased Correlated Random Walks for Statistical Inference on Animal Movement. *PLOS ONE*, *10*(4), e0122947. https://doi.org/10.1371/journal.pone.0122947

Earth Engine Data Catalog. (n.d.). *NASA SRTM Digital Elevation 30m*. Google for Developers. Retrieved July 12, 2025, from https://developers.google.com/earth-engine/datasets/catalog/USGS_SRTMGL1_003

Fieberg, J., Signer, J., Smith, B., & Avgar, T. (2021). A 'How to' guide for interpreting parameters in habitat-selection analyses. *Journal of Animal Ecology*, *90*(5), 1027–1043. https://doi.org/10.1111/1365-2656.13441

Florko, K. R. N., Togunov, R. R., Gryba, R., Sidrow, E., Ferguson, S. H., Yurkowski, D. J., & Auger-Méthé, M. (2025). An introduction to statistical models used to characterize species-habitat associations with animal movement data. *Movement Ecology*, *13*(1), 27. https://doi.org/10.1186/s40462-025-00549-2

Forester, J. D., Im, H. K., & Rathouz, P. J. (2009). Accounting for animal movement in estimation of resource selection functions: Sampling and data analysis. *Ecology*, *90*(12), 3554–3565. https://doi.org/10.1890/08-0874.1

Giglio, L., & Justice, C. (2015a). *MOD14A1 MODIS/Terra Thermal Anomalies/Fire Daily L3 Global 1km SIN Grid V006* [Dataset]. NASA Land Processes Distributed Active Archive Center. https://doi.org/10.5067/MODIS/MOD14A1.006

Giglio, L., & Justice, C. (2015b). *MYD14A1 MODIS/Aqua Thermal Anomalies/Fire Daily L3 Global 1km SIN Grid V006* [Dataset]. NASA Land Processes Distributed Active Archive Center. https://doi.org/10.5067/MODIS/MYD14A1.006

Gilbert, T. (2004). The Reintroduction of Scimitar-horned Oryx to Senegal. In *The Biology, Husbandry and Conservation of Scimitar-horned Oryx (Oryx dammah)* (pp. 82–83). MARWELL PRESERVATION TRUST.

Gilbert, T., & Woodfine, T. (2004). *The biology, husbandry and conservation of scimitar-horned oryx (Oryx dammah)*. ResearchGate. https://www.researchgate.net/publication/301295360_The_biology_husbandry_and_conservation_of_scimitar-horned_oryx_Oryx_dammah

Gillet, H. (1965). L'Oryx algazelle et l'Addax au Tchad. *Revue d'écologie*, *3*, 257–272.

Gordon, I. J., & Gill, J. Paul. (1993). Reintroduction of Scimitar-horned oryx Oryx dammah to Bou-Hedma National Park, Tunisia. *International Zoo Yearbook*, *32*(1), 69–73. https://doi.org/10.1111/j.1748-1090.1993.tb03517.x

Gu, Z., Chen, J., Shi, P., & Xu, M. (2007). Correlation analysis of Normalized Different Vegetation Index (NDVI) difference series and climate variables in the Xilingole steppe, China from 1983 to 1999. *Frontiers of Biology in China*, *2*(2), 218–228.

https://doi.org/10.1007/s11515-007-0033-3

Hillen, T., J. Painter, K., C. Swan, A., D. Murtha, A., 1. University of Alberta, Centre for Mathematical Biology, Edmonton, Alberta, T6G2G1, Canada, 2. Department of Mathematics, Heriot-Watt University, Edinburgh, EH14 4AS, UK, & 3. Cross Cancer Institute, 11560-University Ave NW, Edmonton, Alberta, T6G 1Z2, Canada. (2017). Moments of von mises and fisher distributions and applications. *Mathematical Biosciences and Engineering*, *14*(3), 673–694. https://doi.org/10.3934/mbe.2017038

Hofmann, D. D., Cozzi, G., & Fieberg, J. (2024). Methods for implementing integrated step-selection functions with incomplete data. *Movement Ecology*, *12*(1), 37. https://doi.org/10.1186/s40462-024-00476-8

IUCN SSC Antelope Specialist Group. (2023). *Oryx dammah. The IUCN Red List of Threatened Species 2023*. https://dx.doi.org/10.2305/IUCN.UK.2023- 1.RLTS.T15568A197393805.en

Jiang, Q. (2009). *ON FITTING A MIXTURE OF TWO VON MISES DISTRIBUTIONS, WITH APPLICATIONS*. https://www.stat.sfu.ca/content/dam/sfu/stat/alumnitheses/MiscellaniousTheses/Jiang-2009.pdf

Kingdon, J. (2015). *The kingdon field guide to african mammals* (2nd Edition). Bloomsbury.

Kingdon, J., & Hoffmann, M. (2013). *Pigs, Hippopotamuses, chevrotains, giraffes, deer and bovids* (Bloomsbury).

LAADS. (n.d.-a). *MOD09GQ MODIS/Terra Surface Reflectance Daily L2G Global 250m SIN Grid V006* [Dataset]. Retrieved June 24, 2025, from https://lpdaac.usgs.gov/products/mod09gqv006/

LAADS. (n.d.-b). *MYD09GQ MODIS/Aqua Surface Reflectance Daily L2G Global 250m SIN Grid V006* [Dataset]. Retrieved June 24, 2025, from https://lpdaac.usgs.gov/products/myd09gqv006/

Landsat Acquisition Tool. (n.d.). Retrieved December 8, 2024, from https://landsat.usgs.gov/landsat_acq

Leimgruber, P., Songsasen, N., Stabach, J. A., Horning, M., Reed, D., Buk, T., Harwood, A., Layman, L., Mathews, C., Vance, M., Marinari, P., Helmick, K. E., Delaski, K. M., Ware, L. H., Jones, J. C., Silva, J. L. P., Laske, T. G., & Moraes, R. N. (2023). Providing baseline data for conservation–Heart rate monitoring in captive scimitar-horned oryx. *Frontiers in Physiology*, *14*, 1079008. https://doi.org/10.3389/fphys.2023.1079008

Liu, T., Yan, H., & Zhai, L. (2015). Extract relevant features from DEM for groundwater potential mapping. *Remote Sensing of Spatial Information Sciences*, *60*. https://123dok.com/document/q2e828rq-isprsarchives-xl-w.html

Louhichi, M., Khorchani, T., Eifler, D., Eifler, M., Orton, M., Dadi, K., Zaydi, A., Jarray, M., & Chammem, M. (2024). Modelling habitat suitability of reintroduced scimitar-horned oryx (Oryx dammah) in Sidi Toui National Park, Tunisia. *JOURNAL OF OASIS AGRICULTURE AND SUSTAINABLE DEVELOPMENT*, *6*(02), Article 02. https://doi.org/10.56027/JOASD.162024

Lox, S. (2024). Vegetation and bush fires: An interdependent relationship. *Sandscript*, *32*, 14–15.

Majaliwa, M. M., Hughey, L. F., Stabach, J. A., Songer, M., Whyle, K., Alhashmi, A. E. A., Al Remeithi, M., Pusey, R., Chaibo, H. A., Ngari Walsoumon, A., Hassan Hatcha, M., Wacher, T., Ngaba, C., Newby, J., Leimgruber, P., & Mertes, K. (2022). Experience and social factors influence movement and habitat selection in scimitar-horned oryx (Oryx dammah) reintroduced into Chad. *Movement Ecology*, *10*(1), 47. https://doi.org/10.1186/s40462-022-00348-z

Mertes, K., Stabach, J. A., Songer, M., Wacher, T., Newby, J., Chuven, J., Al Dhaheri, S., Leimgruber, P., & Monfort, S. (2019). Management Background and Release Conditions Structure Post-release Movements in Reintroduced Ungulates. *Frontiers in Ecology and Evolution*, *7*. https://doi.org/10.3389/fevo.2019.00470

Meyer, A. (2016). *25 Scimitar-Horned Oryx To Be Reintroduced to the Wild in Chad*. Smithsonian Institution. https://www.si.edu/newsdesk/releases/25-scimitar-horned-oryx-be-reintroduced-wild-chad

Moreno, E., Sane, A., Benzal, J., Ibáñez, B., Sanz-Zuasti, J., & Espeso, G. (2012). Changes in Habitat Structure May Explain Decrease in Reintroduced Mohor Gazelle Population in the Guembeul Fauna Reserve, Senegal. *Animals*, *2*(3), Article 3. https://doi.org/10.3390/ani2030347

Naawa, A. M., Folega, F., Kobo-bah, A., Walz, Y., Wala, K., & Amponsah, A. (2025). Spatio-temporal dynamics, drivers of wildfire occurrence and distribution in the northern savannah ecological zone of Ghana. *Scientific African*, *27*, e02580. https://doi.org/10.1016/j.sciaf.2025.e02580

Odadi, W. O., Kimuyu, D. M., Sensenig, R. L., Veblen, K. E., Riginos, C., & Young, T. P. (2017). Fire-induced negative nutritional outcomes for cattle when sharing habitat with native ungulates in an African savanna. *Journal of Applied Ecology*, *54*(3), 935–944. https://doi.org/10.1111/1365-2664.12785

Owen-Smith, N. (Ed.). (2021). How Large Herbivores Transform Savanna Ecosystems. In *Only in Africa: The Ecology of Human Evolution* (pp. 199–219). Cambridge University Press. https://doi.org/10.1017/9781108961646.020

Petretto, M., & Gilbert, T. (2024). *Conservation of Sahelo-Saharan fauna & their arid steppe*

habitats in Tunisia: Report on 2023 conservation action and impact. Marwell Wildlife.

Pohle, J., Signer, J., Eccard, J. A., Dammhahn, M., & Schlägel, U. E. (2024). How to account for behavioral states in step-selection analysis: A model comparison. *PeerJ*, *12*. https://doi.org/10.7717/peerj.16509

Pringle, R. M., Abraham, J. O., Anderson, T. M., Coverdale, T. C., Davies, A. B., Dutton, C. L., Gaylard, A., Goheen, J. R., Holdo, R. M., Hutchinson, M. C., Kimuyu, D. M., Long, R. A., Subalusky, A. L., & Veldhuis, M. P. (2023). Impacts of large herbivores on terrestrial ecosystems. *Current Biology*, *33*(11), R584–R610. https://doi.org/10.1016/j.cub.2023.04.024

Qi, J., Chehbouni, A., Huete, A. R., Kerr, Y. H., & Sorooshian, S. (1994). A modified soil adjusted vegetation index. *Remote Sensing of Environment*, *48*(2), 119–126. https://doi.org/10.1016/0034-4257(94)90134-1

Razali, N. B., Mansor, M. S., Farinordin, F. A., Zaini, M. I.-H. A., Razali, S. H. A., Patah, P. A., Husin, S. M., Hussein, M. S. R., & Nor, S. M. (2025). Mineral supplementation by artificial salt licks is comparatively effective as natural salt licks for Malaysian mammals. *Ecological Processes*, *14*(1), 2. https://doi.org/10.1186/s13717-024-00564-y

Ripple, W. J., Estes, J. A., Beschta, R. L., Wilmers, C. C., Ritchie, E. G., Hebblewhite, M., Berger, J., Elmhagen, B., Letnic, M., Nelson, M. P., Schmitz, O. J., Smith, D. W., Wallach, A. D., & Wirsing, A. J. (2014). Status and Ecological Effects of the World's Largest Carnivores. *Science*. https://doi.org/10.1126/science.1241484

Signer, J., Fieberg, J., & Avgar, T. (2017). Estimating utilization distributions from fitted step-selection functions. *Ecosphere*, *8*(4), e01771. https://doi.org/10.1002/ecs2.1771

Signer, J., Fieberg, J., Reineking, B., Schlägel, U., Smith, B., Balkenhol, N., & Avgar, T. (2024). Simulating animal space use from fitted integrated Step-Selection Functions (iSSF). *Methods in Ecology and Evolution*, *15*(1), 43–50. https://doi.org/10.1111/2041-210X.14263

Signer, J., Smith, B., & Reineking, B. (2024). *Package 'amt'* (p. 0.2.2.0). Cran R. https://CRAN.R-project.org/package=amt

Staver, A. C., Abraham, J. O., Hempson, G. P., Karp, A. T., & Faith, J. T. (2021). The past, present, and future of herbivore impacts on savanna vegetation. *Journal of Ecology*, *109*(8), 2804–2822. https://doi.org/10.1111/1365-2745.13685

Thurfjell, H., Ciuti, S., & Boyce, M. S. (2014). Applications of step-selection functions in ecology and conservation. *Movement Ecology*, *2*(1), 4. https://doi.org/10.1186/2051-3933-2-4

Wacher, T., Amin, R., Newby, J., Hatcha, M. H., Abeye, K., Ali, H., Bourtchiakbé, S. Z., & Banlongar, F. N. (2022). Gazelle–livestock interactions and impact of water resource development

in the Ouadi Rimé–Ouadi Achim Reserve, Chad. *Oryx*, *57*(2), 205–215. https://doi.org/10.1017/S0030605321001629

Wahed, M. A., Hamdan, M. A., Ramadan, S., Gebril, M. F., & Khadrah, A. M. A. (2023). Morphometry, migration rates, and environmental hazards of barchan dunes in the Kharga Depression, Western Desert, Egypt. *Arabian Journal of Geosciences*, *16*(3), 159. https://doi.org/10.1007/s12517-023-11238-y

Wang, L., Zhang, Q., Maxwell, S. E., & Bergeman, C. S. (2019). On standardizing within-person effects: Potential problems of global standardization. *Multivariate Behavioral Research*, *54*(3), 382–403. https://doi.org/10.1080/00273171.2018.1532280

Weiss, A. D. (2001). *Topographic Position and Landforms Analysis*. BibBase. https://bibbase.org/network/publication/weiss-topographicpositionandlandformsanalysis-2001

Whitemore, S., & Gilbert, T. (2024). *International studbook for the scimitar-horned oryx Oryx dammah* (Nineteenth edition). Marwell Wildlife. https://www.marwell.org.uk/wp-content/uploads/2024/05/Marwell-Wildlife-Scimitar-horned-oryx-International-Studbook-2024.pdf

Whyle, K., Mertes, K., Pusey, R., & Al Remeithi, M. (in prep). *What is a season to an oryx? Movement rates identify three seasons for scimitar-horned oryx reintroduced into their native range*.

Woodfine, T., & Gilbert, T. (2016). The Fall and Rise of the Scimitar-Horned Oryx. In *Antelope Conservation* (pp. 280–296). John Wiley & Sons, Ltd. https://doi.org/10.1002/9781118409572.ch14

Worgue, Y. L., & Yamtibaye, T. (2023). *PLAN D'AMENAGEMENT ET DE GESTION DE LA RESERVE DE FAUNE DE OUADI RIME - OUADI ACHIM 2023—2032*. https://saharaconservation.org/wp-content/uploads/2024/08/20230921_pag_rforoa_textepreface_sans_annexes_VF.pdf

A. Appendix

A.1. Extraction of binary layers of wildfires

```
////////////////////////////////////////////////////////////////
/////////// Extraction of yearly binary layer of fires ///////////
////////////////////////////////////////////////////////////////

// 1. We load the Elevation image used to define the border of the ROI
var elevationImage = ee.Image('projects/ee-courtjulien5/assets/Elevation');

// 2. We extract the mask of valid pixels
var mask = elevationImage.mask().toInt();  // convert to integer

// 3. We convert the mask to vectors (polygons)
var elevationContours = mask.reduceToVectors({
  geometry: mask.geometry(),
  scale: 30,
  geometryType: 'polygon',
  eightConnected: false,
  labelProperty: 'mask',
  maxPixels: 1e13
});

// 4. We merge the polygons into a single unified contour that defines the ROI
var unionContour = elevationContours.union().geometry();
var roi = elevationContours.geometry().bounds();

// 5. We define the range of years for analysis
var years = ee.List.sequence(2016, 2023);

// 6. We define a processing function for each year
var yearlyFireImages = years.map(function(year) {
  year = ee.Number(year);
  var startDate = ee.Date.fromYMD(year, 1, 1);
  var endDate   = startDate.advance(1, 'year');

  // We merge the Terra and Aqua MODIS fire products
  var modisFires = ee.ImageCollection('MODIS/061/MOD14A1')
    .merge(ee.ImageCollection('MODIS/061/MYD14A1'))
    .filterDate(startDate, endDate)
    .filterBounds(roi)
    .select('FireMask');

  // We convert the format to binary data and compute the max value per pixel
across the year
  var fireBinary = modisFires.map(function(img) {
    return img.gte(7).selfMask();
  }).max();

  // We add useful metadata for export and identification
```

```
  return fireBinary.set('year', year)
                   .set('system:index', year.format()); // useful for batch
export
});

// 7. We convert the mapped results into an ImageCollection
var fireBinaryCollection = ee.ImageCollection.fromImages(yearlyFireImages);

// 8. We visualize the first year's result for verification
Map.centerObject(roi, 8);
Map.addLayer(fireBinaryCollection.first(), {palette: ['red']}, 'Detected Fires
- 2016');
Map.addLayer(elevationImage, {}, 'Elevation');
Map.addLayer(roi, {color: 'blue'}, 'Elevation Contour');

// 9. We can now export the yearly fire images
years.getInfo().forEach(function(year) {
  var image = fireBinaryCollection.filter(ee.Filter.eq('year', year)).first();
  Export.image.toDrive({
    image: image,
    description: 'Fire_yearly_' + year,
    folder: 'RFOROA_IMAGES',
    scale: 1000,
    region: roi,
    crs: 'EPSG:32634',
    maxPixels: 1e13
  });
});
```

A.2. Creation of the raster: sum of fires 2004 to 2023

A.2.1. Extraction of the yearly sum of wildfires

```
////////////////////////////////////////////////////////////////////////////
// Source Code: last edited by KJM on 12/16/2022
// Aim: We calculate fire frequency across RFOROA (2004–2023, annual batches)
// Adapted by: Caleb Ngaba, 04/23/2025
// Readapted by : Julien Court, 05/15/2025
////////////////////////////////////////////////////////////////////////////

// 1) We load the Elevation image (raster)
var elevationImage = ee.Image('projects/ee-courtjulien5/assets/Elevation');

// 2) We extract the mask of valid pixels and convert it to integer
// because reduceToVectors requires integer input
var mask = elevationImage.mask().toInt();

// 3) We convert the mask to vectors (polygons)
var elevationContours = mask.reduceToVectors({
  geometry: mask.geometry(),
  scale: 30,  // we adjust according to Elevation resolution
```

```
  geometryType: 'polygon',
  eightConnected: false,
  labelProperty: 'mask',
  maxPixels: 1e13
});

// 4) We merge all polygons into a single unified contour
var unionContour = elevationContours.union().geometry({maxError: 1});

// 5) We use this merged contour as our region of interest (ROI)
var rforoaBox = elevationContours.geometry();

// 6) We define global parameters for our analysis
var gDriveFolder = 'RFOROA_IMAGES';
var fireMaskPalette = [
  'C0C0C0','C0C0C0','0000FF','FFFFFF','808000','808080',
  'FFFF00','FF8000','FF0000'
];
var startYear = 2004;
var endYear   = 2023;
var years = ee.List.sequence(startYear, endYear);
var dayMillis = 24 * 60 * 60 * 1000;

// 7) We create a function that builds and sums daily fire composites for one
year
function annualComposite(year){
  year = ee.Number(year);
  var startDate = ee.Date.fromYMD(year,1,1);
  var endDate   = startDate.advance(1, 'year');

  // 7.1) We load active fire collections from Terra and Aqua satellites
  var mod = ee.ImageCollection('MODIS/061/MOD14A1')
              .filterDate(startDate, endDate)
              .filterBounds(rforoaBox)
              .select('FireMask');
  var myd = ee.ImageCollection('MODIS/061/MYD14A1')
              .filterDate(startDate, endDate)
              .filterBounds(rforoaBox)
              .select('FireMask');
  // We merge both collections, cast the FireMask band to float, and sort by
time
  var mcd = mod.merge(myd)
               .cast({bandTypes:{'FireMask':'float'}, bandOrder:['FireMask']})
               .sort('system:time_start');

  // 7.2) We build daily binary fire images by iterating over each day
  var days = ee.List.sequence(startDate.millis(),
endDate.millis().subtract(dayMillis), dayMillis);
  var daily = ee.ImageCollection.fromImages(
    days.map(function(dm){
      var d = ee.Date(dm);
      var imgs = mcd.filterDate(d, d.advance(1,'day'));
      var composite = ee.Algorithms.If(
        imgs.size().gt(0),
        imgs.qualityMosaic('FireMask').clip(rforoaBox),
```

```
        ee.Image(0).rename('FireMask').clip(rforoaBox)
      );
      composite = ee.Image(composite);
      // We convert fire mask to binary: fire detected if FireMask >= 7
      var bin = composite.gte(7).updateMask(composite.gte(7));
      return bin.set('system:time_start', d.millis())
                .set('year', year)
                .set('date', d.format('YYYY-MM-dd'));
    })
  );

  // 7.3) We sum the daily binary images over the entire year
  var sum = daily.reduce(ee.Reducer.sum()).toFloat()
                 .set('year', year)
                 .set('system:time_start', startDate.millis());
  return sum;
}

// 8) We iterate over each year, compute the annual composite, display it, and
export it
years.getInfo().forEach(function(y){
  var comp = annualComposite(y);
  Map.addLayer(comp, {min:0, max:365, palette:['white','yellow','red']},
'FireSum_'+y);
  Export.image.toDrive({
    image: comp,
    description: 'FireFreq_Sum_'+ y,
    folder: gDriveFolder,
    fileNamePrefix: 'FireFreq_'+ y,
    region: rforoaBox,
    scale: 1000,
    crs: 'EPSG:32634',
    maxPixels: 1e13
  });
});

// 9) We perform a visual check of the first year's composite
var sample = annualComposite(startYear);
Map.centerObject(rforoaBox, 6);
Map.addLayer(sample, {min:0, max:120, palette:['white','yellow','red']},
'Sample '+ startYear);
print('Annual composite for ' + startYear, sample);
```

A.2.2. Sum of the yearly layers of wildfires

```
#########################################################
################ BURNED AREAS SUM ###################
#########################################################
#install.packages("terra")
library(terra)

# We load the fire frequency rasters for each year from 2004 to 2023. Adapt
your paths to where your data are stored
```

```
FireFreq_2004 <- rast("C:/Users/court/Documents/FireFreq_2004_june.tif")
FireFreq_2005 <- rast("C:/Users/court/Documents/FireFreq_2005_june.tif")
FireFreq_2006 <- rast("C:/Users/court/Documents/FireFreq_2006_june.tif")
FireFreq_2007 <- rast("C:/Users/court/Documents/FireFreq_2007_june.tif")
FireFreq_2008 <- rast("C:/Users/court/Documents/FireFreq_2008_june.tif")
FireFreq_2009 <- rast("C:/Users/court/Documents/FireFreq_2009_june.tif")
FireFreq_2010 <- rast("C:/Users/court/Documents/FireFreq_2010_june.tif")
FireFreq_2011 <- rast("C:/Users/court/Documents/FireFreq_2011_june.tif")
FireFreq_2012 <- rast("C:/Users/court/Documents/FireFreq_2012_june.tif")
FireFreq_2013 <- rast("C:/Users/court/Documents/FireFreq_2013_june.tif")
FireFreq_2014 <- rast("C:/Users/court/Documents/FireFreq_2014_june.tif")
FireFreq_2015 <- rast("C:/Users/court/Documents/FireFreq_2015_june.tif")
FireFreq_2016 <- rast("C:/Users/court/Documents/FireFreq_2016_june.tif")
FireFreq_2017 <- rast("C:/Users/court/Documents/FireFreq_2017_june.tif")
FireFreq_2018 <- rast("C:/Users/court/Documents/FireFreq_2018_june.tif")
FireFreq_2019 <- rast("C:/Users/court/Documents/FireFreq_2019_june.tif")
FireFreq_2020 <- rast("C:/Users/court/Documents/FireFreq_2020_june.tif")
FireFreq_2021 <- rast("C:/Users/court/Documents/FireFreq_2021_june.tif")
FireFreq_2022 <- rast("C:/Users/court/Documents/FireFreq_2022_june.tif")
FireFreq_2023 <- rast("C:/Users/court/Documents/FireFreq_2023_june.tif")

# We combine all the yearly rasters into a list.
fire_rasters <- list(
  FireFreq_2004, FireFreq_2005, FireFreq_2006, FireFreq_2007, FireFreq_2008,
  FireFreq_2009, FireFreq_2010, FireFreq_2011, FireFreq_2012, FireFreq_2013,
  FireFreq_2014, FireFreq_2015, FireFreq_2016, FireFreq_2017, FireFreq_2018,
  FireFreq_2019, FireFreq_2020, FireFreq_2021, FireFreq_2022, FireFreq_2023
)

# We create a raster stack from the list of yearly rasters.
stacked_rasters <- rast(fire_rasters)

# We replace NA values in the stack with zeros to avoid issues during
summation.
stacked_rasters[is.na(stacked_rasters)] <- 0

# We sum all the yearly rasters to get the total fire frequency over the entire
period.
sum_raster <- sum(stacked_rasters)

# We save the result (adapt your path)
# writeRaster(sum_raster,
"C:/Users/court/Documents/FireFreq_total_2004_2023.tif", overwrite = TRUE)
```

A.3. Topographic Position Index calculation code

```
################################################
########## Creation of the TPI layers ##########
################################################

# If necessary, download the package
```

```
# install.packages("whitebox")

# Load required libraries
library(whitebox)

############ TPI at fine scale (3×3 window)
# We calculate the Topographic Position Index (TPI) as the difference
# between each cell's elevation and the mean elevation of its 3×3 neighborhood.
wbt_diff_from_mean_elev(
  dem    = "C:/Users/court/Documents/Elevation.tif",
  output = "C:/Users/court/Documents/tpi_3x3.tif",
  filterx = 3,    # window width in pixels
  filtery = 3     # window height in pixels
)

############ TPI at medium scale (7×7 window)
# Same calculation, but using a larger 7×7 pixels window
# to capture broader-scale topographic variation.
wbt_diff_from_mean_elev(
  dem    = "C:/Users/court/Documents/Elevation.tif",
  output = "C:/Users/court/Documents/tpi_7x7.tif",
  filterx = 7,
  filtery = 7
)
```

A.4. Topographic Convergence Index calculation code

```
#######################################################
############ Creation of the TCI layers ############
#######################################################

#install.packages("stars", dependencies = TRUE)
#install.packages("starsExtra")
#install.packages("terra")

library(stars)
library(terra)
library(starsExtra)

############ Convergence Index

elevation <- rast("C:/Users/court/Documents/Elevation.tif")

asp <- terrain(elevation, v = "aspect", unit = "degrees")

asp_stars <- st_as_stars(asp, proxy = FALSE)

#saveRDS(asp_stars, " C:/Users/court/Documents/stars.rds")

ci <- CI(asp_stars, k = 3, na.rm = TRUE)
```

```
asp_stars <- readRDS("C:/Users/court/Documents/stars.rds")

ci2 <- CI(asp_stars, k = 7, na.rm = TRUE)

ci_terra <- rast(ci)
ci2_terra <- rast(ci2)

writeRaster(ci_terra, "C:/Users/court/Documents/TCI_7x7.tif", overwrite = TRUE)

writeRaster(ci2_terra, "C:/Users/court/Documents/TCI_7x7.tif", overwrite =
TRUE)

############### TCI 7x7 neighborhood

# We calculate de Topographic Convergence Index, with k = 3 => 3 neighbours
ci <- CI(asp_stars, k = 3, na.rm = TRUE)

# We retransform the result intro a terra format
ci_terra <- rast(ci)

# We save the result
# Adapt the path to your device
writeRaster(ci_terra, "C:/Users/court/Documents/TCI_3x3.tif", overwrite = TRUE)
```

A.5. Creation of the layer of human activity

A.5.1. R code used for the creation of the human activity layer

```
############################################################
########## Creation of the human activity raster ##########
############################################################

#Install the package if it’s the first time you use this
#library on your device
#install.packages("sf")
#install.packages("sp")
#install.packages("raster")
#install.packages("terra")

# Load the required libraries
library(sf)
library(sp)
library(raster)
library(terra)

# We load the different raster layers we will use
# Adapt each path to your computer
wacher_2017 <-
st_read("C:/Users/court/Documents/wacher_named_settlements_2017.shp")
```

```
HC_KM <- st_read("C:/Users/court/Documents/HCSettlements_KM.shp")
wacher_2020 <-
st_read("C:/Users/court/Documents/Wacher_hafiris_named_Mar2020.shp")
# The water layer contains the wells, wadis wells and hafiris layers already
merged
water <-
st_read("C:/Users/court/Documents/waters_with_transhumants_merged.shp")
transhumant_paths <-
st_read("C:/Users/court/Documents/Itineraire_de_transhumance.shp")
transhumant <- st_read("C:/Users/court/Documents/Zone de concentration des
grands transhumants.shp")


# We reproject the layers to the CRS 32634 for those that naturally have
another one
wacher_2017 <- st_transform(wacher_2017, crs = 32634)
HC_KM <- st_transform(HC_KM, crs = 32634)
wacher_2020 <- st_transform(wacher_2020, crs = 32634)
transhumant_paths <- st_transform(transhumant_paths, crs = 32634)
transhumant <- st_transform(transhumant, crs = 32634)

# To allow to a correct fusion of all layers,
# we remove all attributes to keep only the geometries of layers that contain
them
geoms_lines <- st_geometry(transhumant_paths)
geoms_water <- st_geometry(water)
geoms_transhumant <- st_geometry(transhumant)

# We create a polygon with an extent that we will use
# to harmonize the extents of each layer
# We exclude HC_KM of that because it has an extent problem
#and observations showed this layer is not determinant for the extent
bbox_all <- st_as_sfc(st_bbox(st_union(st_geometry(wacher_2017),
                                       st_geometry(wacher_2020),
                                       st_geometry(transhumant_paths),
                                       st_geometry(water))))


# We change the extent of each layer to make it fit the polygon bbox_all
HC_KM_clipped        <- st_intersection(HC_KM, bbox_all)
wacher_2020_clipped <- st_intersection(wacher_2020, bbox_all)
geoms_water_clipped <- st_intersection(geoms_water, bbox_all)
geoms_lines_clipped <- st_intersection(geoms_lines, bbox_all)
transhumant_clipped <- st_intersection(geoms_transhumant, bbox_all)

# We extract the extent of the reference polygon for the extent
ext_bbox <- st_bbox(bbox_all)

# We create a raster of resolution 30 m and with the harmonized extent
# We add 6km outside of the extent because some features can be next to the
border
# of the extent and we are interested to have 5 km around each feature
# This allows to include all useful information in the raster
offset <- 6000

base_raster_common_expanded <- raster(
```

```
  xmn = ext_bbox["xmin"] - 6000,  # left - 6 km
  xmx = ext_bbox["xmax"] + 6000,  # right + 6 km
  ymn = ext_bbox["ymin"] - 6000,  # down - 6 km
  ymx = ext_bbox["ymax"] + 6000,  # up + 6 km
  res = 30,
  crs = st_crs(wacher_2017)$proj4string
)

# We create a function to convert a vectorial layer with feature into a raster
# layer with a buffer zone of 5 km around the position of the vectorial
features

# The 1st argument is the vector layer, the 2nd is the model of raster layer
and
# the 3rd layer is the size of the buffer
create_distance_raster <- function(sf_object, base_raster, buffer_distance =
5000) {
  # We create the buffers
  buffer <- st_buffer(sf_object, dist = buffer_distance)

  # We convert the objects into Spatial format (necessary to rasterize)
  sf_sp <- as(sf_object, "Spatial")
  buffer_sp <- as(buffer, "Spatial")

  # We rasterize the vector layer using the raster layer model
  # Where a feature is present, we have a 1, where it's not, 0
  ref_raster <- rasterize(sf_sp, base_raster, field = 1)

  # We create a distance raster indicating for each pixel
  # the distance with the nearest 1 pixel
  dist_raster <- distance(ref_raster)

  # We remove data outside of the buffers
  buffer_mask <- rasterize(buffer_sp, base_raster, field = 1)
  dist_raster_masked <- mask(dist_raster, buffer_mask)

  return(dist_raster_masked)
}

# We apply the function on each vectorial layer to transform them into buffer
rasters
raster_water <- create_distance_raster(geoms_water_clipped,
base_raster_common_expanded)
raster_2020 <- create_distance_raster(wacher_2020_clipped,
base_raster_common_expanded)
raster_2017 <- create_distance_raster(wacher_2017, base_raster_common_expanded)
raster_HC_KM <- create_distance_raster(HC_KM_clipped,
base_raster_common_expanded)
raster_transhumant <- create_distance_raster(transhumant_clipped,
base_raster_common_expanded)

#raster_lines <- create_distance_raster(geoms_lines_clipped,
base_raster_common_expanded) # does not work !

# We can plot and check the characteristics of the resulting layers
plot(raster_2017)
```

```
summary(raster_2017)

# We save the intermediary raster
# Adapt the paths to your device
writeRaster(raster_water, filename =
"C:/Users/court/Documents/Rasterwater.tif", overwrite = TRUE)
writeRaster(raster_2020, filename = "C:/Users/court/Documents/raster_2020.tif",
overwrite = TRUE)
writeRaster(raster_2017, filename = "C:/Users/court/Documents/raster_2017.tif",
overwrite = TRUE)
writeRaster(raster_HC_KM, filename =
"C:/Users/court/Documents/raster_HC_KM.tif", overwrite = TRUE)
writeRaster(raster_transhumant, filename =
"C:/Users/court/Documents/raster_transhumant.tif", overwrite = TRUE)

# If needed, the layers can be uploaded after saved by this code
# adapt the paths to your device
#raster_water <- rast("C:/Users/court/Documents/Rasterwater.tif")
#raster_2020 <- rast("C:/Users/court/Documents/raster_2020.tif")
#raster_2017 <- rast("C:/Users/court/Documents/raster_2017.tif")
#raster_HC_KM <- rast("C:/Users/court/Documents/raster_HC_KM.tif")
#raster_transhumant <- rast("C:/Users/court/Documents/raster_transhumant.tif")

# We give them smaller names
s1 <- as(raster_HC_KM, "SpatRaster")
s2 <- as(raster_2017, "SpatRaster")
s3 <- as(raster_2020, "SpatRaster")
s4 <- as(raster_water, "SpatRaster")
s5 <- as(raster_transhumant, "SpatRaster")

# We upload the result of the problematic layer created in QGIS
raster_lines <-
rast("C:/Users/court/Documents/test_distance_buffer_transhumance.tif")

# We resample the problematic layer to make sure it's compatible with the
others
raster_lines_resampled <- resample(raster_lines, s3, method = "bilinear")

# We save the problematic layer resampled
# Adapt the path to your device
writeRaster(raster_lines_resampled, filename =
"C:/Users/court/Documents/raster_lines_resampled2.tif", overwrite = TRUE)

# If needed, it can be loaded with this code
#raster_lines_resampled <-
rast("C:/Users/court/Documents/raster_lines_resampled2.tif")

# We add it to the group of rasters
s6 <- as(raster_lines_resampled, "SpatRaster")

# We combine all the raster together in a vector
s <- c(s1, s2, s3, s4, s5, s6)

# We create a single layer "merged" that will always keep the smallest distance
#between those available int the raster for each pixel
merged <- app(s, fun = function(x) {
```

```
  if (all(is.na(x))) {
    NA
  } else {
    min(x, na.rm = TRUE)
  }
},
filename = "C:/Users/court/Documents/merged.tif",
overwrite = TRUE)

plot(merged)
summary(merged)

# We can save the merged layer with this code
# Adapt the path to your device
writeRaster(merged, filename =
"C:/Users/court/Documents/distance_to_settlements_5km_2.tif", overwrite = TRUE)

# We can load the result with this code if needed
# Adapt the path to your device
#merged <- rast("C:/Users/court/Documents/distance_to_settlements_5km_2.tif")

# Now we have a unique raster indicating the distance of each pixel to the
nearest point of human activity
# We can introduce the probability values
# (1 - x / 5000)^2 is the function that allows us to convert the distance into
probabilities varying between 0 (for the limit of the buffer) and
# 1 (for the center of the buffer) with a quadratic relationship
exp <- app(merged, fun = function(x) {
  out <- (1 - x / 5000)^2
  out[is.na(x)] <- NA
  return(out)
})

# We just have to save this raster and this section is finished
writeRaster(exp, filename =
"C:/Users/court/Documents/probability_settlements.tif", overwrite = TRUE)
```

A.5.2. Processing workflow for the transhumant paths layer

To obtain the buffer zone alternatively for this layer, we upload it in QGIS.

We use the tool vector > geoprocessing tools > buffer, and then convert the result to raster using the tool GDAL > Vector conversion > Rasterize (vector to raster)

The options used to rasterize are :

- fixed value to burn : 1,
- output raster size unit : georeferenced units,
- Width/horizontal resolution : 0.003,
- Height/vertical resolution : 0.003,

- Assign a specific nodata value to output bands [optional] : -99999

Then, we take again the original vector layer and convert it into a raster using the exact same method than for the buffer layer. We transform the rasterized lines layer into a distance raster using the QGIS tool GDAL > raster analysis > proximity (raster distance).

The options we use are the same than to rasterize :

- Fixed value to burn : 1,
- Output raster size unit : georeferenced units,
- Width/horizontal resolution : 0.003,
- Height/vertical resolution : 0.003,
- Assign a specific nodata value to output bands [optional] : -99999

Finally, we can use the raster calculator and use the following expression to keep the distance values only inside the buffers :

```
"IF("zones_tampon@1" != -99999, "proximity_map@1", -99999)"
```

Once it's done, we have a raster similar to what does the R function for the other layers.

A.6. First operations on the dataset

```
# If not already done, download the packages
# install.packages("lubridate")
# install.packages("dplyr")

# We load the useful libraries
library(lubridate)
library(dplyr)

# We import the CSV file into a dataframe (adapt the path)
data <- read.csv2("C:/Users/court/Documents/oryx_locs_07112024.csv", sep=",",
header = TRUE) # put the path of the dataset

# We check the structure of the imported dataframe
str(data)

# We convert the 'WAT' column into date-time objects with the appropriate
timezone
data$WAT <- as.POSIXct(data$WAT, format="%Y-%m-%d %H:%M:%S", tz="Africa/Lagos")

# We create new columns extracting only the date and only the time from 'WAT'
data$date <- as.Date(data$WAT)
data$time <- format(data$WAT, format="%H:%M:%S")
data$year <- year(data$WAT)
```

```
# We order the dataframe by 'burst_ID' and 'WAT' (time)
data <- data %>%
  arrange(burst_ID, WAT)

# We add a unique ID to each row to keep the order
data <- data %>%
  mutate(id = row_number())  # Create a unique ID for each row

# We remove duplicated records based on 'burst_ID' and 'WAT'
data <- data[!duplicated(data[, c("burst_ID", "WAT")]), ]

# We examine the structure of the cleaned dataframe
str(data)

# We write the cleaned dataframe back to CSV (adapt the path)
# write.csv(data, "C:/Users/court/Documents/data.csv", row.names = FALSE)
```

A.7. Creation of the summary table

```
################################################################
#################### SUMMARY TABLE #############################
################################################################

# If not already done, we install the required packages
#install.packages("dplyr")
#install.packages("DT")
#install.packages("lubridate")


# We load the useful libraries
library(dplyr)
library(DT)
library(lubridate)

# In this section, we create a summary table to get a better overview of our
data

# We import the CSV containing information about data gaps
sho_aug <- read.csv2("C:/Users/court/Documents/sho_master_info_Aug2024.csv",
sep=",", header = TRUE) # replace by the correct path to the file

# We convert the date columns to proper date format in R
sho_aug <- sho_aug %>%
  mutate(
    gap_begin = as.POSIXct(gap_begin, format = "%m/%d/%Y"),
    gap_end = as.POSIXct(gap_end, format = "%m/%d/%Y")
  )

# We create a table containing the earliest date for every burst_ID
min_dates <- data %>%
```

```
  group_by(burst_ID) %>%
  summarise(min_date_burst = min(date))

# We create the summary table grouping by burst_ID
new_table <- data %>%
  group_by(burst_ID) %>%
  summarise(
    collar_ID = unique(collar_ID),  # We return the unique collar_ID value for
each burst_ID
    Date_Start = min(date),          # We return the earliest date for each
burst_ID
    # If the earliest date for this burst_ID is also the earliest for the whole
collar_ID,
    # then we remove the acclimatisation period of the first 70 days by
shifting Date_Start_valid
    Date_Start_valid = if_else(
      Date_Start == min_dates$min_date_burst[min_dates$burst_ID ==
unique(burst_ID)],
      Date_Start + 70,
      Date_Start
    ),
    Date_End = max(date)             # We return the latest date for each
burst_ID
  )

# We check the structure of the resulting summary table
str(new_table)

# We convert the date columns to POSIXct format for consistency
new_table <- new_table %>%
  mutate(
    Date_Start = as.POSIXct(Date_Start),
    Date_Start_valid = as.POSIXct(Date_Start_valid),
    Date_End = as.POSIXct(Date_End)
  )
```

A.7.1. Addition of the date gaps

```
########################################################
############# Addition of the gap columns #############
########################################################

# If not done already, download the required packages
#install.packages("dplyr")
#install.packages("DT")


# Load the libraries
library(dplyr)
library(DT)


# We join the summary table with the data gaps table by 'collar_ID', selecting
only relevant columns
```

```
table <- new_table %>%
  left_join(
    sho_aug %>%
      dplyr::select(collar_ID, gap_begin, gap_end),  # We select only the
necessary columns
    by = "collar_ID"
  ) %>%
  # We keep gap dates only if they fall within the tracking period defined by
Date_Start and Date_End
  mutate(
    gap_begin = if_else(
      (gap_begin >= Date_Start & gap_begin <= Date_End) |
      (gap_end >= Date_Start & gap_end <= Date_End),
      gap_begin,
      as.POSIXct(NA)
    ),
    gap_end = if_else(
      (gap_begin >= Date_Start & gap_begin <= Date_End) |
      (gap_end >= Date_Start & gap_end <= Date_End),
      gap_end,
      as.POSIXct(NA)
    )
  )

# We examine the structure of the resulting table with added gap information
str(table)

##### Division of bursts based on data gaps and filter #####

# When gap columns contain non-NA values, we split those rows into two:
# The first part ends at the beginning of the gap
original_rows <- table %>%
  mutate(
    Date_End = if_else(!is.na(gap_begin), gap_begin, Date_End)  # We replace
Date_End by gap_begin if gap_begin exists
  )

# The second part starts at the end of the gap and continues until the original
end
duplicated_rows <- table %>%
  filter(!is.na(gap_begin) & !is.na(gap_end)) %>%  # We select rows with non-NA
gaps
  mutate(
    Date_Start = gap_end                                # We replace Date_Start
by gap_end
  )

# We combine the original and new rows to create the updated summary table
final_table <- bind_rows(original_rows, duplicated_rows)

# We check the structure of the resulting dataframe
str(final_table)

# We convert date columns to POSIXct format with timezone for consistency
final_table <- final_table %>%
```

```
  mutate(
    Date_Start = as.POSIXct(Date_Start, format = "%Y-%m-%d",
tz="Africa/Lagos"),
    Date_End = as.POSIXct(Date_End, format = "%Y-%m-%d", tz="Africa/Lagos"),
    Date_Start_valid = as.POSIXct(Date_Start_valid, format = "%Y-%m-%d",
tz="Africa/Lagos")
  )

# We update Date_Start_valid for the new rows that have shifted their
Date_Start
final_table <- final_table %>%
  mutate(
    Date_Start_valid = if_else(Date_Start > Date_Start_valid, Date_Start,
Date_Start_valid)
  )

# We create a new column 'points' counting the number of data records for each
burst_ID
# between Date_Start_valid and Date_End
final_table <- final_table %>%
  rowwise() %>%
  mutate(
    points = sum(
      data$burst_ID == burst_ID &
      data$WAT >= Date_Start_valid &
      data$WAT <= Date_End
    )
  ) %>%
  ungroup()

# We create a new column 'duration' giving the number of days between
Date_Start_valid and Date_End
final_table <- final_table %>%
  mutate(
    duration = as.numeric(difftime(Date_End, Date_Start_valid, units = "days"))
  )

# We create a new column 'interval_h' calculating the average interval in hours
between consecutive data points
final_table <- final_table %>%
  mutate(
    interval_h = if_else(points > 0, 24 * duration / points, NA_real_)
  )

# We create a new column 'interval_category' by rounding 'interval_h' to
facilitate classification
final_table$interval_category <- round(final_table$interval_h)

# We check the final result interactively
datatable(final_table)

# We save the summary table as a CSV file (adjust path as needed)
# write.csv(final_table, "C:/Users/court/Documents/summary_table_base.csv",
row.names = FALSE)
```

A.7.2. Filtering of the incompatible time intervals

```
##############################################################################
################ Filter of invalid time intervals ########################
##############################################################################

# If not done already, download the required packages
#install.packages("dplyr")
#install.packages("tidyverse")

library(dplyr)
library(ggplot2)

# We load the summary table if needed (currently commented out)
# final_table <- read.csv2("C:/Users/court/Documents/summary_table_base.csv",
sep=",", header = TRUE)

# We check the structure of the summary table
str(final_table)

# We filter the data to keep only bursts with interval categories of 1, 2, or 4
hours
table <- final_table %>%
  filter(interval_category %in% c(1, 2, 4))

# We create a bar plot showing the distribution of bursts by rounded average
time interval
p <- ggplot(table, aes(x = interval_category)) +
  geom_bar(fill = "steelblue") +
  labs(x = "Rounded average time interval between data points (hours)",
       y = "Number of bursts") +
  theme_minimal()

# We save the plot as a PNG file with specified size and resolution
ggsave(
  filename = "C:/Users/court/Documents/burst_distribution_may.png",  # output
filename
  plot     = p,                                  # ggplot object to save
  width    = 8,    # width in inches
  height   = 5,    # height in inches
  dpi      = 300   # resolution in dots per inch
)

# We check the structure of the filtered data
str(table)

# We write the filtered table to CSV (adapt the path)
# write.csv(table, "C:/Users/court/Documents/summary_table_interval.csv",
row.names = FALSE)
```

A.7.3. Addition of seasons

```
##############################################
################ Add seasons ################
##############################################


# If not done already, download the required packages
#install.packages("dplyr")
#install.packages("lubridate")


# Load the library
library(dplyr)
library(lubridate)

# We load the summary table if necessary (adapt the path)
# table <-
read.csv2("C:/Users/court/Documents/summary_table_interval_filtered.csv",
sep=",", header = TRUE)

# We check the structure of the summary table
str(table)

# We define a function to check if a season is fully included within the date
interval of a burst
is_fully_in <- function(start_date, end_date, season_beginning, season_end) {
  return(start_date <= season_beginning & end_date >= season_end)
}

# We define the years of interest
years <- seq(2016, 2024)

# We create an empty dataframe to store seasons and their date ranges
table_seasons <- data.frame()

# We define seasons for each year and add them to the table_seasons dataframe
for (year in years) {
  table_seasons <- rbind(table_seasons,
    data.frame(
      season = c("hot_dry", "rainy"),
      date_beginning = as.POSIXct(c(paste(year, "-03-13", sep=""),
                                    paste(year, "-07-11", sep="")), tz =
"Africa/Lagos"),
      date_end = as.POSIXct(c(paste(year, "-07-10", sep=""),
                              paste(year, "-10-01", sep="")), tz =
"Africa/Lagos")
    )
  )

  # We add the "cool_dry" season that spans from October 2nd to March 12th of
the following year
  table_seasons <- rbind(table_seasons,
    data.frame(
      season = "cool_dry",
      date_beginning = as.POSIXct(paste(year, "-10-02", sep=""), tz =
"Africa/Lagos"),
      date_end = as.POSIXct(paste(year + 1, "-03-12", sep=""), tz =
```

```
"Africa/Lagos")
    )
  )
}

# We create a new column combining the season name and the year for easier
reference
table_seasons <- table_seasons %>%
  mutate(column_name = paste(season, format(date_beginning, "%Y"), sep = "_"))

# For each season, we create a new column in the summary table:
# It contains 1 if the season is fully included within the burst interval,
otherwise 0
for (col in table_seasons$column_name) {
  season_beginning <- table_seasons$date_beginning[table_seasons$column_name ==
col]
  season_end <- table_seasons$date_end[table_seasons$column_name == col]
  Date_Start_valid <- as.POSIXct(table$Date_Start_valid, tz = "Africa/Lagos")
  Date_End <- as.POSIXct(table$Date_End, tz = "Africa/Lagos")

  table[[col]] <- ifelse(is_fully_in(Date_Start_valid, Date_End,
season_beginning, season_end), 1, 0)
}

# We check the structure of the seasons table
str(table_seasons)

# We save the seasons dataframe if needed (adapt the path)
# write.csv(table_seasons,
"C:/Users/court/Documents/table_seasons_intermediary.csv", row.names = FALSE)

# We prepare to organize season columns in the right order
organized_columns <- colnames(table)
organized_new_columns <- c()

# We build ordered column names for each season and year (adjust variable name
if needed)
for (year in years) {
  cool_dry_1_col <- paste0("cool_dry_1_", year)
  hot_dry_col <- paste0("hot_dry_", year)
  rainy_col <- paste0("rainy_", year)
  cool_dry_2_col <- paste0("cool_dry_2_", year)
  if (all(c(cool_dry_1_col, hot_dry_col, rainy_col, cool_dry_2_col) %in%
organized_columns)) {
    organized_new_columns <- c(organized_new_columns, cool_dry_1_col,
hot_dry_col, rainy_col, cool_dry_2_col)
  }
}

# We add original columns and reorder them accordingly
organized_columns_left <- setdiff(colnames(table), organized_new_columns)
organized_columns_final <- c(organized_columns_left, organized_new_columns)
table <- table %>% dplyr::select(all_of(organized_columns_final))

# We get the list of all years in the dataset and sort them ascending
years <- unique(gsub(".*_(\\d{4})", "\\1", colnames(table)))
```

```
years <- sort(as.numeric(years))

# We merge "cool_dry_2" season of one year with "cool_dry_1" season of the next
year
for (i in 1:(length(years) - 1)) {
  year_current <- years[i]
  year_next <- years[i + 1]

  cool_dry_2_col <- paste0("cool_dry_2_", year_current)
  cool_dry_1_col_next <- paste0("cool_dry_1_", year_next)

  if (cool_dry_2_col %in% colnames(table) && cool_dry_1_col_next %in%
colnames(table)) {
    merged_col <- paste0("cool_dry_", year_current, "_", year_next)
    # We set merged column to 1 only if both original columns are 1; otherwise
0
    table[[merged_col]] <- ifelse(table[[cool_dry_2_col]] == 1 &
table[[cool_dry_1_col_next]] == 1, 1, 0)
    # We remove the original split columns after merging
    table[[cool_dry_2_col]] <- NULL
    table[[cool_dry_1_col_next]] <- NULL
  }
}

# We reorder columns so seasons appear in the order: hot_dry, rainy, cool_dry
for each year
season_order <- c("hot_dry_", "rainy_", "cool_dry_")
for (year in years) {
  ordered_columns <- paste0(season_order, year)
  existing_columns <- intersect(ordered_columns, colnames(table))
  other_columns <- setdiff(colnames(table), existing_columns)
  table <- table[, c(other_columns, existing_columns)]
}

# We create a new column "no_season" indicating how many seasons each burst
covers
no_season <- rowSums(table[, 13:ncol(table)], na.rm = TRUE)
# We insert "no_season" column between the 11th and 12th columns
table <- cbind(table[, 1:12], no_season = no_season, table[, 13:ncol(table)])

# We remove irrelevant columns and rows: columns with only zeros/NA and rows
with zero seasons
table <- table %>%
  dplyr::select(where(~ any(!is.na(.) & . != 0))) %>%
  filter(no_season > 0)

# We check the final structure of the table
str(table)

# We save the final table with seasons if needed (adapt the path)
# write.csv(table, "C:/Users/court/Documents/summary_table_season.csv",
row.names = FALSE)
```

A.7.4. Seasonal table creation

```
################################################################
################## STATISTICS BY SEASON ########################
################################################################

#If not done already, download the required package
#install.packages("DT")


# Load the library
library(DT)

# Here, we create a seasonal summary table indicating the number of valid
bursts for each season to ensure that each season has a sufficient sample size

# We load the data if necessary (adapt the path)
# table <- read.csv2("C:/Users/court/Documents/summary_table_season.csv",
sep=",", header = TRUE)

# We display the data interactively for inspection
datatable(table)

# We select the columns corresponding to seasons
subset_table <- table[, 13:33]

# We convert all selected columns to numeric type
subset_table[] <- lapply(subset_table, as.numeric)

# We calculate the sum of valid bursts for each season (i.e., number of 1s in
each column)
sum_season <- colSums(subset_table, na.rm = TRUE)

# We create a new table summarizing the counts per season
seasonal_table <- data.frame(
  Name = colnames(subset_table),
  Sum = sum_season
)

# We display the seasonal summary interactively
datatable(seasonal_table)

# We save the seasonal summary as a CSV file if needed (adapt the path)
# write.csv(seasonal_table, "C:/Users/court/Documents/seasonal.csv", row.names
= FALSE)
```

A.8. Filtering of the original dataset

A.8.1. Filtering of irrelevant bursts

```
####################################################
```

```
############    FILTER ORIGINAL DATASET   ############
#####################################################

# If not done already, download the required package
#install.packages("dplyr")


# Load the library
library(dplyr)


#We load the summary table if necessary
#table <- read.csv2("C:/Users/court/Documents/summary_table_season.csv",
sep=",", header = TRUE)

# We load the "table_season" dataframe
#table_seasons <- read.csv2("C:/Users/court/Documents/seasonal_table.csv",
sep=",", header = TRUE)

str(table)

# We load the dataset (adapt the path)
data2 <- read.csv2("C:/Users/court/Documents/data.csv", sep=",", header = TRUE)
str(data2)

############### FILTER BURSTS
# We create a new dataframe containing only rows with burst_IDs present in the
summary table
filtered_data <- data %>%
  filter(burst_ID %in% table$burst_ID)

# We check the structure of the filtered data
str(filtered_data)

# We convert 'utm_e' and 'utm_n' columns to numeric type
filtered_data$utm_e <- as.numeric(filtered_data$utm_e)
filtered_data$utm_n <- as.numeric(filtered_data$utm_n)

# We ensure 'WAT' column is in POSIXct datetime format with the correct
timezone
filtered_data$WAT <- as.POSIXct(filtered_data$WAT, format = "%Y-%m-%d
%H:%M:%S", tz="Africa/Lagos")

# We count the number of missing values in 'WAT'
sum(is.na(filtered_data$WAT))

# We remove rows where 'WAT' is missing (NA)
filtered_data <- filtered_data %>%
  filter(!is.na(WAT))
```

A.8.2. Filtering of incomplete seasons

```
#########################################################################
###################### FILTER INCOMPLETE SEASONS  ######################
```

```
##############################################################################

# If not done already, download the required packages
#install.packages("dplyr")
#install.packages("lubridate")
#install.packages("stringr")
#install.packages("tidyr")
#install.packages("DT")


# Load the libraries
library(dplyr)
library(lubridate)
library(stringr)
library(tidyr)
library(DT)


# We assign a season label to each record based on the month and day
filtered_data_season <- filtered_data %>%
  mutate(
    m = month(date),
    d = day(date),
    season = case_when(
      # From March 13 (03-13) to July 10 (07-10)
      (m == 3  & d >= 13) |
      (m %in% 4:6) |
      (m == 7  & d <= 10)       ~ "hot_dry",

      # From July 11 (07-11) to October 1 (10-01)
      (m == 7  & d >= 11) |
      (m %in% 8:9) |
      (m == 10 & d <= 1)        ~ "rainy",

      # From October 2 (10-02) to December 31 (12-31)
      (m == 10 & d >= 2) |
      (m %in% 11:12)            ~ "cool_dry1",

      # The rest of the year (January to March 12)
      TRUE                      ~ "cool_dry2",
    )
  ) %>%
  select(-m, -d)

# We add the year column extracted from the datetime
filtered_data_season <- filtered_data_season %>%
  mutate(year = year(WAT))

# We check the structure of the data
str(filtered_data_season)

# We create a combined season-year label,
# adjusting the year for "cool_dry2" season to the previous calendar year
filtered_data_season_y <- filtered_data_season %>%
  mutate(
```

```
    season_year = if_else(
      season == "cool_dry2",
      paste0(season, "_", year - 1),
      paste0(season, "_", year)
    )
  )

# We verify unique season-year combinations
unique(filtered_data_season_y$season_year)

# We unify "cool_dry1" and "cool_dry2" labels into a single "cool_dry"
filtered_data_season_y_cool <- filtered_data_season_y %>%
  mutate(
    season_year = str_replace(season_year, "cool_dry[12]", "cool_dry"),
    season = str_replace(season, "cool_dry[12]", "cool_dry")
  )

# We check the structure and unique seasons again
str(filtered_data_season_y_cool)
unique(filtered_data_season_y_cool$season)

# We save the cleaned seasonal data if needed (commented out)
# write.csv(filtered_data_season_y_cool,
"C:/Users/court/Documents/filtered_data_season_y_cool_june.csv", row.names =
FALSE)

# 1) We identify the season_year columns in the summary table
season_cols <- grep("_(\\d{4})$", names(table), value = TRUE)

# 2) We reshape 'table' to long format with columns (burst_ID, season_year,
flag)
table_long <- table %>%
  select(burst_ID, all_of(season_cols)) %>%
  pivot_longer(
    cols      = all_of(season_cols),
    names_to  = "season_year",
    values_to = "flag"
  )

# We display the reshaped table interactively
datatable(table_long)

# 3) We join the seasonal filtered data with the summary table flags,
# filter to keep only rows with flag == 1, then remove the flag column
filtered_data_total <- filtered_data_season_y_cool %>%
  left_join(
    table_long,
    by = c("burst_ID", "season_year"),
    relationship = "many-to-many"    # Explicitly specify expected join type
  ) %>%
  filter(flag == 1) %>%
  select(-flag)

# We check the structure and unique seasons in the joined dataset
str(filtered_data_total)
```

```
unique(filtered_data_total$season)

# We check the date range of the filtered dataset
max(filtered_data_total$WAT)
min(filtered_data_total$WAT)

# We save the fully filtered dataset if needed (adapt the path)
# write.csv(filtered_data_total,
"C:/Users/court/Documents/filtered_data_total_june.csv", row.names = FALSE)

# We reload the fully filtered dataset from disk if needed (adapt the path)
filtered_data_total <-
read.csv2("C:/Users/court/Documents/filtered_data_total_june.csv", sep=",",
header = TRUE)
```

A.8.3. Resampling of the dataset

```
############## RESAMPLING AND CALCULATION OF MOVEMENT STATISTICS ##############

# If not done already, download the required package
#install.packages("amt")
#install.packages("dplyr")
#install.packages("purrr")
#install.packages("lubridate")


# Load the library
library(amt)
library(dplyr)
library(purrr)
library(lubridate)

# We create a track object from the filtered data, specifying coordinates,
time, and identifiers
data_track <- make_track(filtered_data_total, utm_e, utm_n, WAT, collar_ID =
collar_ID, collar = collar, burst_ID = burst_ID, season = season, crs = 32634)

# We check the structure of the track object
str(data_track)

# We resample the track data by burst_ID to a fixed 4-hour interval with a 10-
minute tolerance
track_resampled <- data_track %>%
  group_by(burst_ID) %>%                      # We group by animal burst
  arrange(t_) %>%                             # We sort each group
chronologically
  group_split() %>%                           # We split into a list by groups
  map(~ track_resample(.x, rate = hours(4), tolerance = minutes(10))) %>% # We
resample each group's data
  bind_rows() %>%                            # We recombine the results into a
single dataframe
  mutate(
    burst2 = cumsum(burst_ != lag(burst_, default = first(burst_)) |
                    burst_ID != lag(burst_ID, default = first(burst_ID))) + 1
```

```
# We create a new burst counter starting at 1
  ) %>%
  select(-burst_) %>%                          # We remove the old 'burst_'
column
  rename(burst_ = burst2)                      # We rename 'burst2' to 'burst_'

# We ensure coordinates are numeric
track_resampled <- track_resampled %>%
  mutate(
    x_ = as.numeric(x_),
    y_ = as.numeric(y_)
  )

# We check the structure of the resampled track data
str(track_resampled)
```

A.9. Extraction of movement statistics

```
# We calculate step-level data (movement steps grouped by burst)
data3 <- steps_by_burst(track_resampled)

# We check the structure of the steps data
str(data3)

# We rename columns in the track to prepare for joining with steps data
track_resampled <- track_resampled %>%
  rename(
    x1_ = x_,
    y1_ = y_,
    t1_ = t_
  )

# We join the steps data with the track data to include additional attributes
like season
data4 <- data3 %>%
  left_join(
    track_resampled %>% select(x1_, y1_, t1_, burst_, burst_ID, season),
    by = c("x1_", "y1_", "t1_", "burst_"),
  )

# We check the structure and unique seasons present in the final dataset
str(data4)
unique(data4$season)

# We save the processed steps data as an RDS file for future use (adapt the
path)
# saveRDS(data4, "C:/Users/court/Documents/steps_390k.rds")

# We reload the steps data from the RDS file if needed (adapt the path)
data4 <- readRDS("C:/Users/court/Documents/project/steps_390k.rds")
```

```
########## Division in seasons ##########
# We extract the data of each season to put them as individual datsets
cool <- data4 %>%
  filter(season == "cool_dry")

rainy <- data4 %>%
  filter(season == "rainy")

hot <- data4 %>%
  filter(season == "hot_dry")

# We save the results (adapt the paths)
saveRDS(hot, "C:/Users/court/Documents/steps_390k_hot.rds")
saveRDS(rainy, "C:/Users/court/Documents/steps_390k_rainy.rds")
saveRDS(cool, "C:/Users/court/Documents/steps_390k_cool.rds")
```

A.10. Creation of available steps

A.10.1. Creation of parametric curves for the relative turning angles

```
################################################################################
####### Calculation of fitted parametric bimodal von Mises distribution #######
################################################################################

# We download the packages
install.packages("movMF")
install.packages("dplyr")
install.packages("circular")

# We load the required libraries
library(dplyr)
library(movMF)
library(circular)

# We load the datasets (adapt the paths)
hot <- readRDS("C:/Users/court/Documents/steps_390k_hot.rds")
rainy <- readRDS("C:/Users/court/Documents/steps_390k_rainy.rds")
cool <- readRDS("C:/Users/court/Documents/steps_390k_cool.rds")

# 1. We define a generic function to fit a von Mises mixture model
fit_vonmises_mixture <- function(df,
                                 angle_col = "ta_", # name of the column
containing angles
                                 k = 2,          # number of mixture components
                                 kappa_init = 5, # initial kappa value
                                 seed = 123) {   # seed for reproducibility
  # 1.1. We extract the angle column and convert it to circular format in
radians
  ta_all <- circular(df[[angle_col]], units = "radians")

  # 1.2. We convert to numeric and remove NA values
```

```
  theta <- as.numeric(ta_all)
  theta <- theta[!is.na(theta)]

  # 1.3. We project angles onto the unit circle (cos, sin)
  X <- cbind(cos(theta), sin(theta))

  # 1.4. We manually initialize the parameters for the mixture components
  init_par <- rbind(
    kappa_init * c(1, 0),
    kappa_init * c(-1, 0)
  )

  # 1.5. We fit the von Mises mixture model with the specified number of
components
  set.seed(seed)
  movMF(X, k = k, par = init_par)
}

# 2. We retrieve the data.frames by their names (assuming 'hot', 'rainy', and
'cool' exist in the environment)
dfs <- mget(c("hot", "rainy", "cool"))

# 3. We apply the fitting function to each dataset
fits <- lapply(dfs, fit_vonmises_mixture)

# We save the fitted models for future use (adapt the paths)
# saveRDS(fits$hot, "C:/Users/court/Documents/von_mise_hot.rds")
# saveRDS(fits$rainy, "C:/Users/court/Documents/von_mise_rainy.rds")
# saveRDS(fits$cool, "C:/Users/court/Documents/von_mise_cool.rds")
```

A.10.2. Extraction of the available steps

```
################################################################
################## CREATE RANDOM STEPS #########################
################################################################

# We download the packages
install.packages("amt")
install.packages("dplyr")
install.packages("circular")

# We load the required libraries
library(circular)
library(dplyr)
library(amt)

# We load the fitted von Mises mixture models for each season (adapt the paths)
angle_hot <- readRDS("C:/Users/court/Documents/von_mise_hot.rds")
angle_rainy <- readRDS("C:/Users/court/Documents/von_mise_rainy.rds")
angle_cool <- readRDS("C:/Users/court/Documents/von_mise_cool.rds")

# We define a function to simulate turning angles from a von Mises mixture
distribution
```

```
simulate_turning_angles <- function(angle_distr, N) {
  # We extract mixture weights and parameter matrix theta
  alphas    <- angle_distr$alpha      # vector of mixture weights
  theta_mat <- angle_distr$theta      # 2x2 matrix: each row = kappa *
(cos(mu), sin(mu))

  # Helper function to extract mean direction (mu) and concentration (kappa)
from vector
  extract_vm <- function(vec) {
    κ     <- sqrt(sum(vec^2))
    μ_rad <- atan2(vec[2], vec[1])
    list(mu    = circular(μ_rad, units = "radians"),
         kappa = κ)
  }

  vm1 <- extract_vm(theta_mat[1, ])
  vm2 <- extract_vm(theta_mat[2, ])
  p1  <- alphas[1]

  # Internal function to generate random samples from the mixture
  rand_ta_mix <- function(n) {
    comps <- rbinom(n, size = 1, prob = p1)
    out   <- numeric(n)
    n1    <- sum(comps == 1)
    if (n1 > 0) out[comps == 1] <- rvonmises(n1, vm1$mu, vm1$kappa)
    if (n - n1 > 0) out[comps == 0] <- rvonmises(n - n1, vm2$mu, vm2$kappa)
    out
  }

  # We draw samples and wrap angles into [-π, π]
  raw <- as.numeric(rand_ta_mix(N))
  recaled <- (raw + pi) %% (2*pi) - pi
  circular(recaled)
}

# Number of values generated for each piece of data (must equal the number of
available steps generated
n_ctrl <- 10

# We define the number of values that must be generated for each dataset
Ns <- c(
  hot   = nrow(hot)   * n_ctrl,
  rainy = nrow(rainy) * n_ctrl,
  cool  = nrow(cool)  * n_ctrl
)

fits <- list(
  hot   = angle_hot,
  rainy = angle_rainy,
  cool  = angle_cool
)

# We apply the simulation function in parallel and get a list of circular angle
vectors
rand_angles <- mapply(
```

```
  FUN       = simulate_turning_angles,
  angle_distr = fits,
  N           = Ns,
  SIMPLIFY    = FALSE
)

# We extract the random angles generated for each season
rand_hot   <- rand_angles$hot
rand_rainy <- rand_angles$rainy
rand_cool  <- rand_angles$cool

# We set a seed for reproducibility
set.seed(481226)

# We generate random steps using the random turning angles and control number
random_hot <- random_steps(hot, n_control = 10, rand_ta = rand_hot)
random_hot2 <- random_hot %>%
  mutate(log_sl_ = log(sl_), cos_ta_ = cos(ta_))

random_rainy <- random_steps(rainy, n_control = 10, rand_ta = rand_rainy)
random_rainy2 <- random_rainy %>%
  mutate(log_sl_ = log(sl_), cos_ta_ = cos(ta_))

random_cool <- random_steps(cool, n_control = 10, rand_ta = rand_cool)
random_cool2 <- random_cool %>%
  mutate(log_sl_ = log(sl_), cos_ta_ = cos(ta_))

# We save the processed random steps for future use (adapt the path)
# saveRDS(random_hot2, "C:/Users/court/Documents/random_hot.rds")
# saveRDS(random_rainy2, "C:/Users/court/Documents/random_rainy.rds")
# saveRDS(random_cool2, "C:/Users/court/Documents/random_cool.rds")

# We reload the processed random steps if needed (adapt the path)
random_hot2 <- readRDS("C:/Users/court/Documents/random_hot.rds")
random_rainy2 <- readRDS("C:/Users/court/Documents/random_rainy.rds")
random_cool2 <- readRDS("C:/Users/court/Documents/random_cool.rds")
```

A.11. Extraction on the non-time-dependent indicators

```
#################################################################
################## COVARIATES EXTRACTION ########################
#################################################################

# We download the packages
install.packages("terra")
install.packages("dplyr")
install.packages("purrr")
install.packages("tidyr")
install.packages("lubridate")
```

```
# We load the libraries
library(terra)
library(dplyr)
library(tidyr)
library(purrr)
library(lubridate)

# We load all the raster layers representing covariates (adapt the paths)
probability_settlements <-
rast("C:/Users/court/Documents/probability_settlements.tif")
small_tpi <- rast("C:/Users/court/Documents/small_tpi.tif")
large_tpi <- rast("C:/Users/court/Documents/large_tpi.tif")
TCI_3x3 <- rast("C:/Users/court/Documents/convergence_index.tif")
TCI_7x7 <- rast("C:/Users/court/Documents/convergence_index_k7.tif")
fire_20 <- rast("C:/Users/court/Documents/FireFreq_total_2004_2023.tif")
fire_2016 <- rast("C:/Users/court/Docuemnts/fires/Fire_yearly_2016.tif")
fire_2017 <- rast("C:/Users/court/Documents/fires/Fire_yearly_2017.tif")
fire_2018 <- rast("C:/Users/court/Documents/fires/Fire_yearly_2018.tif")
fire_2019 <- rast("C:/Users/court/Documents/fires/Fire_yearly_2019.tif")
fire_2020 <- rast("C:/Users/court/Documents/fires/Fire_yearly_2020.tif")
fire_2021 <- rast("C:/Users/court/Documents/fires/Fire_yearly_2021.tif")
fire_2022 <- rast("C:/Users/court/Documents/fires/Fire_yearly_2022.tif")
shrub_density <- rast("C:/Users/court/Documents/MSAVI_sum_2021to2023.tif")

# We project shrub density and fire frequency rasters to the CRS of small_tpi
using appropriate methods
shrub_density_utm <- project(shrub_density, small_tpi, method = "bilinear")
fire20_utm <- project(fire_20, small_tpi, method = "near")

# We resample rasters to align them spatially to the small_tpi raster grid
probability_settlements_aligned <- resample(probability_settlements, small_tpi,
method = "bilinear")
shrub_aligned <- resample(shrub_density_utm, small_tpi, method = "bilinear")
fire_20_aligned <- resample(fire_20, small_tpi, method = "near")
aligned_fire_2016 <- resample(fire_2016, small_tpi, method = "near")
aligned_fire_2017 <- resample(fire_2017, small_tpi, method = "near")
aligned_fire_2018 <- resample(fire_2018, small_tpi, method = "near")
aligned_fire_2019 <- resample(fire_2019, small_tpi, method = "near")
aligned_fire_2020 <- resample(fire_2020, small_tpi, method = "near")
aligned_fire_2021 <- resample(fire_2021, small_tpi, method = "near")
aligned_fire_2022 <- resample(fire_2022, small_tpi, method = "near")

# We rename the layers for more clarity
names(probability_settlements_aligned) <- "proba_human"
names(small_tpi) <- "TPI_3x3"
names(large_tpi) <- "TPI_7x7"
names(TCI_3x3) <- "TCI_3x3"
names(TCI_7x7) <- "TCI_7x7"
names(fire_20_aligned) <- "fire_20y"
names(aligned_fire_2016) <- "fire_2016"
names(aligned_fire_2017) <- "fire_2017"
names(aligned_fire_2018) <- "fire_2018"
names(aligned_fire_2019) <- "fire_2019"
names(aligned_fire_2020) <- "fire_2020"
```

```
names(aligned_fire_2021) <- "fire_2021"
names(aligned_fire_2022) <- "fire_2022"
names(shrub_aligned) <- "shrub_density"

# We combine all aligned rasters into a single raster stack
rasters <- c(small_tpi, large_tpi, probability_settlements_aligned, TCI_3x3,
TCI_7x7, fire_20_aligned, aligned_fire_2016, aligned_fire_2017,
  aligned_fire_2018, aligned_fire_2019, aligned_fire_2020, aligned_fire_2021,
aligned_fire_2022, shrub_aligned)

# 1. We define a function to extract covariates from points and process them
process_covariates <- function(random_points, rasters) {
  # We extract all covariates at given points
  covariates <- extract_covariates(random_points, rasters)

  # We fix negative values in shrub_density by replacing them with 0
  cov2 <- covariates %>%
    mutate(shrub_density = if_else(shrub_density < 0, 0, shrub_density))

  # We calculate fire_prev_year from the corresponding previous year's fire
column
  cov_fires <- cov2 %>%
    mutate(
      fire_prev_year = pmap_dbl(
        list(
          year = as.integer(format(t2_, "%Y")),
          row  = row_number()
        ),
        function(year, row) {
          colname <- paste0("fire_", year - 1)
          covariates[[colname]][row]  # Lookup fire data from the initial
covariate dataset
        }
      )
    ) %>%
    # We replace NA by 0 and coerce to binary 0/1 (forcing any 2 to 1)
    mutate(
      fire_prev_year = replace_na(fire_prev_year, 0),
      fire_prev_year = as.numeric(fire_prev_year),
      fire_prev_year = if_else(fire_prev_year == 2, 1, fire_prev_year)
    )

  return(cov_fires)
}

# 2. We manually apply the function to the random steps of each season
cov_hot   <- process_covariates(random_hot2, rasters)
cov_rainy <- process_covariates(random_rainy2, rasters)
cov_cool  <- process_covariates(random_cool2, rasters)

# We create a new date column shifted 10 days back from t2_ for each season
cov_hot <- cov_hot %>%
  mutate(date_10 = t2_ - lubridate::days(10))

cov_rainy <- cov_rainy %>%
```

```
  mutate(date_10 = t2_ - lubridate::days(10))

cov_cool <- cov_cool %>%
  mutate(date_10 = t2_ - lubridate::days(10))

# We order the data by step_id_ and case_, then add a unique id per row
cov_hot <- cov_hot %>%
  arrange(step_id_, case_) %>%
  mutate(id = row_number())

cov_rainy <- cov_rainy %>%
  arrange(step_id_, case_) %>%
  mutate(id = row_number())

cov_cool <- cov_cool %>%
  arrange(step_id_, case_) %>%
  mutate(id = row_number())

# We save the processed covariates datasets for rainy and cool seasons (hot is
commented out)
# saveRDS(cov_hot, "C:/Users/court/Documents/cov_hot.rds")
saveRDS(cov_rainy, "C:/Users/court/Documents/cov_rainy.rds")
saveRDS(cov_cool, "C:/Users/court/Documents/cov_cool.rds")

# We load the saved covariates for rainy and cool seasons if needed (hot is
commented out)
# cov_hot <- readRDS("C:/Users/court/Documents/cov_hot.rds")
cov_rainy <- readRDS("C:/Users/court/Documents/cov_rainy.rds")
cov_cool <- readRDS("C:/Users/court/Documents/cov_cool.rds")

# We export the covariates to CSV files for further use or inspection
write.csv(cov_hot, "C:/Users/court/Documents/cov_hot.csv", row.names = FALSE)
write.csv(cov_rainy, "C:/Users/court/Documents/cov_rainy.csv", row.names =
FALSE)
write.csv(cov_cool, "C:/Users/court/Docuemnts/cov_cool.csv", row.names = FALSE)
```

A.12. Dynamic extraction of time-dependent indicators

A.12.1. Extraction of MSAVI Landsat

```
// We import the FeatureCollection from the assets
// Apply the code to the three seasonal datasets
var table = ee.FeatureCollection("projects/ee-courtjulien5/assets/cov_cool");

// We define the UTM zone 34N projection (EPSG:32634)
var utmProjection = ee.Projection('EPSG:32634');

// We define a function to compute MSAVI for each feature
var getMSAVI = function(feature) {
  // We retrieve raw x2_ and y2_ coordinate values
```

```
  var x2 = feature.get('x2_');
  var y2 = feature.get('y2_');

  // We check if coordinates are "NA" strings and mark as invalid if so
  var invalid = ee.Algorithms.If(
    ee.Algorithms.IsEqual(x2, "NA"),
    true,
    ee.Algorithms.If(ee.Algorithms.IsEqual(y2, "NA"), true, false)
  );
  var validCoords = ee.Algorithms.If(invalid, false, true);

  // If coordinates are valid, we proceed; otherwise, we set MSAVI and
MSAVI_date_gap to null
  return ee.Algorithms.If(validCoords, (function() {
    // We convert coordinates to numbers
    var utm_e = ee.Number.parse(x2);
    var utm_n = ee.Number.parse(y2);

    // We create a point geometry from the coordinates using UTM projection
    var point = ee.Geometry.Point([utm_e, utm_n], utmProjection);

    // We get the date of the feature and format it in ISO standard
    var date = ee.Date(feature.getString('t2_').replace(' ', 'T'));

    // We filter the Landsat 8 Surface Reflectance (Level 2) image collection
    // by the point location and by date +/- 15 days around the target date
    var l8_sr = ee.ImageCollection('LANDSAT/LC08/C02/T1_L2')
                  .filterBounds(point)
                  .filterDate(date.advance(-15, 'day'), date.advance(15,
'day'))
                  .map(function(image) {
                    // We compute the absolute difference in days to the target
date
                    var diff =
ee.Date(image.get('system:time_start')).difference(date, 'day').abs();
                    return image.set('date_diff', diff);
                  })
                  .sort('date_diff'); // We sort images by proximity to target
date

    // We count how many images are available
    var numImages = l8_sr.size();

    // If at least one image is found, we proceed with calculations
    return ee.Algorithms.If(numImages.gt(0), (function() {
      // We get the closest image in time
      var closestImage = ee.Image(l8_sr.first());

      // We convert raw digital numbers to reflectance using scaling
coefficients
      var scaledImage = closestImage.multiply(0.0000275).add(-0.2);

      // We calculate MSAVI using the formula:
      // MSAVI = (2 * NIR + 1 - sqrt((2 * NIR + 1)^2 - 8 * (NIR - RED))) / 2
      var msavi = scaledImage.expression(
```

```
        '((2 * NIR + 1 - sqrt((2 * NIR + 1) ** 2 - 8 * (NIR - RED))) / 2)', {
          'NIR': scaledImage.select('SR_B5'),  // Near Infrared band
          'RED': scaledImage.select('SR_B4')   // Red band
        }).rename('MSAVI');

      // We calculate the absolute date difference between image and target
date
      var imageDate = ee.Date(closestImage.get('system:time_start'));
      var dateDifference = date.difference(imageDate, 'days').abs();

      // We add MSAVI mean value at the point and the date gap as properties to
the feature
      return feature.set({
        'MSAVI': msavi.reduceRegion({
          reducer: ee.Reducer.mean(),
          geometry: point,
          scale: 30,
          maxPixels: 1e9
        }).get('MSAVI'),
        'MSAVI_date_gap': dateDifference
      });
    })(), feature.set({'MSAVI': null, 'MSAVI_date_gap': null}));

  })(), feature.set({'MSAVI': null, 'MSAVI_date_gap': null}));
};

// We apply the MSAVI computation function to each feature in the table
var tableWithMSAVI = table.map(getMSAVI);

// We export the updated FeatureCollection as CSV to Google Drive
Export.table.toDrive({
  collection: tableWithMSAVI,
  description: 'oryx_filtered_with_MSAVI_Landsat',
  fileFormat: 'CSV'
});
```

A.12.2. Extraction of MSAVI (MODIS)

```
// We load the FeatureCollection from the assets
// Apply the code to the three seasonal datasets by changing the name
var table = ee.FeatureCollection("projects/ee-courtjulien5/assets/cov_cool");

// We define the UTM zone 34N projection (EPSG:32634)
var utmProjection = ee.Projection('EPSG:32634');

// We define a function to compute MSAVI from MODIS for each feature by testing
multiple images
var getMSAVI = function(feature) {
  // We retrieve coordinates x2_ and y2_
  var x2 = feature.get('x2_');
  var y2 = feature.get('y2_');

  // We check if the coordinates are "NA" strings and mark as invalid if so
  var invalid = ee.Algorithms.If(
```

```
    ee.Algorithms.IsEqual(x2, "NA"),
    true,
    ee.Algorithms.If(ee.Algorithms.IsEqual(y2, "NA"), true, false)
  );
  var validCoords = ee.Algorithms.If(invalid, false, true);

  // If coordinates are valid, we proceed with processing; otherwise, we set
MSAVI_modis and gap to null
  return ee.Algorithms.If(validCoords, (function() {
    // We parse coordinates to numbers and create a point geometry in UTM
    var utm_e = ee.Number.parse(x2);
    var utm_n = ee.Number.parse(y2);
    var point = ee.Geometry.Point([utm_e, utm_n], utmProjection);

    // We get the target date and format it to ISO standard
    var date = ee.Date(feature.getString('t2_').replace(' ', 'T'));

    // We load the MODIS Terra and Aqua Surface Reflectance collections
    var terra = ee.ImageCollection('MODIS/061/MOD09GQ');
    var aqua  = ee.ImageCollection('MODIS/061/MYD09GQ');

    // We merge the two collections, filter by location and +/- 3 days from
target date,
    // and assign a 'date_diff' property representing absolute difference in
days to target date
    var modis = terra.merge(aqua)
      .filterBounds(point)
      .filterDate(date.advance(-3, 'day'), date.advance(3, 'day'))
      .map(function(image) {
        var diff = ee.Date(image.get('system:time_start')).difference(date,
'day').abs();
        return image.set('date_diff', diff);
      })
      .sort('date_diff');

    // We limit the maximum number of candidate images to test
    var maxCandidates = 5;
    var candList = modis.toList(maxCandidates);

    // We initialize a dictionary to store results of the iteration
    var init = ee.Dictionary({
      'msavi': null,
      'gap': null,
      'found': false
    });

    // We iterate over candidates to find the first image with a valid MSAVI
value
    var result = ee.List.sequence(0, maxCandidates - 1).iterate(
      function(idx, prevDict) {
        prevDict = ee.Dictionary(prevDict);
        // If a valid MSAVI has already been found, return previous dictionary
unchanged
        return ee.Algorithms.If(prevDict.get('found'),
          prevDict,
```

```
          (function() {
            // Otherwise, process the image at index idx
            var img = ee.Image(candList.get(idx));
            var scaled = img.multiply(0.0001);
            // We compute MSAVI_modis using bands sur_refl_b02 (NIR) and
sur_refl_b01 (RED)
            var ms = scaled.expression(
              '((2 * NIR + 1 - sqrt((2 * NIR + 1) ** 2 - 8 * (NIR - RED))) /
2)', {
                'NIR': scaled.select('sur_refl_b02'),
                'RED': scaled.select('sur_refl_b01')
              }
            ).rename('MSAVI_modis');

            // We extract the mean MSAVI value at the point
            var val = ms.reduceRegion({
              reducer: ee.Reducer.mean(),
              geometry: point,
              scale: 250,
              maxPixels: 1e9
            }).get('MSAVI_modis');

            // We compute the absolute date gap between the image and the
target date
            var gap = date.difference(ee.Date(img.get('system:time_start')),
'day').abs();

            // We check if the MSAVI value is not null
            var found = ee.Algorithms.If(
              ee.Algorithms.IsEqual(val, null),
              false,
              true
            );

            // We return a dictionary with the current MSAVI value, gap, and
found status
            return ee.Dictionary({
              'msavi': val,
              'gap': gap,
              'found': found
            });
          })()
        );
      },
      init
    );

    // We get the final dictionary from the iteration
    var dict = ee.Dictionary(result);

    // We set the MSAVI_modis and MSAVI_date_gap_modis properties to the
feature
    return feature.set({
      'MSAVI_modis': dict.get('msavi'),
      'MSAVI_date_gap_modis': dict.get('gap')
```

```
    });

  })(),
  // If coordinates are invalid, we set the properties to null
  feature.set({
    'MSAVI_modis': null,
    'MSAVI_date_gap_modis': null
  })
  );
};

// We apply the MSAVI calculation function to each feature in the collection
var tableWithMSAVI = table.map(getMSAVI);

// We export the resulting FeatureCollection with MSAVI values to Google Drive
as CSV
Export.table.toDrive({
  collection: tableWithMSAVI,
  description: 'oryx_filtered_with_MSAVI_MODIS',
  fileFormat: 'CSV'
});
```

A.12.3. Extraction of D-MSAVI

```
// We load the FeatureCollection from the assets
// We change the name of the assets to apply the code on each seasonal dataset
var table = ee.FeatureCollection("projects/ee-courtjulien5/assets/cov_cool");

// We define the UTM zone 34N projection (EPSG:32634)
var utmProjection = ee.Projection('EPSG:32634');

// We define a function to compute MSAVI from MODIS 10-day composites,
// testing multiple images to find the closest with valid data
var getMSAVI = function(feature) {
  // We retrieve the coordinate values
  var x2 = feature.get('x2_');
  var y2 = feature.get('y2_');

  // We check for invalid coordinates labeled as "NA"
  var invalid = ee.Algorithms.If(
    ee.Algorithms.IsEqual(x2, "NA"),
    true,
    ee.Algorithms.If(ee.Algorithms.IsEqual(y2, "NA"), true, false)
  );
  var validCoords = ee.Algorithms.If(invalid, false, true);

  // If coordinates are valid, we proceed with calculations; else we set nulls
  return ee.Algorithms.If(validCoords, (function() {
    // We parse coordinates to numeric values and create a point geometry in
UTM
    var utm_e = ee.Number.parse(x2);
    var utm_n = ee.Number.parse(y2);
    var point = ee.Geometry.Point([utm_e, utm_n], utmProjection);
```

```
    // We get the target date and convert to ISO format
    var date = ee.Date(feature.getString('date_10').replace(' ', 'T'));

    // We load MODIS Terra and Aqua 10-day surface reflectance collections
    var terra = ee.ImageCollection('MODIS/061/MOD09GQ');
    var aqua  = ee.ImageCollection('MODIS/061/MYD09GQ');

    // We merge the collections, filter spatially and temporally (+/-3 days),
    // compute absolute date difference for each image, and sort by proximity
    var modis = terra.merge(aqua)
      .filterBounds(point)
      .filterDate(date.advance(-3, 'day'), date.advance(3, 'day'))
      .map(function(image) {
        var diff = ee.Date(image.get('system:time_start')).difference(date,
'day').abs();
        return image.set('date_diff', diff);
      })
      .sort('date_diff');

    // We set the maximum number of candidate images to test
    var maxCandidates = 5;
    var candList = modis.toList(maxCandidates);

    // We initialize a dictionary to store results of the iteration
    var init = ee.Dictionary({
      'msavi': null,
      'gap': null,
      'found': false
    });

    // We iterate through candidates to find the first with a valid MSAVI value
    var result = ee.List.sequence(0, maxCandidates - 1).iterate(
      function(idx, prevDict) {
        prevDict = ee.Dictionary(prevDict);
        // If a valid MSAVI was already found, we return previous results
unchanged
        return ee.Algorithms.If(prevDict.get('found'),
          prevDict,
          (function() {
            var img = ee.Image(candList.get(idx));
            var scaled = img.multiply(0.0001);
            // We calculate MSAVI_modis using bands sur_refl_b02 (NIR) and
sur_refl_b01 (RED)
            var ms = scaled.expression(
              '((2 * NIR + 1 - sqrt((2 * NIR + 1) ** 2 - 8 * (NIR - RED))) /
2)', {
                'NIR': scaled.select('sur_refl_b02'),
                'RED': scaled.select('sur_refl_b01')
              }
            ).rename('MSAVI_10_modis');

            // We extract the mean MSAVI value at the point
            var val = ms.reduceRegion({
              reducer: ee.Reducer.mean(),
```

```
              geometry: point,
              scale: 250,
              maxPixels: 1e9
            }).get('MSAVI_10_modis');

            // We calculate the absolute date gap
            var gap = date.difference(ee.Date(img.get('system:time_start')),
'day').abs();

            // We determine if a valid MSAVI value was found
            var found = ee.Algorithms.If(ee.Algorithms.IsEqual(val, null),
false, true);

            // We return updated dictionary with current candidate's results
            return ee.Dictionary({
              'msavi': val,
              'gap': gap,
              'found': found
            });
          })()
        );
      },
      init
    );

    // We get the final dictionary from the iteration
    var dict = ee.Dictionary(result);

    // We set MSAVI and gap properties on the feature
    return feature.set({
      'MSAVI_10_modis': dict.get('msavi'),
      'MSAVI_10_date_gap_modis': dict.get('gap')
    });

  })(),
  // If coordinates are invalid, we set MSAVI and gap to null
  feature.set({
    'MSAVI_10_modis': null,
    'MSAVI_10_date_gap_modis': null
  })
  );
};

// We map the function over each feature in the table
var tableWithMSAVI = table.map(getMSAVI);

// We export the resulting FeatureCollection as a CSV to Google Drive
Export.table.toDrive({
  collection: tableWithMSAVI,
  description: 'oryx_filtered_with_MSAVI_10_MODIS',
  fileFormat: 'CSV'
});
```

A.12.4. Merging of the data by season

```
#################################################################
#################### IMPORT GEE DATA ############################
#################################################################

# We download the packages
install.packages("terra")

# Load the library
library(terra)

# We define a function to merge covariate data with Landsat and MODIS MSAVI
values

# The first argument is the dataset before export to GEE
# the second is the dataset resulting from the MSAVI extraction with Landsat,
# the third is the dataset resulting from the MSAVI extraction with Modis
# and the fourth is the dataset resulting from the extraction of the D-MSAVI
merge_MSAVI <- function(cov, landsat, MSAVI_modis, date_10) {
  # We select only the relevant columns for joining
  landsat_sel <- dplyr::select(landsat, id, MSAVI)
  MSAVI_sel   <- dplyr::select(MSAVI_modis, id, MSAVI_modis)
  date_10_sel <- dplyr::select(date_10, id, MSAVI_10_modis)

  # We perform successive left joins to merge all MSAVI data into the
covariates table
  res <- cov %>%
    left_join(landsat_sel, by = "id") %>%
    left_join(MSAVI_sel, by = "id") %>%
    left_join(date_10_sel, by = "id")

  # We convert MSAVI columns to numeric and compute the difference between
MODIS versions
  res <- res %>%
    mutate(
      MSAVI = as.numeric(MSAVI),
      MSAVI_modis = as.numeric(MSAVI_modis),
      MSAVI_10_modis = as.numeric(MSAVI_10_modis),
      D_MSAVI = MSAVI_modis - MSAVI_10_modis
    )

  return(res)
}

# We import Landsat, MODIS, and 10-day MODIS MSAVI CSV files for the hot season
# Adapt the path
landsat_hot <-
read.csv2("C:/Users/court/Downloads/oryx_filtered_with_MSAVI_Landsat_hot.csv",
sep=",", header = TRUE)
MSAVI_hot <-
read.csv2("C:/Users/court/Downloads/oryx_filtered_with_MSAVI_MODIS_hot.csv",
sep=",", header = TRUE)
date_10_hot <-
read.csv2("C:/Users/court/Downloads/oryx_filtered_with_MSAVI_10_MODIS_hot.csv",
sep=",", header = TRUE)
```

```
# We merge the hot season covariates with the MSAVI datasets
cov_hot_GEE <- merge_MSAVI(cov_hot, landsat_hot, MSAVI_hot, date_10_hot)

# We save the merged hot season covariates if needed (adapt the path)
# saveRDS(cov_hot_GEE, "C:/Users/court/Documents/cov_hot_GEE.rds")

# We repeat the import and merging process for the rainy season
landsat_rainy <-
read.csv2("C:/Users/court/Downloads/oryx_filtered_with_MSAVI_Landsat_rainy.csv"
, sep=",", header = TRUE)
MSAVI_rainy <-
read.csv2("C:/Users/court/Downloads/oryx_filtered_with_MSAVI_MODIS_rainy.csv",
sep=",", header = TRUE)
date_10_rainy <-
read.csv2("C:/Users/court/Downloads/oryx_filtered_with_MSAVI_10_MODIS_rainy.csv
", sep=",", header = TRUE)

cov_rainy_GEE <- merge_MSAVI(cov_rainy, landsat_rainy, MSAVI_rainy,
date_10_rainy)

# We save the merged rainy season covariates if needed (adapt the path)
# saveRDS(cov_rainy_GEE, "C:/Users/court/Documents/cov_rainy_GEE.rds")


# We repeat the import and merging process for the cool season
landsat_cool <-
read.csv2("C:/Users/court/Downloads/oryx_filtered_with_MSAVI_Landsat_cool.csv",
sep=",", header = TRUE)
MSAVI_cool <-
read.csv2("C:/Users/court/Downloads/oryx_filtered_with_MSAVI_MODIS_cool.csv",
sep=",", header = TRUE)
date_10_cool <-
read.csv2("C:/Users/court/Downloads/oryx_filtered_with_MSAVI_10_MODIS_cool.csv"
, sep=",", header = TRUE)

# We save the merged cool season covariates if needed (adapt the path)
cov_cool_GEE <- merge_MSAVI(cov_cool, landsat_cool, MSAVI_cool, date_10_cool)

# saveRDS(cov_cool_GEE, "C:/Users/court/Documents/cov_cool_GEE.rds")
```

A.13. Pair-to-pair collinearity assessment

A.13.1. Code used for the creation of the collinearity assessment

```
############################################################
########### Pair-to-pair collinearity assessement ###########
############################################################

#install.packages("corrplot")

library(corrplot)

# We create a function to calculate the correlation matrix of selected
```

```
variables,
# save it as a CSV file, and generate a correlation plot PNG with a season-
specific title
save_corr_matrix_and_plot <- function(data, suffix, vars, output_dir) {
  # We calculate the correlation matrix using complete observations only
  corr_matrix <- cor(
    data[, vars],
    use = "complete.obs"
  )

  # We export the correlation matrix to a CSV file
  write.csv(
    corr_matrix,
    file = file.path(output_dir, paste0("correlation_matrix_", suffix, ".csv"))
  )

  # We open a PNG device to save the correlation plot
  png(
    file.path(output_dir, paste0("corrplot_", suffix, ".png")),
    width = 1000, height = 1000
  )

  # We create a color correlation plot with coefficients and custom formatting
  corrplot(
    corr_matrix,
    method     = "color",
    type       = "lower",
    addCoef.col= "black",
    tl.cex     = 2.3,
    number.cex = 2.3,
    cl.cex     = 2.3,
    col        = colorRampPalette(c("blue","white","red"))(200),
    diag       = FALSE
  )

  # We add a main title based on the suffix indicating the season
  main_title <- switch(
    suffix,
    "hot"   = "Hot dry season",
    "rainy" = "Rainy season",
    "cool"  = "Cool dry season",
    "Season"
  )
  title(main = main_title, cex.main = 3)

  # We close the PNG device
  dev.off()
}

# We define the list of variables for which to compute correlations
vars <- c(
  "D_MSAVI",
  "MSAVI",
  "MSAVI_modis",
  "shrub_density",
```

```
  "TPI_3x3",
  "TPI_7x7",
  "TCI_3x3",
  "TCI_7x7",
  "proba_human",
  "fire_20y",
  "fire_prev_year"
)

# We set the output directory for saving CSV and PNG files (adapt the path)
output_dir <- "C:/Users/court/Documents"

# We call the function for each seasonal dataset to save correlation matrices
and plots
save_corr_matrix_and_plot(cov_hot_GEE,   "hot",   vars, output_dir)
save_corr_matrix_and_plot(cov_rainy_GEE, "rainy", vars, output_dir)
save_corr_matrix_and_plot(cov_cool_GEE,  "cool",  vars, output_dir)
```

A.13.2. Results of the pair-to-pair collinearity assessment

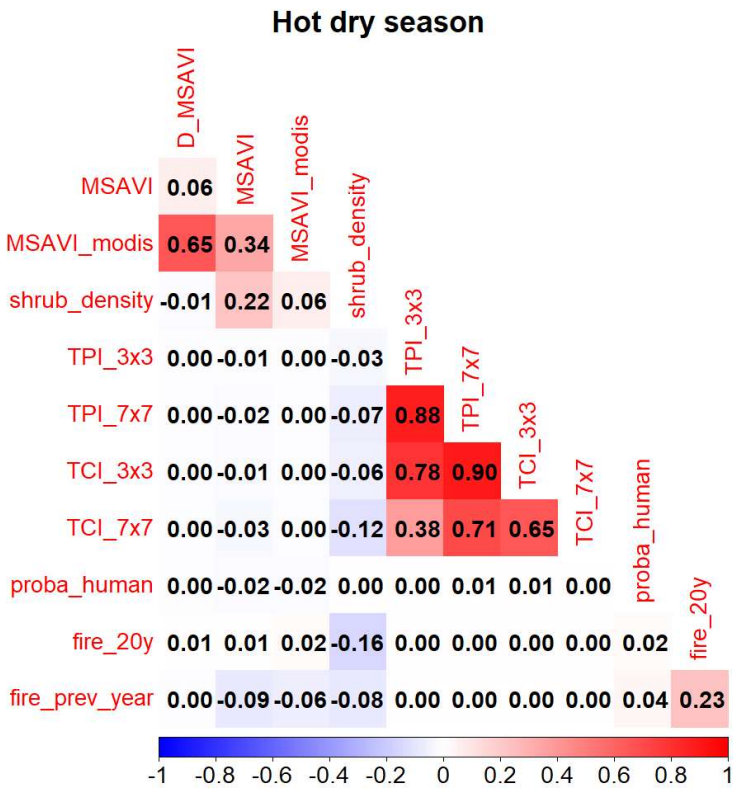

Figure 26: Pair-to-pair correlation matrix for the hot dry season

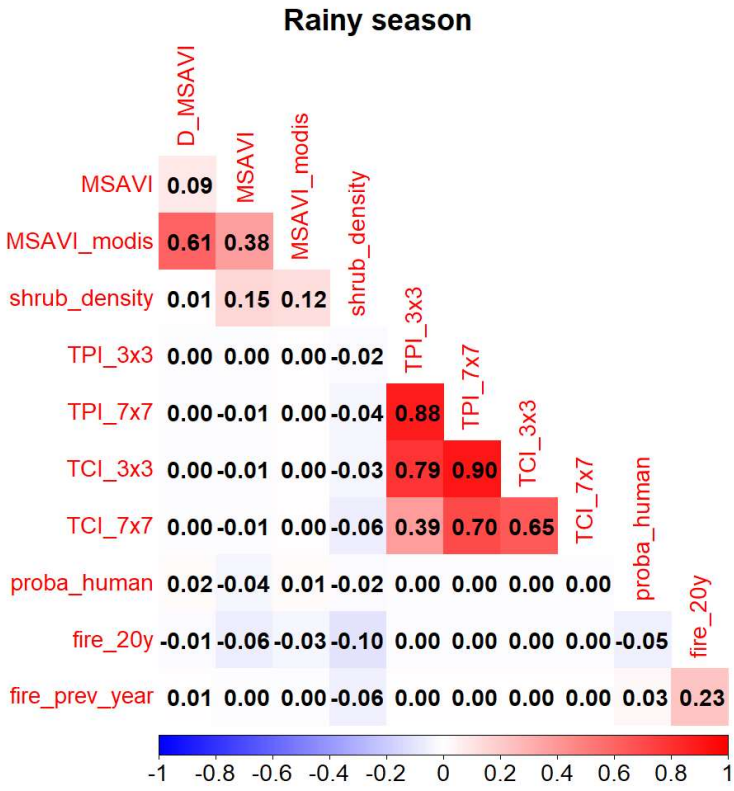

Figure 27: Pair-to-pair correlation matrix for the rainy season

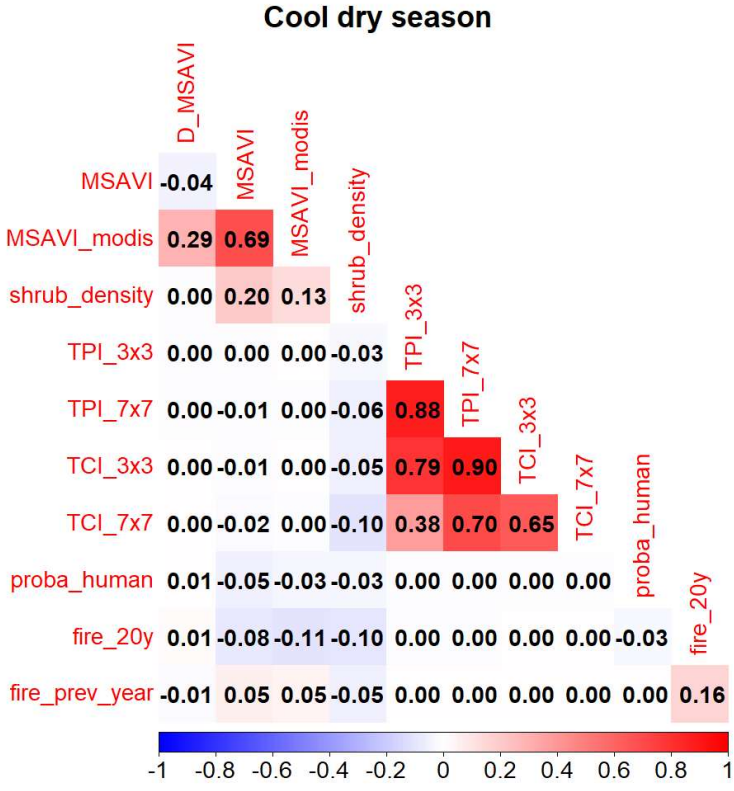

Figure 28: Pair-to-pair correlation matrix for the cool dry season

A.14. Standardization of the environmental variables

```
################################################################
################## SCALE THE COVARIATES ########################
################################################################

library(dplyr)

########### We merge the seasonal covariate datasets into one (adapt the paths)
cov_hot_GEE <- readRDS("C:/Users/court/Documents/cov_hot_GEE.rds")
cov_rainy_GEE <- readRDS("C:/Users/court/Documents/cov_rainy_GEE.rds")
cov_cool_GEE <- readRDS("C:/Users/court/Documents/cov_cool_GEE.rds")

cov_GEE <- rbind(cov_hot_GEE, cov_rainy_GEE, cov_cool_GEE)

############## We standardize (scale) selected continuous covariates
cov_scaled <- cov_GEE %>%
  mutate(
    TPI_3x3_scaled = scale(TPI_3x3),
    TPI_7x7_scaled = scale(TPI_7x7),
    TCI3_scaled = scale(TCI_3x3),
    TCI7_scaled = scale(TCI_7x7),
    proba_human_scaled = scale(proba_human),
    MSAVI_landsat_scaled = scale(MSAVI),
    MSAVI_modis_scaled = scale(MSAVI_modis),
    D_MSAVI_scaled = scale(D_MSAVI),
    shrub_density_scaled = scale(shrub_density),
    fire_20y_scaled = scale(fire_20y)
  )

# We convert burst_ID to numeric for compatibility
cov_scaled$burst_ID <- as.numeric(cov_scaled$burst_ID)

# We replace NA in fire_prev_year with 0 and convert to factor with labels for
it to be correctly interpreted by the ISSF function
cov_scaled$fire_prev_year[is.na(cov_scaled$fire_prev_year)] <- 0
cov_scaled$fire_prev_year <- factor(cov_scaled$fire_prev_year, levels = c(0,
1), labels = c("unburned", "burned"))

# We save the scaled covariate dataset in full first
saveRDS(cov_scaled, "C:/Users/court/Documents/cov_scaled.rds")
write.csv(cov_scaled, "C:/Users/court/Documents/cov_scaled.csv", row.names =
FALSE)

################ Division in seasons
# We divide the dataset in seasons again to calculate the ISSFs seasonally
hot <- cov_scaled %>% filter(season == "hot_dry")
rainy <- cov_scaled %>% filter(season == "rainy")
cool <- cov_scaled %>% filter(season == "cool_dry")

# We save each seasonal subset separately as RDS and CSV files
```

```
saveRDS(hot, "C:/Users/court/Documents/cov_hot_scaled.rds")
saveRDS(rainy, "C:/Users/court/Documents/cov_rainy_scaled.rds")
saveRDS(cool, "C:/Users/court/Documents/cov_cool_scaled.rds")

write.csv(hot, "C:/Users/court/Documents/cov_hot_scaled.csv", row.names =
FALSE)
write.csv(rainy, "C:/Users/court/Documents/cov_rainy_scaled.csv", row.names =
FALSE)
write.csv(cool, "C:/Users/court/Documents/cov_cool_scaled.csv", row.names =
FALSE)
```

A.15. Code for the ISSFs

A.15.1. ISSF models with only linear factors

```
# Download the necessary packages
#install.packages("dplyr")
#install.packages("amt")

# Load libraries
library(amt)
library(dplyr)

##########################################################
##########################################################
################ ISSF linear models ######################
##########################################################
##########################################################


# We load the scaled covariate data split by season
cool <- readRDS("C:/Users/court/Documents/cov_cool_scaled.rds")
rainy <- readRDS("C:/Users/court/Documents/cov_rainy_scaled.rds")
hot <- readRDS("C:/Users/court/Documents/cov_hot_scaled.rds")


# We define the output directory for saving models
output_dir <- "C:/Users/court/Documents/ISSF"

#####################################
############## Rainy ISSF ###########
#####################################


######################### 1st model   ######################

# 1st model of step-selection function with landcover and all the other
environmental factors
issf_tpi3_prev_landsat_rainy <- fit_issf(rainy, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
```

```
                         MSAVI_landsat_scaled + fire_prev_year +
shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 2nd model   ######################

issf_tpi3_prev_modis_rainy <- fit_issf(rainy, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                         MSAVI_modis_scaled + fire_prev_year +
shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 3rd model   ######################

issf_tpi3_prev_dmsavi_rainy <- fit_issf(rainy, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                         D_MSAVI_scaled + fire_prev_year + shrub_density_scaled
+
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 4th model   ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tpi3_20y_landsat_rainy <- fit_issf(rainy, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                         MSAVI_landsat_scaled + fire_20y_scaled +
shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 5th model   ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tpi3_20y_modis_rainy <- fit_issf(rainy, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                         MSAVI_modis_scaled + fire_20y_scaled +
shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 6th model   ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tpi3_20y_dmsavi_rainy <- fit_issf(rainy, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                         D_MSAVI_scaled + fire_20y_scaled + shrub_density_scaled
+
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 7th model   ######################
```

```
# 1st model of step-selection function with landcover and all the other
environmental factors
issf_tci_prev_landsat_rainy <- fit_issf(rainy, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + fire_prev_year +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 8th model   ######################

issf_tci_prev_modis_rainy <- fit_issf(rainy, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + fire_prev_year +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 9th model   ######################

issf_tci_prev_dmsavi_rainy <- fit_issf(rainy, case_ ~ TCI3_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + fire_prev_year + shrub_density_scaled
+
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 10th mode ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tci_20y_landsat_rainy <- fit_issf(rainy, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + fire_20y_scaled +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 11th model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tci_20y_modis_rainy <- fit_issf(rainy, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + fire_20y_scaled +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 12th model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tci_20y_dmsavi_rainy <- fit_issf(rainy, case_ ~ TCI3_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + fire_20y_scaled + shrub_density_scaled
```

```
+
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 13th model ######################

# 1st model of step-selection function with landcover and all the other
environmental factors
issf_tpi7_prev_landsat_rainy <- fit_issf(rainy, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + fire_prev_year +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 14th model ######################

issf_tpi7_prev_modis_rainy <- fit_issf(rainy, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + fire_prev_year +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 15th model ######################

issf_tpi7_prev_dmsavi_rainy <- fit_issf(rainy, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + fire_prev_year + shrub_density_scaled
+
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 16th model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tpi7_20y_landsat_rainy <- fit_issf(rainy, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + fire_20y_scaled +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 17th model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tpi7_20y_modis_rainy <- fit_issf(rainy, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + fire_20y_scaled +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 18th model ######################
```

```
# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tpi7_20y_dmsavi_rainy <- fit_issf(rainy, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                         D_MSAVI_scaled + fire_20y_scaled + shrub_density_scaled
+
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)


# We collect the first 18 rainy season models in a list for batch saving
list_rainy1 <- list(
  issf_tpi3_prev_landsat_rainy = issf_tpi3_prev_landsat_rainy,
  issf_tpi3_prev_modis_rainy = issf_tpi3_prev_modis_rainy,
  issf_tpi3_prev_dmsavi_rainy = issf_tpi3_prev_dmsavi_rainy,

  issf_tpi3_20y_landsat_rainy = issf_tpi3_20y_landsat_rainy,
  issf_tpi3_20y_modis_rainy = issf_tpi3_20y_modis_rainy,
  issf_tpi3_20y_dmsavi_rainy = issf_tpi3_20y_dmsavi_rainy,

  issf_tci_prev_landsat_rainy = issf_tci_prev_landsat_rainy,
  issf_tci_prev_modis_rainy = issf_tci_prev_modis_rainy,
  issf_tci_prev_dmsavi_rainy = issf_tci_prev_dmsavi_rainy,

  issf_tci_20y_landsat_rainy = issf_tci_20y_landsat_rainy,
  issf_tci_20y_modis_rainy = issf_tci_20y_modis_rainy,
  issf_tci_20y_dmsavi_rainy = issf_tci_20y_dmsavi_rainy,

  issf_tpi7_prev_landsat_rainy = issf_tpi7_prev_landsat_rainy,
  issf_tpi7_prev_modis_rainy = issf_tpi7_prev_modis_rainy,
  issf_tpi7_prev_dmsavi_rainy = issf_tpi7_prev_dmsavi_rainy,

  issf_tpi7_20y_landsat_rainy = issf_tpi7_20y_landsat_rainy,
  issf_tpi7_20y_modis_rainy = issf_tpi7_20y_modis_rainy,
  issf_tpi7_20y_dmsavi_rainy = issf_tpi7_20y_dmsavi_rainy
)


# We save each model object to disk as an RDS file named after the model
for (name in names(list_rainy1)) {
  model <- list_rainy1[[name]]
  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}


######################### 19th model ######################

# 1st model of step-selection function with landcover and all the other
environmental factors
issf_tci7_prev_landsat_rainy <- fit_issf(rainy, case_ ~ TCI7_scaled +
proba_human_scaled +
                         MSAVI_landsat_scaled + fire_prev_year +
shrub_density_scaled +
```

```
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 20th model ######################

issf_tci7_prev_modis_rainy <- fit_issf(rainy, case_ ~ TCI7_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + fire_prev_year +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 21st model ######################

issf_tci7_prev_dmsavi_rainy <- fit_issf(rainy, case_ ~ TCI7_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + fire_prev_year + shrub_density_scaled
+
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 22nd model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tci7_20y_landsat_rainy <- fit_issf(rainy, case_ ~ TCI7_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + fire_20y_scaled +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 23rd model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tci7_20y_modis_rainy  <- fit_issf(rainy, case_ ~ TCI7_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + fire_20y_scaled +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 24th model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tci7_20y_dmsavi_rainy <- fit_issf(rainy, case_ ~ TCI7_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + fire_20y_scaled + shrub_density_scaled
+
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)



# We collect the first 18 rainy season models in a list for batch saving
list_rainy2 <- list(
```

```
  issf_tci7_prev_landsat_rainy = issf_tci7_prev_landsat_rainy,
  issf_tci7_prev_modis_rainy = issf_tci7_prev_modis_rainy,
  issf_tci7_prev_dmsavi_rainy = issf_tci7_prev_dmsavi_rainy,
  issf_tci7_20y_landsat_rainy = issf_tci7_20y_landsat_rainy,
  issf_tci7_20y_modis_rainy = issf_tci7_20y_modis_rainy,
  issf_tci7_20y_dmsavi_rainy = issf_tci7_20y_dmsavi_rainy
)


# We save each model object to disk as an RDS file named after the model
for (name in names(list_rainy2)) {
  model <- list_rainy2[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}


#####################################
############## Hot ISSF #############
#####################################

######################### 1st model   ######################

# 1st model of step-selection function with landcover and all the other
environmental factors
issf_tpi3_prev_landsat_hot <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                         MSAVI_landsat_scaled + fire_prev_year +
shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 2nd model   ######################

issf_tpi3_prev_modis_hot <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                         MSAVI_modis_scaled + fire_prev_year +
shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 3rd model   ######################

issf_tpi3_prev_dmsavi_hot <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                         D_MSAVI_scaled + fire_prev_year + shrub_density_scaled
+
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 4th model   ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
```

```
issf_tpi3_20y_landsat_hot <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + fire_20y_scaled +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 5th model   ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tpi3_20y_modis_hot <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + fire_20y_scaled +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 6th model   ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tpi3_20y_dmsavi_hot <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + fire_20y_scaled + shrub_density_scaled
+
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 7th model   ######################

# 1st model of step-selection function with landcover and all the other
environmental factors
issf_tci_prev_landsat_hot <- fit_issf(hot, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + fire_prev_year +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 8th model   ######################

issf_tci_prev_modis_hot <- fit_issf(hot, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + fire_prev_year +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 9th model   ######################

issf_tci_prev_dmsavi_hot <- fit_issf(hot, case_ ~ TCI3_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + fire_prev_year + shrub_density_scaled
+
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)
```

```
######################### 10th mode ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tci_20y_landsat_hot <- fit_issf(hot, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + fire_20y_scaled +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 11th model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tci_20y_modis_hot <- fit_issf(hot, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + fire_20y_scaled +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 12th model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tci_20y_dmsavi_hot <- fit_issf(hot, case_ ~ TCI3_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + fire_20y_scaled + shrub_density_scaled
+
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 13th model ######################

# 1st model of step-selection function with landcover and all the other
environmental factors
issf_tpi7_prev_landsat_hot <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + fire_prev_year +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 14th model ######################

issf_tpi7_prev_modis_hot <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + fire_prev_year +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 15th model ######################

issf_tpi7_prev_dmsavi_hot <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
```

```
proba_human_scaled +
                         D_MSAVI_scaled + fire_prev_year + shrub_density_scaled
+
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 16th model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tpi7_20y_landsat_hot <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                         MSAVI_landsat_scaled + fire_20y_scaled +
shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 17th model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tpi7_20y_modis_hot <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                         MSAVI_modis_scaled + fire_20y_scaled +
shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 18th model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tpi7_20y_dmsavi_hot <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                         D_MSAVI_scaled + fire_20y_scaled + shrub_density_scaled
+
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)


# We collect the first 18 rainy season models in a list for batch saving
list_hot1 <- list(
  issf_tpi3_prev_landsat_hot = issf_tpi3_prev_landsat_hot,
  issf_tpi3_prev_modis_hot = issf_tpi3_prev_modis_hot,
  issf_tpi3_prev_dmsavi_hot = issf_tpi3_prev_dmsavi_hot,

  issf_tpi3_20y_landsat_hot = issf_tpi3_20y_landsat_hot,
  issf_tpi3_20y_modis_hot = issf_tpi3_20y_modis_hot,
  issf_tpi3_20y_dmsavi_hot = issf_tpi3_20y_dmsavi_hot,

  issf_tci_prev_landsat_hot = issf_tci_prev_landsat_hot,
  issf_tci_prev_modis_hot = issf_tci_prev_modis_hot,
  issf_tci_prev_dmsavi_hot = issf_tci_prev_dmsavi_hot,

  issf_tci_20y_landsat_hot = issf_tci_20y_landsat_hot,
  issf_tci_20y_modis_hot = issf_tci_20y_modis_hot,
```

```
  issf_tci_20y_dmsavi_hot = issf_tci_20y_dmsavi_hot,

  issf_tpi7_prev_landsat_hot = issf_tpi7_prev_landsat_hot,
  issf_tpi7_prev_modis_hot = issf_tpi7_prev_modis_hot,
  issf_tpi7_prev_dmsavi_hot = issf_tpi7_prev_dmsavi_hot,

  issf_tpi7_20y_landsat_hot = issf_tpi7_20y_landsat_hot,
  issf_tpi7_20y_modis_hot = issf_tpi7_20y_modis_hot,
  issf_tpi7_20y_dmsavi_hot = issf_tpi7_20y_dmsavi_hot
)


# We save each model object to disk as an RDS file named after the model
for (name in names(list_hot1)) {
  model <- list_hot1[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}


######################### 19th model ######################

# 1st model of step-selection function with landcover and all the other
environmental factors
issf_tci7_prev_landsat_hot <- fit_issf(hot, case_ ~ TCI7_scaled +
proba_human_scaled +
                         MSAVI_landsat_scaled + fire_prev_year +
shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 20th model ######################

issf_tci7_prev_modis_hot <- fit_issf(hot, case_ ~ TCI7_scaled +
proba_human_scaled +
                         MSAVI_modis_scaled + fire_prev_year +
shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 21st model ######################

issf_tci7_prev_dmsavi_hot <- fit_issf(hot, case_ ~ TCI7_scaled +
proba_human_scaled +
                         D_MSAVI_scaled + fire_prev_year + shrub_density_scaled
+
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 22nd model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tci7_20y_landsat_hot <- fit_issf(hot, case_ ~ TCI7_scaled +
```

```
proba_human_scaled +
                         MSAVI_landsat_scaled + fire_20y_scaled +
shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 23rd model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tci7_20y_modis_hot  <- fit_issf(hot, case_ ~ TCI7_scaled +
proba_human_scaled +
                         MSAVI_modis_scaled + fire_20y_scaled +
shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 24th model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tci7_20y_dmsavi_hot <- fit_issf(hot, case_ ~ TCI7_scaled +
proba_human_scaled +
                         D_MSAVI_scaled + fire_20y_scaled + shrub_density_scaled
+
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)



# We collect the first 18 rainy season models in a list for batch saving
list_hot2 <- list(
  issf_tci7_prev_landsat_hot = issf_tci7_prev_landsat_hot,
  issf_tci7_prev_modis_hot = issf_tci7_prev_modis_hot,
  issf_tci7_prev_dmsavi_hot = issf_tci7_prev_dmsavi_hot,
  issf_tci7_20y_landsat_hot = issf_tci7_20y_landsat_hot,
  issf_tci7_20y_modis_hot = issf_tci7_20y_modis_hot,
  issf_tci7_20y_dmsavi_hot = issf_tci7_20y_dmsavi_hot
)




# We save each model object to disk as an RDS file named after the model
for (name in names(list_hot2)) {
  model <- list_hot2[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}


####################################
############## Cool ISSF ############
####################################
```

```
######################### 1st model   ######################

# 1st model of step-selection function with landcover and all the other
environmental factors
issf_tpi3_prev_landsat_cool <- fit_issf(cool, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + fire_prev_year +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 2nd model   ######################

issf_tpi3_prev_modis_cool <- fit_issf(cool, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + fire_prev_year +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 3rd model   ######################

issf_tpi3_prev_dmsavi_cool <- fit_issf(cool, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + fire_prev_year + shrub_density_scaled
+
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 4th model   ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tpi3_20y_landsat_cool <- fit_issf(cool, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + fire_20y_scaled +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 5th model   ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tpi3_20y_modis_cool <- fit_issf(cool, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + fire_20y_scaled +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 6th model   ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
```

```
issf_tpi3_20y_dmsavi_cool <- fit_issf(cool, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + fire_20y_scaled + shrub_density_scaled
+
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 7th model   ######################

# 1st model of step-selection function with landcover and all the other
environmental factors
issf_tci_prev_landsat_cool <- fit_issf(cool, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + fire_prev_year +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 8th model   ######################

issf_tci_prev_modis_cool <- fit_issf(cool, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + fire_prev_year +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 9th model   ######################

issf_tci_prev_dmsavi_cool <- fit_issf(cool, case_ ~ TCI3_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + fire_prev_year + shrub_density_scaled
+
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 10th mode ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tci_20y_landsat_cool <- fit_issf(cool, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + fire_20y_scaled +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 11th model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tci_20y_modis_cool <- fit_issf(cool, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + fire_20y_scaled +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)
```

```
######################### 12th model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tci_20y_dmsavi_cool <- fit_issf(cool, case_ ~ TCI3_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + fire_20y_scaled + shrub_density_scaled
+
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 13th model ######################

# 1st model of step-selection function with landcover and all the other
environmental factors
issf_tpi7_prev_landsat_cool <- fit_issf(cool, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + fire_prev_year +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 14th model ######################

issf_tpi7_prev_modis_cool <- fit_issf(cool, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + fire_prev_year +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 15th model ######################

issf_tpi7_prev_dmsavi_cool <- fit_issf(cool, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + fire_prev_year + shrub_density_scaled
+
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 16th model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tpi7_20y_landsat_cool <- fit_issf(cool, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + fire_20y_scaled +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ + # movement statistics
                        strata(step_id_), model = TRUE)

######################### 17th model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tpi7_20y_modis_cool <- fit_issf(cool, case_ ~ TPI_7x7_scaled +
```

```
proba_human_scaled +
                         MSAVI_modis_scaled + fire_20y_scaled +
shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 18th model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tpi7_20y_dmsavi_cool <- fit_issf(cool, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                         D_MSAVI_scaled + fire_20y_scaled + shrub_density_scaled
+
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)


# We collect the first 18 rainy season models in a list for batch saving
list_cool1 <- list(
  issf_tpi3_prev_landsat_cool = issf_tpi3_prev_landsat_cool,
  issf_tpi3_prev_modis_cool = issf_tpi3_prev_modis_cool,
  issf_tpi3_prev_dmsavi_cool = issf_tpi3_prev_dmsavi_cool,
  issf_tpi3_20y_landsat_cool = issf_tpi3_20y_landsat_cool,
  issf_tpi3_20y_modis_cool = issf_tpi3_20y_modis_cool,
  issf_tpi3_20y_dmsavi_cool = issf_tpi3_20y_dmsavi_cool,
  issf_tci_prev_landsat_cool = issf_tci_prev_landsat_cool,
  issf_tci_prev_modis_cool = issf_tci_prev_modis_cool,
  issf_tci_prev_dmsavi_cool = issf_tci_prev_dmsavi_cool,
  issf_tci_20y_landsat_cool = issf_tci_20y_landsat_cool,
  issf_tci_20y_modis_cool = issf_tci_20y_modis_cool,
  issf_tci_20y_dmsavi_cool = issf_tci_20y_dmsavi_cool,
  issf_tpi7_prev_landsat_cool = issf_tpi7_prev_landsat_cool,
  issf_tpi7_prev_modis_cool = issf_tpi7_prev_modis_cool,
  issf_tpi7_prev_dmsavi_cool = issf_tpi7_prev_dmsavi_cool,
  issf_tpi7_20y_landsat_cool = issf_tpi7_20y_landsat_cool,
  issf_tpi7_20y_modis_cool = issf_tpi7_20y_modis_cool,
  issf_tpi7_20y_dmsavi_cool = issf_tpi7_20y_dmsavi_cool
)


# We save each model object to disk as an RDS file named after the model
for (name in names(list_cool1)) {
  model <- list_cool1[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}

######################### 19th model ######################

# 1st model of step-selection function with landcover and all the other
environmental factors
issf_tci7_prev_landsat_cool <- fit_issf(cool, case_ ~ TCI7_scaled +
```

```
proba_human_scaled +
                         MSAVI_landsat_scaled + fire_prev_year +
shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)


######################### 20th model ######################

issf_tci7_prev_modis_cool <- fit_issf(cool, case_ ~ TCI7_scaled +
proba_human_scaled +
                         MSAVI_modis_scaled + fire_prev_year +
shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 21th model ######################

issf_tci7_prev_dmsavi_cool <- fit_issf(cool, case_ ~ TCI7_scaled +
proba_human_scaled +
                         D_MSAVI_scaled + fire_prev_year + shrub_density_scaled
+
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 22nd model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tci7_20y_landsat_cool <- fit_issf(cool, case_ ~ TCI7_scaled +
proba_human_scaled +
                         MSAVI_landsat_scaled + fire_20y_scaled +
shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 23rd model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tci7_20y_modis_cool <- fit_issf(cool, case_ ~ TCI7_scaled +
proba_human_scaled +
                         MSAVI_modis_scaled + fire_20y_scaled +
shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)

######################### 24th model ######################

# 3rd model of step-selection function excluding every environmental factor
except landcover
issf_tci7_20y_dmsavi_cool <- fit_issf(cool, case_ ~ TCI7_scaled +
proba_human_scaled +
                         D_MSAVI_scaled + fire_20y_scaled + shrub_density_scaled
+
                         sl_ + log_sl_ + cos_ta_ + # movement statistics
                         strata(step_id_), model = TRUE)
```

```
# We collect the first 18 rainy season models in a list for batch saving
list_cool2 <- list(
  issf_tci7_prev_landsat_cool = issf_tci7_prev_landsat_cool,
  issf_tci7_prev_modis_cool = issf_tci7_prev_modis_cool,
  issf_tci7_prev_dmsavi_cool = issf_tci7_prev_dmsavi_cool,
  issf_tci7_20y_landsat_cool = issf_tci7_20y_landsat_cool,
  issf_tci7_20y_modis_cool = issf_tci7_20y_modis_cool,
  issf_tci7_20y_dmsavi_cool = issf_tci7_20y_dmsavi_cool)



# We save each model object to disk as an RDS file named after the model
for (name in names(list_cool2)) {
  model <- list_cool2[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}
```

A.15.2. ISSF models with quadratic topographic factors

```
# Download the necessary packages
#install.packages("dplyr")
#install.packages("amt")


library(amt)
library(dplyr)

#################################################################
#################################################################
################ ISSF quadratic topography ######################
#################################################################
#################################################################


# We load the scaled covariate data split by season
cool <- readRDS("C:/Users/court/Documents/cov_cool_scaled.rds")
rainy <- readRDS("C:/Users/court/Documents/cov_rainy_scaled.rds")
hot <- readRDS("C:/Users/court/Documents/cov_hot_scaled.rds")


# We define the output directory for saving models
output_dir <- "C:/Users/court/Documents/ISSF"

#####################################
############## Rainy ISSF ###########
```

```
#####################################

######################### 1st model ######################

issf_tpi3_prev_landsat_rainy_exp1 <- fit_issf(rainy, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 2nd model ######################

issf_tpi3_prev_modis_rainy_exp1 <- fit_issf(rainy, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 3rd model ######################

issf_tpi3_prev_dmsavi_rainy_exp1 <- fit_issf(rainy, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 4th model ######################

issf_tpi3_20y_landsat_rainy_exp1 <- fit_issf(rainy, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 5th model ######################

issf_tpi3_20y_modis_rainy_exp1 <- fit_issf(rainy, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 6th model ######################

issf_tpi3_20y_dmsavi_rainy_exp1 <- fit_issf(rainy, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)
```

```
######################### 7th model ######################

issf_tci_prev_landsat_rainy_exp1 <- fit_issf(rainy, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 8th model ######################

issf_tci_prev_modis_rainy_exp1 <- fit_issf(rainy, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 9th model ######################

issf_tci_prev_dmsavi_rainy_exp1 <- fit_issf(rainy, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 10th model ######################

issf_tci_20y_landsat_rainy_exp1 <- fit_issf(rainy, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 11th model ######################

issf_tci_20y_modis_rainy_exp1 <- fit_issf(rainy, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 12th model ######################

issf_tci_20y_dmsavi_rainy_exp1 <- fit_issf(rainy, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)
```

```
######################### 13th model ######################

issf_tpi7_prev_landsat_rainy_exp1 <- fit_issf(rainy, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 14th model ######################

issf_tpi7_prev_modis_rainy_exp1 <- fit_issf(rainy, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 15th model ######################

issf_tpi7_prev_dmsavi_rainy_exp1 <- fit_issf(rainy, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 16th model ######################

issf_tpi7_20y_landsat_rainy_exp1 <- fit_issf(rainy, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 17th model ######################

issf_tpi7_20y_modis_rainy_exp1 <- fit_issf(rainy, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 18th model ######################

issf_tpi7_20y_dmsavi_rainy_exp1 <- fit_issf(rainy, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)
```

```
# We collect the first 18 rainy season models in a list for batch saving
list_rainy1 <- list(
  issf_tpi3_prev_landsat_rainy_exp1 = issf_tpi3_prev_landsat_rainy_exp1,
  issf_tpi3_prev_modis_rainy_exp1 = issf_tpi3_prev_modis_rainy_exp1,
  issf_tpi3_prev_dmsavi_rainy_exp1 = issf_tpi3_prev_dmsavi_rainy_exp1,
  issf_tpi3_20y_landsat_rainy_exp1 = issf_tpi3_20y_landsat_rainy_exp1,
  issf_tpi3_20y_modis_rainy_exp1 = issf_tpi3_20y_modis_rainy_exp1,
  issf_tpi3_20y_dmsavi_rainy_exp1 = issf_tpi3_20y_dmsavi_rainy_exp1,
  issf_tci_prev_landsat_rainy_exp1 = issf_tci_prev_landsat_rainy_exp1,
  issf_tci_prev_modis_rainy_exp1 = issf_tci_prev_modis_rainy_exp1,
  issf_tci_prev_dmsavi_rainy_exp1 = issf_tci_prev_dmsavi_rainy_exp1,
  issf_tci_20y_landsat_rainy_exp1 = issf_tci_20y_landsat_rainy_exp1,
  issf_tci_20y_modis_rainy_exp1 = issf_tci_20y_modis_rainy_exp1,
  issf_tci_20y_dmsavi_rainy_exp1 = issf_tci_20y_dmsavi_rainy_exp1,
  issf_tpi7_prev_landsat_rainy_exp1 = issf_tpi7_prev_landsat_rainy_exp1,
  issf_tpi7_prev_modis_rainy_exp1 = issf_tpi7_prev_modis_rainy_exp1,
  issf_tpi7_prev_dmsavi_rainy_exp1 = issf_tpi7_prev_dmsavi_rainy_exp1,
  issf_tpi7_20y_landsat_rainy_exp1 = issf_tpi7_20y_landsat_rainy_exp1,
  issf_tpi7_20y_modis_rainy_exp1 = issf_tpi7_20y_modis_rainy_exp1,
  issf_tpi7_20y_dmsavi_rainy_exp1 = issf_tpi7_20y_dmsavi_rainy_exp1
)



# We save each model object to disk as an RDS file named after the model
for (name in names(list_rainy1)) {
  model <- list_rainy1[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}







######################### 19th model ######################

issf_tci7_prev_landsat_rainy_exp1 <- fit_issf(rainy, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + fire_prev_year +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 20th model ######################

issf_tci7_prev_modis_rainy_exp1 <- fit_issf(rainy, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + fire_prev_year +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
```

```
                        strata(step_id_), model = TRUE)

######################### 21st model ######################

issf_tci7_prev_dmsavi_rainy_exp1 <- fit_issf(rainy, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + fire_prev_year + shrub_density_scaled
+
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 22nd model ######################

issf_tci7_20y_landsat_rainy_exp1 <- fit_issf(rainy, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + fire_20y_scaled +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 23rd model ######################

issf_tci7_20y_modis_rainy_exp1 <- fit_issf(rainy, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + fire_20y_scaled +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 24th model ######################

issf_tci7_20y_dmsavi_rainy_exp1 <- fit_issf(rainy, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + fire_20y_scaled + shrub_density_scaled
+
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)



# We collect the first 18 rainy season models in a list for batch saving
list_rainy2 <- list(
  issf_tci7_prev_landsat_rainy_exp1 = issf_tci7_prev_landsat_rainy_exp1,
  issf_tci7_prev_modis_rainy_exp1 = issf_tci7_prev_modis_rainy_exp1,
  issf_tci7_prev_dmsavi_rainy_exp1 = issf_tci7_prev_dmsavi_rainy_exp1,
  issf_tci7_20y_landsat_rainy_exp1 = issf_tci7_20y_landsat_rainy_exp1,
  issf_tci7_20y_modis_rainy_exp1 = issf_tci7_20y_modis_rainy_exp1,
  issf_tci7_20y_dmsavi_rainy_exp1 = issf_tci7_20y_dmsavi_rainy_exp1
)



# We save each model object to disk as an RDS file named after the model
for (name in names(list_rainy2)) {
  model <- list_rainy2[[name]]
```

```
  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}









#####################################
############## Hot ISSF #############
#####################################

######################### 1st model ######################

issf_tpi3_prev_landsat_hot_exp1 <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 2nd model ######################

issf_tpi3_prev_modis_hot_exp1 <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 3rd model ######################

issf_tpi3_prev_dmsavi_hot_exp1 <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 4th model ######################

issf_tpi3_20y_landsat_hot_exp1 <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 5th model ######################

issf_tpi3_20y_modis_hot_exp1 <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
```

```
                        MSAVI_modis_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 6th model ######################

issf_tpi3_20y_dmsavi_hot_exp1 <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 7th model ######################

issf_tci_prev_landsat_hot_exp1 <- fit_issf(hot, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 8th model ######################

issf_tci_prev_modis_hot_exp1 <- fit_issf(hot, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 9th model ######################

issf_tci_prev_dmsavi_hot_exp1 <- fit_issf(hot, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 10th model ######################

issf_tci_20y_landsat_hot_exp1 <- fit_issf(hot, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 11th model ######################

issf_tci_20y_modis_hot_exp1 <- fit_issf(hot, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled +
```

```
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 12th model ######################

issf_tci_20y_dmsavi_hot_exp1 <- fit_issf(hot, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 13th model ######################

issf_tpi7_prev_landsat_hot_exp1 <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 14th model ######################

issf_tpi7_prev_modis_hot_exp1 <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 15th model ######################

issf_tpi7_prev_dmsavi_hot_exp1 <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 16th model ######################

issf_tpi7_20y_landsat_hot_exp1 <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 17th model ######################

issf_tpi7_20y_modis_hot_exp1 <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled +
                        fire_20y_scaled + shrub_density_scaled +
```

```
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 18th model ######################

issf_tpi7_20y_dmsavi_hot_exp1 <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)



# We collect the first 18 rainy season models in a list for batch saving
list_hot1 <- list(
  issf_tpi3_prev_landsat_hot_exp1 = issf_tpi3_prev_landsat_hot_exp1,
  issf_tpi3_prev_modis_hot_exp1 = issf_tpi3_prev_modis_hot_exp1,
  issf_tpi3_prev_dmsavi_hot_exp1 = issf_tpi3_prev_dmsavi_hot_exp1,
  issf_tpi3_20y_landsat_hot_exp1 = issf_tpi3_20y_landsat_hot_exp1,
  issf_tpi3_20y_modis_hot_exp1 = issf_tpi3_20y_modis_hot_exp1,
  issf_tpi3_20y_dmsavi_hot_exp1 = issf_tpi3_20y_dmsavi_hot_exp1,
  issf_tci_prev_landsat_hot_exp1 = issf_tci_prev_landsat_hot_exp1,
  issf_tci_prev_modis_hot_exp1 = issf_tci_prev_modis_hot_exp1,
  issf_tci_prev_dmsavi_hot_exp1 = issf_tci_prev_dmsavi_hot_exp1,
  issf_tci_20y_landsat_hot_exp1 = issf_tci_20y_landsat_hot_exp1,
  issf_tci_20y_modis_hot_exp1 = issf_tci_20y_modis_hot_exp1,
  issf_tci_20y_dmsavi_hot_exp1 = issf_tci_20y_dmsavi_hot_exp1,
  issf_tpi7_prev_landsat_hot_exp1 = issf_tpi7_prev_landsat_hot_exp1,
  issf_tpi7_prev_modis_hot_exp1 = issf_tpi7_prev_modis_hot_exp1,
  issf_tpi7_prev_dmsavi_hot_exp1 = issf_tpi7_prev_dmsavi_hot_exp1,
  issf_tpi7_20y_landsat_hot_exp1 = issf_tpi7_20y_landsat_hot_exp1,
  issf_tpi7_20y_modis_hot_exp1 = issf_tpi7_20y_modis_hot_exp1,
  issf_tpi7_20y_dmsavi_hot_exp1 = issf_tpi7_20y_dmsavi_hot_exp1
)



# We save each model object to disk as an RDS file named after the model
for (name in names(list_hot1)) {
  model <- list_hot1[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}


######################### 19th model ######################

issf_tci7_prev_landsat_hot_exp1 <- fit_issf(hot, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + fire_prev_year +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
```

```
                        strata(step_id_), model = TRUE)

######################### 20th model ######################

issf_tci7_prev_modis_hot_exp1 <- fit_issf(hot, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + fire_prev_year +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 21st model ######################

issf_tci7_prev_dmsavi_hot_exp1 <- fit_issf(hot, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + fire_prev_year + shrub_density_scaled
+
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 22nd model ######################

issf_tci7_20y_landsat_hot_exp1 <- fit_issf(hot, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + fire_20y_scaled +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 23rd model ######################

issf_tci7_20y_modis_hot_exp1 <- fit_issf(hot, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + fire_20y_scaled +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 24th model ######################

issf_tci7_20y_dmsavi_hot_exp1 <- fit_issf(hot, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + fire_20y_scaled + shrub_density_scaled
+
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)



# We collect the first 18 rainy season models in a list for batch saving
list_hot2 <- list(
  issf_tci7_prev_landsat_hot_exp1 = issf_tci7_prev_landsat_hot_exp1,
  issf_tci7_prev_modis_hot_exp1 = issf_tci7_prev_modis_hot_exp1,
  issf_tci7_prev_dmsavi_hot_exp1 = issf_tci7_prev_dmsavi_hot_exp1,
  issf_tci7_20y_landsat_hot_exp1 = issf_tci7_20y_landsat_hot_exp1,
  issf_tci7_20y_modis_hot_exp1 = issf_tci7_20y_modis_hot_exp1,
```

```
  issf_tci7_20y_dmsavi_hot_exp1 = issf_tci7_20y_dmsavi_hot_exp1
)



# We save each model object to disk as an RDS file named after the model
for (name in names(list_hot2)) {
  model <- list_hot2[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}









#####################################
############## Cool ISSF #############
#####################################


######################### 1st model ######################

issf_tpi3_prev_landsat_cool_exp1 <- fit_issf(cool, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 2nd model ######################

issf_tpi3_prev_modis_cool_exp1 <- fit_issf(cool, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 3rd model ######################

issf_tpi3_prev_dmsavi_cool_exp1 <- fit_issf(cool, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 4th model ######################

issf_tpi3_20y_landsat_cool_exp1 <- fit_issf(cool, case_ ~ TPI_3x3_scaled +
```

```
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 5th model ######################

issf_tpi3_20y_modis_cool_exp1 <- fit_issf(cool, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 6th model ######################

issf_tpi3_20y_dmsavi_cool_exp1 <- fit_issf(cool, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 7th model ######################

issf_tci_prev_landsat_cool_exp1 <- fit_issf(cool, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 8th model ######################

issf_tci_prev_modis_cool_exp1 <- fit_issf(cool, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 9th model ######################

issf_tci_prev_dmsavi_cool_exp1 <- fit_issf(cool, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 10th model ######################

issf_tci_20y_landsat_cool_exp1 <- fit_issf(cool, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
```

```
                        MSAVI_landsat_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 11th model ######################

issf_tci_20y_modis_cool_exp1 <- fit_issf(cool, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 12th model ######################

issf_tci_20y_dmsavi_cool_exp1 <- fit_issf(cool, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 13th model ######################

issf_tpi7_prev_landsat_cool_exp1 <- fit_issf(cool, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 14th model ######################

issf_tpi7_prev_modis_cool_exp1 <- fit_issf(cool, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 15th model ######################

issf_tpi7_prev_dmsavi_cool_exp1 <- fit_issf(cool, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 16th model ######################

issf_tpi7_20y_landsat_cool_exp1 <- fit_issf(cool, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled +
```

```
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 17th model ######################

issf_tpi7_20y_modis_cool_exp1 <- fit_issf(cool, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 18th model ######################

issf_tpi7_20y_dmsavi_cool_exp1 <- fit_issf(cool, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)




# We collect the first 18 rainy season models in a list for batch saving
list_cool1 <- list(
  issf_tpi3_prev_landsat_cool_exp1 = issf_tpi3_prev_landsat_cool_exp1,
  issf_tpi3_prev_modis_cool_exp1 = issf_tpi3_prev_modis_cool_exp1,
  issf_tpi3_prev_dmsavi_cool_exp1 = issf_tpi3_prev_dmsavi_cool_exp1,
  issf_tpi3_20y_landsat_cool_exp1 = issf_tpi3_20y_landsat_cool_exp1,
  issf_tpi3_20y_modis_cool_exp1 = issf_tpi3_20y_modis_cool_exp1,
  issf_tpi3_20y_dmsavi_cool_exp1 = issf_tpi3_20y_dmsavi_cool_exp1,
  issf_tci_prev_landsat_cool_exp1 = issf_tci_prev_landsat_cool_exp1,
  issf_tci_prev_modis_cool_exp1 = issf_tci_prev_modis_cool_exp1,
  issf_tci_prev_dmsavi_cool_exp1 = issf_tci_prev_dmsavi_cool_exp1,
  issf_tci_20y_landsat_cool_exp1 = issf_tci_20y_landsat_cool_exp1,
  issf_tci_20y_modis_cool_exp1 = issf_tci_20y_modis_cool_exp1,
  issf_tci_20y_dmsavi_cool_exp1 = issf_tci_20y_dmsavi_cool_exp1,
  issf_tpi7_prev_landsat_cool_exp1 = issf_tpi7_prev_landsat_cool_exp1,
  issf_tpi7_prev_modis_cool_exp1 = issf_tpi7_prev_modis_cool_exp1,
  issf_tpi7_prev_dmsavi_cool_exp1 = issf_tpi7_prev_dmsavi_cool_exp1,
  issf_tpi7_20y_landsat_cool_exp1 = issf_tpi7_20y_landsat_cool_exp1,
  issf_tpi7_20y_modis_cool_exp1 = issf_tpi7_20y_modis_cool_exp1,
  issf_tpi7_20y_dmsavi_cool_exp1 = issf_tpi7_20y_dmsavi_cool_exp1
)



# We save each model object to disk as an RDS file named after the model
for (name in names(list_cool1)) {
  model <- list_cool1[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
```

```
  saveRDS(model, file = file_path)
}









######################### 19th model ######################

issf_tci7_prev_landsat_cool_exp1 <- fit_issf(cool, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + fire_prev_year +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 20th model ######################

issf_tci7_prev_modis_cool_exp1 <- fit_issf(cool, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + fire_prev_year +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 21st model ######################

issf_tci7_prev_dmsavi_cool_exp1 <- fit_issf(cool, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + fire_prev_year + shrub_density_scaled
+
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 22nd model ######################

issf_tci7_20y_landsat_cool_exp1 <- fit_issf(cool, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + fire_20y_scaled +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 23rd model ######################

issf_tci7_20y_modis_cool_exp1 <- fit_issf(cool, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + fire_20y_scaled +
shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 24st model ######################

issf_tci7_20y_dmsavi_cool_exp1 <- fit_issf(cool, case_ ~ TCI7_scaled +
```

```
I(TCI7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + fire_20y_scaled + shrub_density_scaled
+
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)




# We collect the first 18 rainy season models in a list for batch saving
list_cool2 <- list(
  issf_tci7_prev_landsat_cool_exp1 = issf_tci7_prev_landsat_cool_exp1,
  issf_tci7_prev_modis_cool_exp1 = issf_tci7_prev_modis_cool_exp1,
  issf_tci7_prev_dmsavi_cool_exp1 = issf_tci7_prev_dmsavi_cool_exp1,
  issf_tci7_20y_landsat_cool_exp1 = issf_tci7_20y_landsat_cool_exp1,
  issf_tci7_20y_modis_cool_exp1 = issf_tci7_20y_modis_cool_exp1,
  issf_tci7_20y_dmsavi_cool_exp1 = issf_tci7_20y_dmsavi_cool_exp1
)




# We save each model object to disk as an RDS file named after the model
for (name in names(list_cool2)) {
  model <- list_cool2[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}
```

A.15.3. ISSF models with quadratic vegetation density factors

```
#install.packages("dplyr")
#install.packages("amt")

library(amt)
library(dplyr)


################################################################################
################################################################################
################ ISSF quadratic vegetation density factors ####################
################################################################################
################################################################################

# We load the scaled covariate data split by season
cool <- readRDS("C:/Users/court/Documents/cov_cool_scaled.rds")
rainy <- readRDS("C:/Users/court/Documents/cov_rainy_scaled.rds")
hot <- readRDS("C:/Users/court/Documents/cov_hot_scaled.rds")

# We define the output directory for saving models
output_dir <- "C:/Users/court/Documents/ISSF"
```

```
######################### 1st model ######################

issf_tpi3_prev_landsat_rainy_semi_linear <- fit_issf(rainy, case_ ~
TPI_3x3_scaled + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 2nd model ######################

issf_tpi3_prev_modis_rainy_semi_linear <- fit_issf(rainy, case_ ~
TPI_3x3_scaled + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 3rd model ######################

issf_tpi3_prev_dmsavi_rainy_semi_linear <- fit_issf(rainy, case_ ~
TPI_3x3_scaled + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)



######################### 4th model ######################

issf_tpi3_20y_landsat_rainy_semi_linear <- fit_issf(rainy, case_ ~
TPI_3x3_scaled + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 5th model ######################

issf_tpi3_20y_modis_rainy_semi_linear <- fit_issf(rainy, case_ ~ TPI_3x3_scaled
+ proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 6th model ######################

issf_tpi3_20y_dmsavi_rainy_semi_linear <- fit_issf(rainy, case_ ~
TPI_3x3_scaled + proba_human_scaled +
```

```
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 7th model ######################

issf_tci_prev_landsat_rainy_semi_linear <- fit_issf(rainy, case_ ~ TCI3_scaled
+ proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 8th model ######################

issf_tci_prev_modis_rainy_semi_linear <- fit_issf(rainy, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 9th model ######################

issf_tci_prev_dmsavi_rainy_semi_linear <- fit_issf(rainy, case_ ~ TCI3_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 10th model ######################

issf_tci_20y_landsat_rainy_semi_linear <- fit_issf(rainy, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 11th model ######################

issf_tci_20y_modis_rainy_semi_linear <- fit_issf(rainy, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)
```

```
######################### 12th model ######################

issf_tci_20y_dmsavi_rainy_semi_linear <- fit_issf(rainy, case_ ~ TCI3_scaled +
proba_human_scaled +
                         D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                         fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                         sl_ + log_sl_ + cos_ta_ +
                         strata(step_id_), model = TRUE)

######################### 13th model ######################

issf_tpi7_prev_landsat_rainy_semi_linear <- fit_issf(rainy, case_ ~
TPI_7x7_scaled + proba_human_scaled +
                         MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                         fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                         sl_ + log_sl_ + cos_ta_ +
                         strata(step_id_), model = TRUE)

######################### 14th model ######################

issf_tpi7_prev_modis_rainy_semi_linear <- fit_issf(rainy, case_ ~
TPI_7x7_scaled + proba_human_scaled +
                         MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                         fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                         sl_ + log_sl_ + cos_ta_ +
                         strata(step_id_), model = TRUE)

######################### 15th model ######################

issf_tpi7_prev_dmsavi_rainy_semi_linear <- fit_issf(rainy, case_ ~
TPI_7x7_scaled + proba_human_scaled +
                         D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                         fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                         sl_ + log_sl_ + cos_ta_ +
                         strata(step_id_), model = TRUE)

######################### 16th model ######################

issf_tpi7_20y_landsat_rainy_semi_linear <- fit_issf(rainy, case_ ~
TPI_7x7_scaled + proba_human_scaled +
                         MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                         fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                         sl_ + log_sl_ + cos_ta_ +
                         strata(step_id_), model = TRUE)

######################### 17th model ######################

issf_tpi7_20y_modis_rainy_semi_linear <- fit_issf(rainy, case_ ~ TPI_7x7_scaled
+ proba_human_scaled +
                         MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                         fire_20y_scaled + shrub_density_scaled +
```

```
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 18th model ######################

issf_tpi7_20y_dmsavi_rainy_semi_linear <- fit_issf(rainy, case_ ~
TPI_7x7_scaled + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)


# We collect the first 18 rainy season models in a list for batch saving
list_rainy1 <- list(
  issf_tpi3_prev_landsat_rainy_semi_linear =
issf_tpi3_prev_landsat_rainy_semi_linear,
  issf_tpi3_prev_modis_rainy_semi_linear =
issf_tpi3_prev_modis_rainy_semi_linear,
  issf_tpi3_prev_dmsavi_rainy_semi_linear =
issf_tpi3_prev_dmsavi_rainy_semi_linear,
  issf_tpi3_20y_landsat_rainy_semi_linear =
issf_tpi3_20y_landsat_rainy_semi_linear,
  issf_tpi3_20y_modis_rainy_semi_linear =
issf_tpi3_20y_modis_rainy_semi_linear,
  issf_tpi3_20y_dmsavi_rainy_semi_linear =
issf_tpi3_20y_dmsavi_rainy_semi_linear,
  issf_tci_prev_landsat_rainy_semi_linear =
issf_tci_prev_landsat_rainy_semi_linear,
  issf_tci_prev_modis_rainy_semi_linear =
issf_tci_prev_modis_rainy_semi_linear,
  issf_tci_prev_dmsavi_rainy_semi_linear =
issf_tci_prev_dmsavi_rainy_semi_linear,
  issf_tci_20y_landsat_rainy_semi_linear =
issf_tci_20y_landsat_rainy_semi_linear,
  issf_tci_20y_modis_rainy_semi_linear = issf_tci_20y_modis_rainy_semi_linear,
  issf_tci_20y_dmsavi_rainy_semi_linear =
issf_tci_20y_dmsavi_rainy_semi_linear,
  issf_tpi7_prev_landsat_rainy_semi_linear =
issf_tpi7_prev_landsat_rainy_semi_linear,
  issf_tpi7_prev_modis_rainy_semi_linear =
issf_tpi7_prev_modis_rainy_semi_linear,
  issf_tpi7_prev_dmsavi_rainy_semi_linear =
issf_tpi7_prev_dmsavi_rainy_semi_linear,
  issf_tpi7_20y_landsat_rainy_semi_linear =
issf_tpi7_20y_landsat_rainy_semi_linear,
  issf_tpi7_20y_modis_rainy_semi_linear =
issf_tpi7_20y_modis_rainy_semi_linear,
  issf_tpi7_20y_dmsavi_rainy_semi_linear =
issf_tpi7_20y_dmsavi_rainy_semi_linear
)



# We save each model object to disk as an RDS file named after the model
for (name in names(list_rainy1)) {
```

```
  model <- list_rainy1[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}



######################### 19th model ######################

issf_tci7_prev_landsat_rainy_semi_linear <- fit_issf(rainy, case_ ~ TCI7_scaled
+ proba_human_scaled +
                         MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                         fire_prev_year + shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ +
                         strata(step_id_), model = TRUE)

######################### 20th model ######################

issf_tci7_prev_modis_rainy_semi_linear <- fit_issf(rainy, case_ ~ TCI7_scaled +
proba_human_scaled +
                         MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                         fire_prev_year + shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ +
                         strata(step_id_), model = TRUE)

######################### 21st model ######################

issf_tci7_prev_dmsavi_rainy_semi_linear <- fit_issf(rainy, case_ ~ TCI7_scaled
+ proba_human_scaled +
                         D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                         fire_prev_year + shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ +
                         strata(step_id_), model = TRUE)

######################### 22nd model ######################

issf_tci7_20y_landsat_rainy_semi_linear <- fit_issf(rainy, case_ ~ TCI7_scaled
+ proba_human_scaled +
                         MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                         fire_20y_scaled + shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ +
                         strata(step_id_), model = TRUE)

######################### 23rd model ######################

issf_tci7_20y_modis_rainy_semi_linear <- fit_issf(rainy, case_ ~ TCI7_scaled +
proba_human_scaled +
                         MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                         fire_20y_scaled + shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ +
                         strata(step_id_), model = TRUE)

######################### 24th model ######################
```

```
issf_tci7_20y_dmsavi_rainy_semi_linear <- fit_issf(rainy, case_ ~ TCI7_scaled +
proba_human_scaled +
                         D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                         fire_20y_scaled + shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ +
                         strata(step_id_), model = TRUE)


# We collect the first 18 rainy season models in a list for batch saving
list_rainy2 <- list(
  issf_tci7_prev_landsat_rainy_semi_linear =
issf_tci7_prev_landsat_rainy_semi_linear,
  issf_tci7_prev_modis_rainy_semi_linear =
issf_tci7_prev_modis_rainy_semi_linear,
  issf_tci7_prev_dmsavi_rainy_semi_linear =
issf_tci7_prev_dmsavi_rainy_semi_linear,
  issf_tci7_20y_landsat_rainy_semi_linear =
issf_tci7_20y_landsat_rainy_semi_linear,
  issf_tci7_20y_modis_rainy_semi_linear =
issf_tci7_20y_modis_rainy_semi_linear,
  issf_tci7_20y_dmsavi_rainy_semi_linear =
issf_tci7_20y_dmsavi_rainy_semi_linear
)



# We save each model object to disk as an RDS file named after the model
for (name in names(list_rainy2)) {
  model <- list_rainy2[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}


#####################################
############## Hot ISSF #############
#####################################


######################### 1st model ######################

issf_tpi3_prev_landsat_hot_semi_linear <- fit_issf(hot, case_ ~ TPI_3x3_scaled
+ proba_human_scaled +
                         MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                         fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                         sl_ + log_sl_ + cos_ta_ +
                         strata(step_id_), model = TRUE)

######################### 2nd model ######################

issf_tpi3_prev_modis_hot_semi_linear <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
```

```
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 3rd model ######################

issf_tpi3_prev_dmsavi_hot_semi_linear <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 4th model ######################

issf_tpi3_20y_landsat_hot_semi_linear <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 5th model ######################

issf_tpi3_20y_modis_hot_semi_linear <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 6th model ######################

issf_tpi3_20y_dmsavi_hot_semi_linear <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 7th model ######################

issf_tci_prev_landsat_hot_semi_linear <- fit_issf(hot, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)
```

```
######################### 8th model ######################

issf_tci_prev_modis_hot_semi_linear <- fit_issf(hot, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 9th model ######################

issf_tci_prev_dmsavi_hot_semi_linear <- fit_issf(hot, case_ ~ TCI3_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 10th model ######################

issf_tci_20y_landsat_hot_semi_linear <- fit_issf(hot, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 11th model ######################

issf_tci_20y_modis_hot_semi_linear <- fit_issf(hot, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 12th model ######################

issf_tci_20y_dmsavi_hot_semi_linear <- fit_issf(hot, case_ ~ TCI3_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 13th model ######################

issf_tpi7_prev_landsat_hot_semi_linear <- fit_issf(hot, case_ ~ TPI_7x7_scaled
+ proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
```

```
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 14th model ######################

issf_tpi7_prev_modis_hot_semi_linear <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 15th model ######################

issf_tpi7_prev_dmsavi_hot_semi_linear <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 16th model ######################

issf_tpi7_20y_landsat_hot_semi_linear <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 17th model ######################

issf_tpi7_20y_modis_hot_semi_linear <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 18th model ######################

issf_tpi7_20y_dmsavi_hot_semi_linear <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)


# We collect the first 18 rainy season models in a list for batch saving
```

```
list_hot1 <- list(
  issf_tpi3_prev_landsat_hot_semi_linear =
issf_tpi3_prev_landsat_hot_semi_linear,
  issf_tpi3_prev_modis_hot_semi_linear = issf_tpi3_prev_modis_hot_semi_linear,
  issf_tpi3_prev_dmsavi_hot_semi_linear =
issf_tpi3_prev_dmsavi_hot_semi_linear,
  issf_tpi3_20y_landsat_hot_semi_linear =
issf_tpi3_20y_landsat_hot_semi_linear,
  issf_tpi3_20y_modis_hot_semi_linear = issf_tpi3_20y_modis_hot_semi_linear,
  issf_tpi3_20y_dmsavi_hot_semi_linear = issf_tpi3_20y_dmsavi_hot_semi_linear,
  issf_tci_prev_landsat_hot_semi_linear =
issf_tci_prev_landsat_hot_semi_linear,
  issf_tci_prev_modis_hot_semi_linear = issf_tci_prev_modis_hot_semi_linear,
  issf_tci_prev_dmsavi_hot_semi_linear = issf_tci_prev_dmsavi_hot_semi_linear,
  issf_tci_20y_landsat_hot_semi_linear = issf_tci_20y_landsat_hot_semi_linear,
  issf_tci_20y_modis_hot_semi_linear = issf_tci_20y_modis_hot_semi_linear,
  issf_tci_20y_dmsavi_hot_semi_linear = issf_tci_20y_dmsavi_hot_semi_linear,
  issf_tpi7_prev_landsat_hot_semi_linear =
issf_tpi7_prev_landsat_hot_semi_linear,
  issf_tpi7_prev_modis_hot_semi_linear = issf_tpi7_prev_modis_hot_semi_linear,
  issf_tpi7_prev_dmsavi_hot_semi_linear =
issf_tpi7_prev_dmsavi_hot_semi_linear,
  issf_tpi7_20y_landsat_hot_semi_linear =
issf_tpi7_20y_landsat_hot_semi_linear,
  issf_tpi7_20y_modis_hot_semi_linear = issf_tpi7_20y_modis_hot_semi_linear,
  issf_tpi7_20y_dmsavi_hot_semi_linear = issf_tpi7_20y_dmsavi_hot_semi_linear
)



# We save each model object to disk as an RDS file named after the model
for (name in names(list_hot1)) {
  model <- list_hot1[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}






######################### 19th model ######################

issf_tci7_prev_landsat_hot_semi_linear <- fit_issf(hot, case_ ~ TCI7_scaled +
proba_human_scaled +
                         MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                         fire_prev_year + shrub_density_scaled +
                         sl_ + log_sl_ + cos_ta_ +
                         strata(step_id_), model = TRUE)

######################### 20th model ######################

issf_tci7_prev_modis_hot_semi_linear <- fit_issf(hot, case_ ~ TCI7_scaled +
proba_human_scaled +
```

```
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 21st model ######################

issf_tci7_prev_dmsavi_hot_semi_linear <- fit_issf(hot, case_ ~ TCI7_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 22nd model ######################

issf_tci7_20y_landsat_hot_semi_linear <- fit_issf(hot, case_ ~ TCI7_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 23rd model ######################

issf_tci7_20y_modis_hot_semi_linear <- fit_issf(hot, case_ ~ TCI7_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 24th model ######################

issf_tci7_20y_dmsavi_hot_semi_linear <- fit_issf(hot, case_ ~ TCI7_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)



# We collect the first 18 rainy season models in a list for batch saving
list_hot2 <- list(
  issf_tci7_prev_landsat_hot_semi_linear =
issf_tci7_prev_landsat_hot_semi_linear,
  issf_tci7_prev_modis_hot_semi_linear = issf_tci7_prev_modis_hot_semi_linear,
  issf_tci7_prev_dmsavi_hot_semi_linear =
issf_tci7_prev_dmsavi_hot_semi_linear,
  issf_tci7_20y_landsat_hot_semi_linear =
issf_tci7_20y_landsat_hot_semi_linear,
  issf_tci7_20y_modis_hot_semi_linear = issf_tci7_20y_modis_hot_semi_linear,
  issf_tci7_20y_dmsavi_hot_semi_linear = issf_tci7_20y_dmsavi_hot_semi_linear
)
```

```
# We save each model object to disk as an RDS file named after the model
for (name in names(list_hot2)) {
  model <- list_hot2[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}




#####################################
############## Cool ISSF #############
#####################################


######################### 1st model ######################

issf_tpi3_prev_landsat_cool_semi_linear <- fit_issf(cool, case_ ~
TPI_3x3_scaled + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 2nd model ######################

issf_tpi3_prev_modis_cool_semi_linear <- fit_issf(cool, case_ ~ TPI_3x3_scaled
+ proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 3rd model ######################

issf_tpi3_prev_dmsavi_cool_semi_linear <- fit_issf(cool, case_ ~ TPI_3x3_scaled
+ proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 4th model ######################

issf_tpi3_20y_landsat_cool_semi_linear <- fit_issf(cool, case_ ~ TPI_3x3_scaled
+ proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
```

```
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 5th model ######################

issf_tpi3_20y_modis_cool_semi_linear <- fit_issf(cool, case_ ~ TPI_3x3_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 6th model ######################

issf_tpi3_20y_dmsavi_cool_semi_linear <- fit_issf(cool, case_ ~ TPI_3x3_scaled
+ proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 7th model ######################

issf_tci_prev_landsat_cool_semi_linear <- fit_issf(cool, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 8th model ######################

issf_tci_prev_modis_cool_semi_linear <- fit_issf(cool, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 9th model ######################

issf_tci_prev_dmsavi_cool_semi_linear <- fit_issf(cool, case_ ~ TCI3_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 10th model ######################
```

```
issf_tci_20y_landsat_cool_semi_linear <- fit_issf(cool, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 11th model ######################

issf_tci_20y_modis_cool_semi_linear <- fit_issf(cool, case_ ~ TCI3_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 12th model ######################

issf_tci_20y_dmsavi_cool_semi_linear <- fit_issf(cool, case_ ~ TCI3_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 13th model ######################

issf_tpi7_prev_landsat_cool_semi_linear <- fit_issf(cool, case_ ~
TPI_7x7_scaled + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 14th model ######################

issf_tpi7_prev_modis_cool_semi_linear <- fit_issf(cool, case_ ~ TPI_7x7_scaled
+ proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 15th model ######################

issf_tpi7_prev_dmsavi_cool_semi_linear <- fit_issf(cool, case_ ~ TPI_7x7_scaled
+ proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
```

```
                        strata(step_id_), model = TRUE)

######################### 16th model ######################

issf_tpi7_20y_landsat_cool_semi_linear <- fit_issf(cool, case_ ~ TPI_7x7_scaled
+ proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 17th model ######################

issf_tpi7_20y_modis_cool_semi_linear <- fit_issf(cool, case_ ~ TPI_7x7_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 18th model ######################

issf_tpi7_20y_dmsavi_cool_semi_linear <- fit_issf(cool, case_ ~ TPI_7x7_scaled
+ proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)



# We collect the first 18 rainy season models in a list for batch saving
list_cool1 <- list(
  issf_tpi3_prev_landsat_cool_semi_linear =
issf_tpi3_prev_landsat_cool_semi_linear,
  issf_tpi3_prev_modis_cool_semi_linear =
issf_tpi3_prev_modis_cool_semi_linear,
  issf_tpi3_prev_dmsavi_cool_semi_linear =
issf_tpi3_prev_dmsavi_cool_semi_linear,
  issf_tpi3_20y_landsat_cool_semi_linear =
issf_tpi3_20y_landsat_cool_semi_linear,
  issf_tpi3_20y_modis_cool_semi_linear = issf_tpi3_20y_modis_cool_semi_linear,
  issf_tpi3_20y_dmsavi_cool_semi_linear =
issf_tpi3_20y_dmsavi_cool_semi_linear,
  issf_tci_prev_landsat_cool_semi_linear =
issf_tci_prev_landsat_cool_semi_linear,
  issf_tci_prev_modis_cool_semi_linear = issf_tci_prev_modis_cool_semi_linear,
  issf_tci_prev_dmsavi_cool_semi_linear =
issf_tci_prev_dmsavi_cool_semi_linear,
  issf_tci_20y_landsat_cool_semi_linear =
issf_tci_20y_landsat_cool_semi_linear,
  issf_tci_20y_modis_cool_semi_linear = issf_tci_20y_modis_cool_semi_linear,
  issf_tci_20y_dmsavi_cool_semi_linear = issf_tci_20y_dmsavi_cool_semi_linear,
  issf_tpi7_prev_landsat_cool_semi_linear =
issf_tpi7_prev_landsat_cool_semi_linear,
```

```
  issf_tpi7_prev_modis_cool_semi_linear =
issf_tpi7_prev_modis_cool_semi_linear,
  issf_tpi7_prev_dmsavi_cool_semi_linear =
issf_tpi7_prev_dmsavi_cool_semi_linear,
  issf_tpi7_20y_landsat_cool_semi_linear =
issf_tpi7_20y_landsat_cool_semi_linear,
  issf_tpi7_20y_modis_cool_semi_linear = issf_tpi7_20y_modis_cool_semi_linear,
  issf_tpi7_20y_dmsavi_cool_semi_linear = issf_tpi7_20y_dmsavi_cool_semi_linear
)


# We save each model object to disk as an RDS file named after the model
for (name in names(list_cool1)) {
  model <- list_cool1[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}







######################### 19th model ######################

issf_tci7_prev_landsat_cool_semi_linear <- fit_issf(cool, case_ ~ TCI7_scaled +
proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 20th model ######################

issf_tci7_prev_modis_cool_semi_linear <- fit_issf(cool, case_ ~ TCI7_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 21st model ######################

issf_tci7_prev_dmsavi_cool_semi_linear <- fit_issf(cool, case_ ~ TCI7_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 22nd model ######################

issf_tci7_20y_landsat_cool_semi_linear <- fit_issf(cool, case_ ~ TCI7_scaled +
proba_human_scaled +
```

```
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 23rd model ######################

issf_tci7_20y_modis_cool_semi_linear <- fit_issf(cool, case_ ~ TCI7_scaled +
proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 24th model ######################

issf_tci7_20y_dmsavi_cool_semi_linear <- fit_issf(cool, case_ ~ TCI7_scaled +
proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)



# We collect the first 18 rainy season models in a list for batch saving
list_cool2 <- list(
  issf_tci7_prev_landsat_cool_semi_linear =
issf_tci7_prev_landsat_cool_semi_linear,
  issf_tci7_prev_modis_cool_semi_linear =
issf_tci7_prev_modis_cool_semi_linear,
  issf_tci7_prev_dmsavi_cool_semi_linear =
issf_tci7_prev_dmsavi_cool_semi_linear,
  issf_tci7_20y_landsat_cool_semi_linear =
issf_tci7_20y_landsat_cool_semi_linear,
  issf_tci7_20y_modis_cool_semi_linear = issf_tci7_20y_modis_cool_semi_linear,
  issf_tci7_20y_dmsavi_cool_semi_linear = issf_tci7_20y_dmsavi_cool_semi_linear
)



# We save each model object to disk as an RDS file named after the model
for (name in names(list_cool2)) {
  model <- list_cool2[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}
```

A.15.4. ISSF models with topographic and vegetation factors both quadratic

```
# Download the necessary packages
#install.packages("dplyr")
#install.packages("amt")

library(amt)
library(dplyr)

################################################################################
################################################################################
########## ISSF quadratic vegetation density and topographic factors ##########
################################################################################
################################################################################

# We load the scaled covariate data split by season
cool <- readRDS("C:/Users/court/Documents/cov_cool_scaled.rds")
rainy <- readRDS("C:/Users/court/Documents/cov_rainy_scaled.rds")
hot <- readRDS("C:/Users/court/Documents/cov_hot_scaled.rds")

# We define the output directory for saving models
output_dir <- "C:/Users/court/Documents/ISSF"

#####################################
############## Rainy ISSF ###########
#####################################

######################### 1st model ######################

issf_tpi3_prev_landsat_rainy_exp2 <- fit_issf(rainy, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 2nd model ######################

issf_tpi3_prev_modis_rainy_exp2 <- fit_issf(rainy, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 3rd model ######################

issf_tpi3_prev_dmsavi_rainy_exp2 <- fit_issf(rainy, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)
```

```
######################### 4th model ######################

issf_tpi3_20y_landsat_rainy_exp2 <- fit_issf(rainy, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 5th model ######################

issf_tpi3_20y_modis_rainy_exp2 <- fit_issf(rainy, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 6th model ######################

issf_tpi3_20y_dmsavi_rainy_exp2 <- fit_issf(rainy, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 7th model ######################

issf_tci_prev_landsat_rainy_exp2 <- fit_issf(rainy, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 8th model ######################

issf_tci_prev_modis_rainy_exp2 <- fit_issf(rainy, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 9th model ######################

issf_tci_prev_dmsavi_rainy_exp2 <- fit_issf(rainy, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
```

```
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 10th model ######################

issf_tci_20y_landsat_rainy_exp2 <- fit_issf(rainy, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 11th model ######################

issf_tci_20y_modis_rainy_exp2 <- fit_issf(rainy, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 12th model ######################

issf_tci_20y_dmsavi_rainy_exp2 <- fit_issf(rainy, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 13th model ######################

issf_tpi7_prev_landsat_rainy_exp2 <- fit_issf(rainy, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 14th model ######################

issf_tpi7_prev_modis_rainy_exp2 <- fit_issf(rainy, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 15th model ######################
```

```
issf_tpi7_prev_dmsavi_rainy_exp2 <- fit_issf(rainy, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 16th model ######################

issf_tpi7_20y_landsat_rainy_exp2 <- fit_issf(rainy, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 17th model ######################

issf_tpi7_20y_modis_rainy_exp2 <- fit_issf(rainy, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 18th model ######################

issf_tpi7_20y_dmsavi_rainy_exp2 <- fit_issf(rainy, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)



# We collect the first 18 rainy season models in a list for batch saving
list_rainy1 <- list(
  issf_tpi3_prev_landsat_rainy_exp2 = issf_tpi3_prev_landsat_rainy_exp2,
  issf_tpi3_prev_modis_rainy_exp2 = issf_tpi3_prev_modis_rainy_exp2,
  issf_tpi3_prev_dmsavi_rainy_exp2 = issf_tpi3_prev_dmsavi_rainy_exp2,
  issf_tpi3_20y_landsat_rainy_exp2 = issf_tpi3_20y_landsat_rainy_exp2,
  issf_tpi3_20y_modis_rainy_exp2 = issf_tpi3_20y_modis_rainy_exp2,
  issf_tpi3_20y_dmsavi_rainy_exp2 = issf_tpi3_20y_dmsavi_rainy_exp2,
  issf_tci_prev_landsat_rainy_exp2 = issf_tci_prev_landsat_rainy_exp2,
  issf_tci_prev_modis_rainy_exp2 = issf_tci_prev_modis_rainy_exp2,
  issf_tci_prev_dmsavi_rainy_exp2 = issf_tci_prev_dmsavi_rainy_exp2,
  issf_tci_20y_landsat_rainy_exp2 = issf_tci_20y_landsat_rainy_exp2,
  issf_tci_20y_modis_rainy_exp2 = issf_tci_20y_modis_rainy_exp2,
  issf_tci_20y_dmsavi_rainy_exp2 = issf_tci_20y_dmsavi_rainy_exp2,
  issf_tpi7_prev_landsat_rainy_exp2 = issf_tpi7_prev_landsat_rainy_exp2,
  issf_tpi7_prev_modis_rainy_exp2 = issf_tpi7_prev_modis_rainy_exp2,
```

```
  issf_tpi7_prev_dmsavi_rainy_exp2 = issf_tpi7_prev_dmsavi_rainy_exp2,
  issf_tpi7_20y_landsat_rainy_exp2 = issf_tpi7_20y_landsat_rainy_exp2,
  issf_tpi7_20y_modis_rainy_exp2 = issf_tpi7_20y_modis_rainy_exp2,
  issf_tpi7_20y_dmsavi_rainy_exp2 = issf_tpi7_20y_dmsavi_rainy_exp2
)


# We save each model object to disk as an RDS file named after the model
for (name in names(list_rainy1)) {
  model <- list_rainy1[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}


######################### 19th model ######################

issf_tci7_prev_landsat_rainy_exp2 <- fit_issf(rainy, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
fire_prev_year + shrub_density_scaled + I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 20th model ######################

issf_tci7_prev_modis_rainy_exp2 <- fit_issf(rainy, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
fire_prev_year + shrub_density_scaled + I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 21st model ######################

issf_tci7_prev_dmsavi_rainy_exp2 <- fit_issf(rainy, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) + fire_prev_year +
shrub_density_scaled + I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 22nd model ######################

issf_tci7_20y_landsat_rainy_exp2 <- fit_issf(rainy, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
fire_20y_scaled + shrub_density_scaled + I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 23rd model ######################

issf_tci7_20y_modis_rainy_exp2 <- fit_issf(rainy, case_ ~ TCI7_scaled +
```

```
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
fire_20y_scaled + shrub_density_scaled + I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 24th model ######################

issf_tci7_20y_dmsavi_rainy_exp2 <- fit_issf(rainy, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) + fire_20y_scaled
+ shrub_density_scaled + I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)


# We collect the first 18 rainy season models in a list for batch saving
list_rainy2 <- list(
  issf_tci7_prev_landsat_rainy_exp2 = issf_tci7_prev_landsat_rainy_exp2,
  issf_tci7_prev_modis_rainy_exp2 = issf_tci7_prev_modis_rainy_exp2,
  issf_tci7_prev_dmsavi_rainy_exp2 = issf_tci7_prev_dmsavi_rainy_exp2,
  issf_tci7_20y_landsat_rainy_exp2 = issf_tci7_20y_landsat_rainy_exp2,
  issf_tci7_20y_modis_rainy_exp2 = issf_tci7_20y_modis_rainy_exp2,
  issf_tci7_20y_dmsavi_rainy_exp2 = issf_tci7_20y_dmsavi_rainy_exp2
)



# We save each model object to disk as an RDS file named after the model
for (name in names(list_rainy2)) {
  model <- list_rainy2[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}


####################################
############## Hot ISSF #############
####################################

######################### 1st model ######################

issf_tpi3_prev_landsat_hot_exp2 <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 2nd model ######################

issf_tpi3_prev_modis_hot_exp2 <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
```

```
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 3rd model ######################

issf_tpi3_prev_dmsavi_hot_exp2 <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 4th model ######################

issf_tpi3_20y_landsat_hot_exp2 <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 5th model ######################

issf_tpi3_20y_modis_hot_exp2 <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 6th model ######################

issf_tpi3_20y_dmsavi_hot_exp2 <- fit_issf(hot, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 7th model ######################

issf_tci_prev_landsat_hot_exp2 <- fit_issf(hot, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)
```

```
######################### 8th model ######################

issf_tci_prev_modis_hot_exp2 <- fit_issf(hot, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 9th model ######################

issf_tci_prev_dmsavi_hot_exp2 <- fit_issf(hot, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 10th model ######################

issf_tci_20y_landsat_hot_exp2 <- fit_issf(hot, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 11th model ######################

issf_tci_20y_modis_hot_exp2 <- fit_issf(hot, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 12th model ######################

issf_tci_20y_dmsavi_hot_exp2 <- fit_issf(hot, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 13th model ######################

issf_tpi7_prev_landsat_hot_exp2 <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
```

```
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 14th model ######################

issf_tpi7_prev_modis_hot_exp2 <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 15th model ######################

issf_tpi7_prev_dmsavi_hot_exp2 <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 16th model ######################

issf_tpi7_20y_landsat_hot_exp2 <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 17th model ######################

issf_tpi7_20y_modis_hot_exp2 <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 18th model ######################

issf_tpi7_20y_dmsavi_hot_exp2 <- fit_issf(hot, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)
```

```
# We collect the first 18 rainy season models in a list for batch saving
list_hot1 <- list(
  issf_tpi3_prev_landsat_hot_exp2 = issf_tpi3_prev_landsat_hot_exp2,
  issf_tpi3_prev_modis_hot_exp2 = issf_tpi3_prev_modis_hot_exp2,
  issf_tpi3_prev_dmsavi_hot_exp2 = issf_tpi3_prev_dmsavi_hot_exp2,
  issf_tpi3_20y_landsat_hot_exp2 = issf_tpi3_20y_landsat_hot_exp2,
  issf_tpi3_20y_modis_hot_exp2 = issf_tpi3_20y_modis_hot_exp2,
  issf_tpi3_20y_dmsavi_hot_exp2 = issf_tpi3_20y_dmsavi_hot_exp2,
  issf_tci_prev_landsat_hot_exp2 = issf_tci_prev_landsat_hot_exp2,
  issf_tci_prev_modis_hot_exp2 = issf_tci_prev_modis_hot_exp2,
  issf_tci_prev_dmsavi_hot_exp2 = issf_tci_prev_dmsavi_hot_exp2,
  issf_tci_20y_landsat_hot_exp2 = issf_tci_20y_landsat_hot_exp2,
  issf_tci_20y_modis_hot_exp2 = issf_tci_20y_modis_hot_exp2,
  issf_tci_20y_dmsavi_hot_exp2 = issf_tci_20y_dmsavi_hot_exp2,
  issf_tpi7_prev_landsat_hot_exp2 = issf_tpi7_prev_landsat_hot_exp2,
  issf_tpi7_prev_modis_hot_exp2 = issf_tpi7_prev_modis_hot_exp2,
  issf_tpi7_prev_dmsavi_hot_exp2 = issf_tpi7_prev_dmsavi_hot_exp2,
  issf_tpi7_20y_landsat_hot_exp2 = issf_tpi7_20y_landsat_hot_exp2,
  issf_tpi7_20y_modis_hot_exp2 = issf_tpi7_20y_modis_hot_exp2,
  issf_tpi7_20y_dmsavi_hot_exp2 = issf_tpi7_20y_dmsavi_hot_exp2
)


# We save each model object to disk as an RDS file named after the model
for (name in names(list_hot1)) {
  model <- list_hot1[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}



######################### 19th model ######################

issf_tci7_prev_landsat_hot_exp2 <- fit_issf(hot, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
fire_prev_year + shrub_density_scaled + I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 20th model ######################

issf_tci7_prev_modis_hot_exp2 <- fit_issf(hot, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
fire_prev_year + shrub_density_scaled + I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 21st model ######################
```

```
issf_tci7_prev_dmsavi_hot_exp2 <- fit_issf(hot, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) + fire_prev_year +
shrub_density_scaled + I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 22nd model ######################

issf_tci7_20y_landsat_hot_exp2 <- fit_issf(hot, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
fire_20y_scaled + shrub_density_scaled + I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 23rd model ######################

issf_tci7_20y_modis_hot_exp2 <- fit_issf(hot, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
fire_20y_scaled + shrub_density_scaled + I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 24th model ######################

issf_tci7_20y_dmsavi_hot_exp2 <- fit_issf(hot, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) + fire_20y_scaled
+ shrub_density_scaled + I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)


# We collect the first 18 rainy season models in a list for batch saving
list_hot2 <- list(
  issf_tci7_prev_landsat_hot_exp2 = issf_tci7_prev_landsat_hot_exp2,
  issf_tci7_prev_modis_hot_exp2 = issf_tci7_prev_modis_hot_exp2,
  issf_tci7_prev_dmsavi_hot_exp2 = issf_tci7_prev_dmsavi_hot_exp2,
  issf_tci7_20y_landsat_hot_exp2 = issf_tci7_20y_landsat_hot_exp2,
  issf_tci7_20y_modis_hot_exp2 = issf_tci7_20y_modis_hot_exp2,
  issf_tci7_20y_dmsavi_hot_exp2 = issf_tci7_20y_dmsavi_hot_exp2
)


# We save each model object to disk as an RDS file named after the model
for (name in names(list_hot2)) {
  model <- list_hot2[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}
```

```
#####################################
############## Cool ISSF ############
#####################################

######################### 1st model ######################

issf_tpi3_prev_landsat_cool_exp2 <- fit_issf(cool, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 2nd model ######################

issf_tpi3_prev_modis_cool_exp2 <- fit_issf(cool, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 3rd model ######################

issf_tpi3_prev_dmsavi_cool_exp2 <- fit_issf(cool, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 4th model ######################

issf_tpi3_20y_landsat_cool_exp2 <- fit_issf(cool, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 5th model ######################

issf_tpi3_20y_modis_cool_exp2 <- fit_issf(cool, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 6th model ######################
```

```
issf_tpi3_20y_dmsavi_cool_exp2 <- fit_issf(cool, case_ ~ TPI_3x3_scaled +
I(TPI_3x3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 7th model ######################

issf_tci_prev_landsat_cool_exp2 <- fit_issf(cool, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 8th model ######################

issf_tci_prev_modis_cool_exp2 <- fit_issf(cool, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 9th model ######################

issf_tci_prev_dmsavi_cool_exp2 <- fit_issf(cool, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 10th model ######################

issf_tci_20y_landsat_cool_exp2 <- fit_issf(cool, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 11th model ######################

issf_tci_20y_modis_cool_exp2 <- fit_issf(cool, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
```

```
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 12th model ######################

issf_tci_20y_dmsavi_cool_exp2 <- fit_issf(cool, case_ ~ TCI3_scaled +
I(TCI3_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 13th model ######################

issf_tpi7_prev_landsat_cool_exp2 <- fit_issf(cool, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 14th model ######################

issf_tpi7_prev_modis_cool_exp2 <- fit_issf(cool, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 15th model ######################

issf_tpi7_prev_dmsavi_cool_exp2 <- fit_issf(cool, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_prev_year + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 16th model ######################

issf_tpi7_20y_landsat_cool_exp2 <- fit_issf(cool, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 17th model ######################

issf_tpi7_20y_modis_cool_exp2 <- fit_issf(cool, case_ ~ TPI_7x7_scaled +
```

```
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 18th model ######################

issf_tpi7_20y_dmsavi_cool_exp2 <- fit_issf(cool, case_ ~ TPI_7x7_scaled +
I(TPI_7x7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) +
                        fire_20y_scaled + shrub_density_scaled +
I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)



# We collect the first 18 rainy season models in a list for batch saving
list_cool1 <- list(
  issf_tpi3_prev_landsat_cool_exp2 = issf_tpi3_prev_landsat_cool_exp2,
  issf_tpi3_prev_modis_cool_exp2 = issf_tpi3_prev_modis_cool_exp2,
  issf_tpi3_prev_dmsavi_cool_exp2 = issf_tpi3_prev_dmsavi_cool_exp2,
  issf_tpi3_20y_landsat_cool_exp2 = issf_tpi3_20y_landsat_cool_exp2,
  issf_tpi3_20y_modis_cool_exp2 = issf_tpi3_20y_modis_cool_exp2,
  issf_tpi3_20y_dmsavi_cool_exp2 = issf_tpi3_20y_dmsavi_cool_exp2,
  issf_tci_prev_landsat_cool_exp2 = issf_tci_prev_landsat_cool_exp2,
  issf_tci_prev_modis_cool_exp2 = issf_tci_prev_modis_cool_exp2,
  issf_tci_prev_dmsavi_cool_exp2 = issf_tci_prev_dmsavi_cool_exp2,
  issf_tci_20y_landsat_cool_exp2 = issf_tci_20y_landsat_cool_exp2,
  issf_tci_20y_modis_cool_exp2 = issf_tci_20y_modis_cool_exp2,
  issf_tci_20y_dmsavi_cool_exp2 = issf_tci_20y_dmsavi_cool_exp2,
  issf_tpi7_prev_landsat_cool_exp2 = issf_tpi7_prev_landsat_cool_exp2,
  issf_tpi7_prev_modis_cool_exp2 = issf_tpi7_prev_modis_cool_exp2,
  issf_tpi7_prev_dmsavi_cool_exp2 = issf_tpi7_prev_dmsavi_cool_exp2,
  issf_tpi7_20y_landsat_cool_exp2 = issf_tpi7_20y_landsat_cool_exp2,
  issf_tpi7_20y_modis_cool_exp2 = issf_tpi7_20y_modis_cool_exp2,
  issf_tpi7_20y_dmsavi_cool_exp2 = issf_tpi7_20y_dmsavi_cool_exp2
)


# We save each model object to disk as an RDS file named after the model
for (name in names(list_cool1)) {
  model <- list_cool1[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}


######################### 37th model ######################

issf_tci7_prev_landsat_cool_exp2 <- fit_issf(cool, case_ ~ TCI7_scaled +
```

```
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
fire_prev_year + shrub_density_scaled + I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 38th model ######################

issf_tci7_prev_modis_cool_exp2 <- fit_issf(cool, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
fire_prev_year + shrub_density_scaled + I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 39th model ######################

issf_tci7_prev_dmsavi_cool_exp2 <- fit_issf(cool, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) + fire_prev_year +
shrub_density_scaled + I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 40th model ######################

issf_tci7_20y_landsat_cool_exp2 <- fit_issf(cool, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_landsat_scaled + I(MSAVI_landsat_scaled^2) +
fire_20y_scaled + shrub_density_scaled + I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 41st model ######################

issf_tci7_20y_modis_cool_exp2 <- fit_issf(cool, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        MSAVI_modis_scaled + I(MSAVI_modis_scaled^2) +
fire_20y_scaled + shrub_density_scaled + I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)

######################### 42nd model ######################

issf_tci7_20y_dmsavi_cool_exp2 <- fit_issf(cool, case_ ~ TCI7_scaled +
I(TCI7_scaled^2) + proba_human_scaled +
                        D_MSAVI_scaled + I(D_MSAVI_scaled^2) + fire_20y_scaled
+ shrub_density_scaled + I(shrub_density_scaled^2) +
                        sl_ + log_sl_ + cos_ta_ +
                        strata(step_id_), model = TRUE)



# We collect the first 18 rainy season models in a list for batch saving
list_cool2 <- list(
  issf_tci7_prev_landsat_cool_exp2 = issf_tci7_prev_landsat_cool_exp2,
```

```
  issf_tci7_prev_modis_cool_exp2 = issf_tci7_prev_modis_cool_exp2,
  issf_tci7_prev_dmsavi_cool_exp2 = issf_tci7_prev_dmsavi_cool_exp2,
  issf_tci7_20y_landsat_cool_exp2 = issf_tci7_20y_landsat_cool_exp2,
  issf_tci7_20y_modis_cool_exp2 = issf_tci7_20y_modis_cool_exp2,
  issf_tci7_20y_dmsavi_cool_exp2 = issf_tci7_20y_dmsavi_cool_exp2
)



# We save each model object to disk as an RDS file named after the model
for (name in names(list_cool2)) {
  model <- list_cool2[[name]]

  # Sauvegarde en .rds avec le nom du modèle
  file_path <- file.path(output_dir, paste0(name, ".rds"))
  saveRDS(model, file = file_path)
}
```

A.16. AICc analyses

A.16.1. Code for the AICc analyses

```
########################################
########### AICc Analyses #############
########################################

library(AICcmodavg)

########## Rainy ###########
# We define the folder path where our ISSF models are stored
folder <- "C:/Users/court/Documents/project/ISSF"

# We list all the files related to the rainy season models in this folder
files_rainy <- list.files(path = folder, pattern = "rainy.*\\.rds$", full.names
= TRUE)

# We load each rainy model file and extract the fitted model object from it
list_rainy <- lapply(files_rainy, function(f) {
  obj <- readRDS(f)
  return(obj$model)  # We extract the '$model' component from each saved list
})

# We assign meaningful names to the list elements based on the filenames
without extensions
names(list_rainy) <- tools::file_path_sans_ext(basename(files_rainy))



########## Hot ###########
# Similarly, we repeat the process for the hot season models

files_hot <- list.files(path = folder, pattern = "hot.*\\.rds$", full.names =
```

```
TRUE)

list_hot <- lapply(files_hot, function(f) {
  obj <- readRDS(f)
  return(obj$model)  # Extract the model object
})

names(list_hot) <- tools::file_path_sans_ext(basename(files_hot))


########## Cool ###########
# And again, for the cool season models

files_cool <- list.files(path = folder, pattern = "cool.*\\.rds$", full.names =
TRUE)

list_cool <- lapply(files_cool, function(f) {
  obj <- readRDS(f)
  return(obj$model)  # Extract the model
})

names(list_cool) <- tools::file_path_sans_ext(basename(files_cool))


#####################################
################ AIC #################
#####################################

# We calculate the AICc table to compare all rainy season candidate models with
second order correction
models_Comparison_rainy <- aictab(
  cand.set = list_rainy,           # Provide candidate models list
  modnames = names(list_rainy),    # Provide model names
  second.ord = TRUE                # Use AICc for small sample correction
)

# We do the same for hot season models
models_Comparison_hot <- aictab(
  cand.set = list_hot,
  modnames = names(list_hot),
  second.ord = TRUE
)

# And for cool season models as well
models_Comparison_cool <- aictab(
  cand.set = list_cool,
  modnames = names(list_cool),
  second.ord = TRUE
)

# Finally, we export these AICc comparison tables as CSV files for
documentation and further analysis
write.csv(models_Comparison_rainy, "C:/Users/court/Documents/AICc_rainy.csv",
row.names = FALSE)
write.csv(models_Comparison_hot, "C:/Users/court/Documents/AICc_hot.csv",
```

```
row.names = FALSE)
write.csv(models_Comparison_cool, "C:/Users/court/Documents/AICc_cool.csv",
row.names = FALSE)
```

A.16.2. AICc for hot dry season models

Table 2 : AICc analysis results for the hot dry season

Modnames	K	AICc	Delta_AICc	ModelLik	AICcWt	LL	Cum.Wt
issf_tci7_20y_landsat_hot_exp2	11	467183,75	0	1	1	-233580,88	1
issf_tci7_20y_landsat_hot_semi_linear	9	467241,17	57,421	3,4E-13	3,4E-13	-233611,59	1
issf_tci7_prev_landsat_hot_exp2	11	467273,4	89,646	3,42E-20	3,42E-20	-233625,7	1
issf_tci7_prev_landsat_hot_semi_linear	9	467332,67	148,92	4,6E-33	4,6E-33	-233657,33	1
issf_tci7_20y_landsat_hot_exp1	9	467341,48	157,72	5,63E-35	5,63E-35	-233661,74	1
issf_tci7_20y_landsat_hot	8	467344,35	160,59	1,34E-35	1,34E-35	-233664,17	1
issf_tci7_prev_landsat_hot_exp1	9	467431,35	247,6	1,72E-54	1,72E-54	-233706,67	1
issf_tci7_prev_landsat_hot	8	467434,38	250,63	3,77E-55	3,77E-55	-233709,19	1
issf_tpi7_20y_landsat_hot_exp2	11	467711,21	527,45	2,9E-115	2,9E-115	-233844,6	1
issf_tci_20y_landsat_hot_exp2	11	467741,27	557,52	8,6E-122	8,6E-122	-233859,64	1
issf_tci_20y_landsat_hot_semi_linear	10	467744,66	560,91	1,6E-122	1,6E-122	-233862,33	1
issf_tpi7_20y_landsat_hot_semi_linear	10	467752,27	568,52	3,5E-124	3,5E-124	-233866,13	1
issf_tpi7_prev_landsat_hot_exp2	11	467799,88	616,13	1,6E-134	1,6E-134	-233888,94	1
issf_tci_prev_landsat_hot_exp2	11	467829,13	645,37	7,2E-141	7,2E-141	-233903,56	1
issf_tci_prev_landsat_hot_semi_linear	10	467832,51	648,76	1,3E-141	1,3E-141	-233906,26	1
issf_tpi7_prev_landsat_hot_semi_linear	10	467840,57	656,82	2,4E-143	2,4E-143	-233910,28	1
issf_tpi7_20y_landsat_hot_exp1	9	467878,98	695,23	1,1E-151	1,1E-151	-233930,49	1
issf_tci_20y_landsat_hot_exp1	9	467909,41	725,66	2,7E-158	2,7E-158	-233945,71	1
issf_tpi3_20y_landsat_hot_exp2	11	467911,4	727,65	9,9E-159	9,9E-159	-233944,7	1
issf_tci_20y_landsat_hot	8	467912,22	728,47	6,5E-159	6,5E-159	-233948,11	1
issf_tpi7_20y_landsat_hot	8	467918,59	734,83	2,7E-160	2,7E-160	-233951,29	1
issf_tpi7_prev_landsat_hot_exp1	9	467967,89	784,14	5,3E-171	5,3E-171	-233974,95	1
issf_tpi3_20y_landsat_hot_semi_linear	10	467969,8	786,04	2,1E-171	2,1E-171	-233974,9	1
issf_tci_prev_landsat_hot_exp1	9	467997,51	813,75	2E-177	2E-177	-233989,75	1
issf_tpi3_prev_landsat_hot_exp2	11	467999,28	815,53	8,1E-178	8,1E-178	-233988,64	1
issf_tci_prev_landsat_hot	8	468000,31	816,56	4,9E-178	4,9E-178	-233992,15	1
issf_tpi7_prev_landsat_hot	8	468007,11	823,35	1,6E-179	1,6E-179	-233995,55	1
issf_tpi3_prev_landsat_hot_semi_linear	10	468057,81	874,06	1,6E-190	1,6E-190	-234018,91	1
issf_tpi3_20y_landsat_hot_exp1	9	468084,03	900,28	3,2E-196	3,2E-196	-234033,02	1
issf_tpi3_20y_landsat_hot	8	468141,91	958,16	8,7E-209	8,7E-209	-234062,96	1
issf_tpi3_prev_landsat_hot_exp1	9	468172,2	988,45	2,3E-215	2,3E-215	-234077,1	1
issf_tpi3_prev_landsat_hot	8	468230,19	1046,4	5,9E-228	5,9E-228	-234107,1	1
issf_tci7_20y_dmsavi_hot_exp2	11	471506,35	4322,6	0	0	-235742,18	1
issf_tci7_20y_modis_hot_exp2	11	471506,43	4322,7	0	0	-235742,22	1

issf_tci7_20y_dmsavi_hot_semi_linear	9	471562,98	4379,2	0	0	-235772,49	1
issf_tci7_20y_dmsavi_hot_exp1	9	471565,3	4381,5	0	0	-235773,65	1
issf_tci7_20y_modis_hot_exp1	9	471565,66	4381,9	0	0	-235773,83	1
issf_tci7_20y_modis_hot_semi_linear	9	471566,54	4382,8	0	0	-235774,27	1
issf_tci7_20y_dmsavi_hot	8	471568,37	4384,6	0	0	-235776,18	1
issf_tci7_20y_modis_hot	8	471568,75	4385	0	0	-235776,38	1
issf_tci7_prev_modis_hot_exp2	11	471582,35	4398,6	0	0	-235780,17	1
issf_tci7_prev_dmsavi_hot_exp2	11	471582,62	4398,9	0	0	-235780,31	1
issf_tci7_prev_dmsavi_hot_semi_linear	9	471641,31	4457,6	0	0	-235811,66	1
issf_tci7_prev_modis_hot_exp1	9	471642,74	4459	0	0	-235812,37	1
issf_tci7_prev_dmsavi_hot_exp1	9	471642,99	4459,2	0	0	-235812,5	1
issf_tci7_prev_modis_hot_semi_linear	9	471644,53	4460,8	0	0	-235813,26	1
issf_tci7_prev_modis_hot	8	471646,01	4462,3	0	0	-235815	1
issf_tci7_prev_dmsavi_hot	8	471646,23	4462,5	0	0	-235815,12	1
issf_tpi7_20y_dmsavi_hot_exp2	11	472020,97	4837,2	0	0	-235999,49	1
issf_tpi7_20y_modis_hot_exp2	11	472024,66	4840,9	0	0	-236001,33	1
issf_tci_20y_dmsavi_hot_exp2	11	472052,87	4869,1	0	0	-236015,44	1
issf_tci_20y_dmsavi_hot_semi_linear	10	472055,31	4871,6	0	0	-236017,66	1
issf_tci_20y_modis_hot_exp2	11	472056,64	4872,9	0	0	-236017,32	1
issf_tci_20y_modis_hot_semi_linear	10	472059,09	4875,3	0	0	-236019,55	1
issf_tpi7_20y_dmsavi_hot_semi_linear	10	472065,99	4882,2	0	0	-236022,99	1
issf_tpi7_20y_modis_hot_semi_linear	10	472069,72	4886	0	0	-236024,86	1
issf_tpi7_20y_dmsavi_hot_exp1	9	472081,64	4897,9	0	0	-236031,82	1
issf_tpi7_20y_modis_hot_exp1	9	472085,75	4902	0	0	-236033,88	1
issf_tpi7_prev_dmsavi_hot_exp2	11	472096,12	4912,4	0	0	-236037,06	1
issf_tpi7_prev_modis_hot_exp2	11	472099,44	4915,7	0	0	-236038,72	1
issf_tci_20y_dmsavi_hot_exp1	9	472113,3	4929,5	0	0	-236047,65	1
issf_tci_20y_dmsavi_hot	8	472115,46	4931,7	0	0	-236049,73	1
issf_tci_20y_modis_hot_exp1	9	472117,48	4933,7	0	0	-236049,74	1
issf_tci_20y_modis_hot	8	472119,62	4935,9	0	0	-236051,81	1
issf_tpi7_20y_dmsavi_hot	8	472125,34	4941,6	0	0	-236054,67	1
issf_tci_prev_dmsavi_hot_exp2	11	472127,3	4943,6	0	0	-236052,65	1
issf_tpi7_20y_modis_hot	8	472129,38	4945,6	0	0	-236056,69	1
issf_tci_prev_dmsavi_hot_semi_linear	10	472129,75	4946	0	0	-236054,88	1
issf_tci_prev_modis_hot_exp2	11	472130,71	4947	0	0	-236054,35	1
issf_tci_prev_modis_hot_semi_linear	10	472133,17	4949,4	0	0	-236056,58	1
issf_tpi7_prev_dmsavi_hot_semi_linear	10	472140,8	4957,1	0	0	-236060,4	1
issf_tpi7_prev_modis_hot_semi_linear	10	472144,16	4960,4	0	0	-236062,08	1
issf_tpi7_prev_dmsavi_hot_exp1	9	472158,29	4974,5	0	0	-236070,14	1
issf_tpi7_prev_modis_hot_exp1	9	472161,78	4978	0	0	-236071,89	1
issf_tci_prev_dmsavi_hot_exp1	9	472189,22	5005,5	0	0	-236085,61	1
issf_tci_prev_dmsavi_hot	8	472191,38	5007,6	0	0	-236087,69	1
issf_tci_prev_modis_hot_exp1	9	472192,79	5009	0	0	-236087,39	1
issf_tci_prev_modis_hot	8	472194,93	5011,2	0	0	-236089,47	1
issf_tpi7_prev_dmsavi_hot	8	472201,63	5017,9	0	0	-236092,81	1
issf_tpi7_prev_modis_hot	8	472205,05	5021,3	0	0	-236094,53	1

issf_tpi3_20y_dmsavi_hot_exp2	11	472213,97	5030,2	0	0	-236095,99	1
issf_tpi3_20y_modis_hot_exp2	11	472218,57	5034,8	0	0	-236098,29	1
issf_tpi3_20y_dmsavi_hot_exp1	9	472276,54	5092,8	0	0	-236129,27	1
issf_tpi3_20y_dmsavi_hot_semi_linear	10	472279,29	5095,5	0	0	-236129,65	1
issf_tpi3_20y_modis_hot_exp1	9	472281,59	5097,8	0	0	-236131,79	1
issf_tpi3_20y_modis_hot_semi_linear	10	472283,9	5100,2	0	0	-236131,95	1
issf_tpi3_prev_dmsavi_hot_exp2	11	472288,3	5104,5	0	0	-236133,15	1
issf_tpi3_prev_modis_hot_exp2	11	472292,52	5108,8	0	0	-236135,26	1
issf_tpi3_20y_dmsavi_hot	8	472341,27	5157,5	0	0	-236162,64	1
issf_tpi3_20y_modis_hot	8	472346,29	5162,5	0	0	-236165,15	1
issf_tpi3_prev_dmsavi_hot_exp1	9	472352,42	5168,7	0	0	-236167,21	1
issf_tpi3_prev_dmsavi_hot_semi_linear	10	472353,85	5170,1	0	0	-236166,93	1
issf_tpi3_prev_modis_hot_exp1	9	472356,85	5173,1	0	0	-236169,42	1
issf_tpi3_prev_modis_hot_semi_linear	10	472358,08	5174,3	0	0	-236169,04	1
issf_tpi3_prev_dmsavi_hot	8	472417,37	5233,6	0	0	-236200,69	1
issf_tpi3_prev_modis_hot	8	472421,76	5238	0	0	-236202,88	1

A.16.3. AICc for rainy season models

Table 3 : AICc analysis results for the rainy season

Modnames	K	AICc	Delta_AICc	ModelLik	AICcWt	LL	Cum.Wt
issf_tci7_20y_landsat_rainy_exp2	11	335351,7701	0	1	0,999989	-167664,88	1
issf_tci7_20y_landsat_rainy_semi_linear	9	335374,6759	22,91	1,06E-05	1,06E-05	-167678,34	1
issf_tci_20y_landsat_rainy_exp2	11	335397,007	45,24	1,5E-10	1,5E-10	-167687,5	1
issf_tci_20y_landsat_rainy_semi_linear	10	335404,1893	52,42	4,14E-12	4,14E-12	-167692,09	1
issf_tpi7_20y_landsat_rainy_exp2	11	335410,135	58,36	2,12E-13	2,12E-13	-167694,07	1
issf_tpi7_20y_landsat_rainy_semi_linear	10	335412,3729	60,6	6,92E-14	6,92E-14	-167696,19	1
issf_tci7_prev_landsat_rainy_exp2	11	335413,8034	62,03	3,39E-14	3,39E-14	-167695,9	1
issf_tpi3_20y_landsat_rainy_semi_linear	10	335425,1505	73,38	1,16E-16	1,16E-16	-167702,58	1
issf_tpi3_20y_landsat_rainy_exp2	11	335426,2087	74,44	6,85E-17	6,85E-17	-167702,1	1
issf_tci7_prev_landsat_rainy_semi_linear	9	335436,4214	84,65	4,15E-19	4,15E-19	-167709,21	1
issf_tci_prev_landsat_rainy_exp2	11	335458,9705	107,2	5,27E-24	5,27E-24	-167718,49	1
issf_tci_prev_landsat_rainy_semi_linear	10	335466,2922	114,5	1,35E-25	1,35E-25	-167723,15	1
issf_tpi7_prev_landsat_rainy_exp2	11	335472,0295	120,3	7,69E-27	7,69E-27	-167725,01	1
issf_tpi7_prev_landsat_rainy_semi_linear	10	335474,4883	122,7	2,25E-27	2,25E-27	-167727,24	1
issf_tpi3_prev_landsat_rainy_semi_linear	10	335487,1095	135,3	4,09E-30	4,09E-30	-167733,55	1
issf_tpi3_prev_landsat_rainy_exp2	11	335488,1176	136,3	2,47E-30	2,47E-30	-167733,06	1
issf_tci7_20y_landsat_rainy_exp1	9	335685,331	333,6	3,7E-73	3,7E-73	-167833,67	1
issf_tci7_20y_landsat_rainy	8	335700,492	348,7	1,89E-76	1,89E-76	-167842,25	1
issf_tci_20y_landsat_rainy_exp1	9	335727,7296	376	2,3E-82	2,3E-82	-167854,86	1
issf_tci_20y_landsat_rainy	8	335734,8882	383,1	6,41E-84	6,41E-84	-167859,44	1
issf_tpi7_20y_landsat_rainy_exp1	9	335740,8572	389,1	3,24E-85	3,24E-85	-167861,43	1

issf_tpi7_20y_landsat_rainy	8	335743,1792	391,4	1,02E-85	1,02E-85	-167863,59	1
issf_tci7_prev_landsat_rainy_exp1	9	335747,1731	395,4	1,38E-86	1,38E-86	-167864,59	1
issf_tpi3_20y_landsat_rainy	8	335755,1136	403,3	2,6E-88	2,6E-88	-167869,56	1
issf_tpi3_20y_landsat_rainy_exp1	9	335756,1602	404,4	1,54E-88	1,54E-88	-167869,08	1
issf_tci7_prev_landsat_rainy	8	335762,6219	410,9	6,09E-90	6,09E-90	-167873,31	1
issf_tci_prev_landsat_rainy_exp1	9	335789,5095	437,7	8,83E-96	8,83E-96	-167885,75	1
issf_tci_prev_landsat_rainy	8	335796,8024	445	2,3E-97	2,3E-97	-167890,4	1
issf_tpi7_prev_landsat_rainy_exp1	9	335802,552	450,8	1,3E-98	1,3E-98	-167892,28	1
issf_tpi7_prev_landsat_rainy	8	335805,1004	453,3	3,6E-99	3,6E-99	-167894,55	1
issf_tpi3_prev_landsat_rainy	8	335816,8861	465,1	1E-101	1E-101	-167900,44	1
issf_tpi3_prev_landsat_rainy_exp1	9	335817,8796	466,1	6,1E-102	6,1E-102	-167899,94	1
issf_tci7_20y_dmsavi_rainy_exp2	11	338704,6461	3353	0	0	-169341,32	1
issf_tci7_20y_dmsavi_rainy_exp1	9	338718,1452	3366	0	0	-169350,07	1
issf_tci7_20y_modis_rainy_exp2	11	338728,6817	3377	0	0	-169353,34	1
issf_tci7_20y_dmsavi_rainy	8	338732,7592	3381	0	0	-169358,38	1
issf_tci7_20y_dmsavi_rainy_semi_linear	9	338734,1774	3382	0	0	-169358,09	1
issf_tci_20y_dmsavi_rainy_exp2	11	338752,6956	3401	0	0	-169365,35	1
issf_tci7_20y_modis_rainy_semi_linear	9	338756,5665	3405	0	0	-169369,28	1
issf_tci_20y_dmsavi_rainy_semi_linear	10	338759,3066	3408	0	0	-169369,65	1
issf_tci7_prev_dmsavi_rainy_exp2	11	338764,7245	3413	0	0	-169371,36	1
issf_tci_20y_dmsavi_rainy_exp1	9	338764,9227	3413	0	0	-169373,46	1
issf_tpi7_20y_dmsavi_rainy_exp2	11	338766,6692	3415	0	0	-169372,33	1
issf_tpi7_20y_dmsavi_rainy_semi_linear	10	338768,6744	3417	0	0	-169374,34	1
issf_tci_20y_dmsavi_rainy	8	338771,4558	3420	0	0	-169377,73	1
issf_tci7_20y_modis_rainy_exp1	9	338773,7802	3422	0	0	-169377,89	1
issf_tci_20y_modis_rainy_exp2	11	338776,4643	3425	0	0	-169377,23	1
issf_tci7_prev_dmsavi_rainy_exp1	9	338777,727	3426	0	0	-169379,86	1
issf_tpi7_20y_dmsavi_rainy_exp1	9	338778,8579	3427	0	0	-169380,43	1
issf_tpi7_20y_dmsavi_rainy	8	338780,7475	3429	0	0	-169382,37	1
issf_tpi3_20y_dmsavi_rainy_semi_linear	10	338782,4179	3431	0	0	-169381,21	1
issf_tci_20y_modis_rainy_semi_linear	10	338783,1574	3431	0	0	-169381,58	1
issf_tpi3_20y_dmsavi_rainy_exp2	11	338783,4949	3432	0	0	-169380,75	1
issf_tci7_20y_modis_rainy	8	338788,4529	3437	0	0	-169386,23	1
issf_tci7_prev_modis_rainy_exp2	11	338788,8366	3437	0	0	-169383,42	1
issf_tpi7_20y_modis_rainy_exp2	11	338790,4811	3439	0	0	-169384,24	1
issf_tpi7_20y_modis_rainy_semi_linear	10	338792,5626	3441	0	0	-169386,28	1
issf_tci7_prev_dmsavi_rainy	8	338792,6271	3441	0	0	-169388,31	1
issf_tci7_prev_dmsavi_rainy_semi_linear	9	338793,7814	3442	0	0	-169387,89	1
issf_tpi3_20y_dmsavi_rainy	8	338794,0317	3442	0	0	-169389,02	1
issf_tpi3_20y_dmsavi_rainy_exp1	9	338795,1864	3443	0	0	-169388,59	1
issf_tpi3_20y_modis_rainy_semi_linear	10	338806,2366	3454	0	0	-169393,12	1
issf_tpi3_20y_modis_rainy_exp2	11	338807,288	3456	0	0	-169392,64	1
issf_tci_prev_dmsavi_rainy_exp2	11	338812,6535	3461	0	0	-169395,33	1
issf_tci7_prev_modis_rainy_semi_linear	9	338816,2833	3465	0	0	-169399,14	1
issf_tci_prev_dmsavi_rainy_semi_linear	10	338819,399	3468	0	0	-169399,7	1
issf_tci_20y_modis_rainy_exp1	9	338820,4913	3469	0	0	-169401,25	1

issf_tci_prev_dmsavi_rainy_exp1	9	338824,4029	3473	0	0	-169403,2	1
issf_tpi7_prev_dmsavi_rainy_exp2	11	338826,5335	3475	0	0	-169402,27	1
issf_tci_20y_modis_rainy	8	338827,0545	3475	0	0	-169405,53	1
issf_tpi7_prev_dmsavi_rainy_semi_linear	10	338828,7627	3477	0	0	-169404,38	1
issf_tci_prev_dmsavi_rainy	8	338831,0717	3479	0	0	-169407,54	1
issf_tci7_prev_modis_rainy_exp1	9	338834,3539	3483	0	0	-169408,18	1
issf_tpi7_20y_modis_rainy_exp1	9	338834,4168	3483	0	0	-169408,21	1
issf_tpi7_20y_modis_rainy	8	338836,3714	3485	0	0	-169410,19	1
issf_tci_prev_modis_rainy_exp2	11	338836,4873	3485	0	0	-169407,24	1
issf_tpi7_prev_dmsavi_rainy_exp1	9	338838,2489	3486	0	0	-169410,12	1
issf_tpi7_prev_dmsavi_rainy	8	338840,3617	3489	0	0	-169412,18	1
issf_tpi3_prev_dmsavi_rainy_semi_linear	10	338842,3286	3491	0	0	-169411,16	1
issf_tci_prev_modis_rainy_semi_linear	10	338843,3178	3492	0	0	-169411,66	1
issf_tpi3_prev_dmsavi_rainy_exp2	11	338843,3549	3492	0	0	-169410,68	1
issf_tci7_prev_modis_rainy	8	338849,3137	3498	0	0	-169416,66	1
issf_tpi3_20y_modis_rainy	8	338849,5922	3498	0	0	-169416,8	1
issf_tpi7_prev_modis_rainy_exp2	11	338850,4158	3499	0	0	-169414,21	1
issf_tpi3_20y_modis_rainy_exp1	9	338850,733	3499	0	0	-169416,37	1
issf_tpi7_prev_modis_rainy_semi_linear	10	338852,7212	3501	0	0	-169416,36	1
issf_tpi3_prev_dmsavi_rainy	8	338853,4817	3502	0	0	-169418,74	1
issf_tpi3_prev_dmsavi_rainy_exp1	9	338854,5863	3503	0	0	-169418,29	1
issf_tpi3_prev_modis_rainy_semi_linear	10	338866,2186	3514	0	0	-169423,11	1
issf_tpi3_prev_modis_rainy_exp2	11	338867,2198	3515	0	0	-169422,61	1
issf_tci_prev_modis_rainy_exp1	9	338880,9613	3529	0	0	-169431,48	1
issf_tci_prev_modis_rainy	8	338887,6609	3536	0	0	-169435,83	1
issf_tpi7_prev_modis_rainy_exp1	9	338894,7975	3543	0	0	-169438,4	1
issf_tpi7_prev_modis_rainy	8	338896,9768	3545	0	0	-169440,49	1
issf_tpi3_prev_modis_rainy	8	338910,0331	3558	0	0	-169447,02	1
issf_tpi3_prev_modis_rainy_exp1	9	338911,1233	3559	0	0	-169446,56	1

A.16.4. AICc for cool dry season

Table 4 : AICc analysis for the cool dry season

Modnames	K	AICc	Delta_AICc	ModelLik	AICcWt	LL	Cum.Wt
issf_tci7_20y_landsat_cool_exp2	11	510913,65	0	1	0,8223931	-255445,82	0,8
issf_tci7_prev_landsat_cool_exp2	11	510916,71	3,0653	0,215963	0,1776069	-255447,36	1
issf_tci7_20y_landsat_cool_semi_linear	9	510956,87	43,228	4,10E-10	3,37E-10	-255469,44	1
issf_tci7_prev_landsat_cool_semi_linear	9	510959,61	45,965	1,04E-10	8,59E-11	-255470,81	1
issf_tpi7_20y_landsat_cool_exp2	11	511320,15	406,51	5,35E-89	4,40E-89	-255649,08	1
issf_tpi7_prev_landsat_cool_exp2	11	511323,18	409,53	1,18E-89	9,70E-90	-255650,59	1
issf_tpi7_20y_landsat_cool_semi_linear	10	511326,58	412,93	2,15E-90	1,77E-90	-255653,29	1
issf_tpi7_prev_landsat_cool_semi_linear	10	511329,62	415,98	4,69E-91	3,86E-91	-255654,81	1

issf_tci_20y_landsat_cool_exp2	11	511361,55	447,91	5,48E-98	4,50E-98	-255669,78	1
issf_tci_20y_landsat_cool_semi_linear	10	511362,02	448,37	4,34E-98	3,57E-98	-255671,01	1
issf_tci_prev_landsat_cool_exp2	11	511364,64	450,99	1,17E-98	9,61E-99	-255671,32	1
issf_tci_prev_landsat_cool_semi_linear	10	511365,11	451,46	9,26E-99	7,62E-99	-255672,55	1
issf_tpi3_20y_landsat_cool_exp2	11	511438,07	524,43	1,33E-114	1,09E-114	-255708,04	1
issf_tpi3_prev_landsat_cool_exp2	11	511441,22	527,57	2,75E-115	2,26E-115	-255709,61	1
issf_tpi3_20y_landsat_cool_semi_linear	10	511446,29	532,64	2,18E-116	1,79E-116	-255713,14	1
issf_tpi3_prev_landsat_cool_semi_linear	10	511449,42	535,78	4,55E-117	3,74E-117	-255714,71	1
issf_tci7_20y_landsat_cool	8	511963,09	1049,4	1,31E-228	1,07E-228	-255973,55	1
issf_tci7_prev_landsat_cool	8	511964,01	1050,4	8,26E-229	6,79E-229	-255974	1
issf_tci7_20y_landsat_cool_exp1	9	511965,03	1051,4	4,95E-229	4,07E-229	-255973,52	1
issf_tci7_prev_landsat_cool_exp1	9	511965,95	1052,3	3,13E-229	2,57E-229	-255973,98	1
issf_tpi7_20y_landsat_cool_exp1	9	512353,18	1439,5	0	0	-256167,59	1
issf_tpi7_prev_landsat_cool_exp1	9	512354,11	1440,5	0	0	-256168,05	1
issf_tpi7_20y_landsat_cool	8	512358,06	1444,4	0	0	-256171,03	1
issf_tpi7_prev_landsat_cool	8	512359,01	1445,4	0	0	-256171,5	1
issf_tci_20y_landsat_cool_exp1	9	512390,63	1477	0	0	-256186,32	1
issf_tci_20y_landsat_cool	8	512390,92	1477,3	0	0	-256187,46	1
issf_tci_prev_landsat_cool_exp1	9	512391,6	1478	0	0	-256186,8	1
issf_tci_prev_landsat_cool	8	512391,89	1478,2	0	0	-256187,94	1
issf_tpi3_20y_landsat_cool_exp1	9	512464,14	1550,5	0	0	-256223,07	1
issf_tpi3_prev_landsat_cool_exp1	9	512465,17	1551,5	0	0	-256223,59	1
issf_tpi3_20y_landsat_cool	8	512471,13	1557,5	0	0	-256227,57	1
issf_tpi3_prev_landsat_cool	8	512472,16	1558,5	0	0	-256228,08	1
issf_tci7_20y_modis_cool_exp2	11	516170,09	5256,4	0	0	-258074,05	1
issf_tci7_prev_modis_cool_exp2	11	516170,27	5256,6	0	0	-258074,14	1
issf_tci7_20y_modis_cool_semi_linear	9	516195,14	5281,5	0	0	-258088,57	1
issf_tci7_prev_modis_cool_semi_linear	9	516195,2	5281,6	0	0	-258088,6	1
issf_tci7_20y_modis_cool	8	516366,08	5452,4	0	0	-258175,04	1
issf_tci7_prev_modis_cool	8	516366,17	5452,5	0	0	-258175,08	1
issf_tci7_20y_modis_cool_exp1	9	516367,46	5453,8	0	0	-258174,73	1
issf_tci7_prev_modis_cool_exp1	9	516367,56	5453,9	0	0	-258174,78	1
issf_tci7_prev_dmsavi_cool_exp2	11	516505,02	5591,4	0	0	-258241,51	1
issf_tci7_20y_dmsavi_cool_exp2	11	516505,13	5591,5	0	0	-258241,56	1
issf_tci7_prev_dmsavi_cool_semi_linear	9	516520,37	5606,7	0	0	-258251,19	1
issf_tci7_20y_dmsavi_cool_semi_linear	9	516520,52	5606,9	0	0	-258251,26	1
issf_tci7_prev_dmsavi_cool	8	516523,55	5609,9	0	0	-258253,77	1
issf_tci7_20y_dmsavi_cool	8	516523,68	5610	0	0	-258253,84	1
issf_tci7_prev_dmsavi_cool_exp1	9	516524,95	5611,3	0	0	-258253,47	1
issf_tci7_20y_dmsavi_cool_exp1	9	516525,07	5611,4	0	0	-258253,54	1

issf_tpi7_20y_modis_cool_exp2	11	516540,56	5626,9	0	0	-258259,28	1
issf_tpi7_prev_modis_cool_exp2	11	516540,72	5627,1	0	0	-258259,36	1
issf_tpi7_20y_modis_cool_semi_linear	10	516545,44	5631,8	0	0	-258262,72	1
issf_tpi7_prev_modis_cool_semi_linear	10	516545,62	5632	0	0	-258262,81	1
issf_tci_20y_modis_cool_exp2	11	516574,97	5661,3	0	0	-258276,48	1
issf_tci_20y_modis_cool_semi_linear	10	516575,12	5661,5	0	0	-258277,56	1
issf_tci_prev_modis_cool_exp2	11	516575,15	5661,5	0	0	-258276,58	1
issf_tci_prev_modis_cool_semi_linear	10	516575,3	5661,7	0	0	-258277,65	1
issf_tpi3_20y_modis_cool_exp2	11	516644,63	5731	0	0	-258311,32	1
issf_tpi3_prev_modis_cool_exp2	11	516644,85	5731,2	0	0	-258311,42	1
issf_tpi3_20y_modis_cool_semi_linear	10	516652,64	5739	0	0	-258316,32	1
issf_tpi3_prev_modis_cool_semi_linear	10	516652,85	5739,2	0	0	-258316,42	1
issf_tpi7_20y_modis_cool_exp1	9	516741,76	5828,1	0	0	-258361,88	1
issf_tpi7_prev_modis_cool_exp1	9	516741,84	5828,2	0	0	-258361,92	1
issf_tpi7_20y_modis_cool	8	516746,2	5832,6	0	0	-258365,1	1
issf_tpi7_prev_modis_cool	8	516746,29	5832,6	0	0	-258365,14	1
issf_tci_20y_modis_cool	8	516776,09	5862,4	0	0	-258380,05	1
issf_tci_20y_modis_cool_exp1	9	516776,15	5862,5	0	0	-258379,07	1
issf_tci_prev_modis_cool	8	516776,19	5862,5	0	0	-258380,09	1
issf_tci_prev_modis_cool_exp1	9	516776,24	5862,6	0	0	-258379,12	1
issf_tpi3_20y_modis_cool_exp1	9	516846,84	5933,2	0	0	-258414,42	1
issf_tpi3_prev_modis_cool_exp1	9	516846,96	5933,3	0	0	-258414,48	1
issf_tpi3_20y_modis_cool	8	516854,53	5940,9	0	0	-258419,26	1
issf_tpi3_prev_modis_cool	8	516854,64	5941	0	0	-258419,32	1
issf_tpi7_prev_dmsavi_cool_exp2	11	516864,73	5951,1	0	0	-258421,36	1
issf_tpi7_20y_dmsavi_cool_exp2	11	516864,82	5951,2	0	0	-258421,41	1
issf_tpi7_prev_dmsavi_cool_semi_linear	10	516869,56	5955,9	0	0	-258424,78	1
issf_tpi7_20y_dmsavi_cool_semi_linear	10	516869,65	5956	0	0	-258424,83	1
issf_tpi7_prev_dmsavi_cool_exp1	9	516888,22	5974,6	0	0	-258435,11	1
issf_tpi7_20y_dmsavi_cool_exp1	9	516888,34	5974,7	0	0	-258435,17	1
issf_tpi7_prev_dmsavi_cool	8	516892,75	5979,1	0	0	-258438,37	1
issf_tpi7_20y_dmsavi_cool	8	516892,86	5979,2	0	0	-258438,43	1
issf_tci_prev_dmsavi_cool_exp2	11	516898,05	5984,4	0	0	-258438,03	1
issf_tci_20y_dmsavi_cool_exp2	11	516898,15	5984,5	0	0	-258438,07	1
issf_tci_prev_dmsavi_cool_semi_linear	10	516898,28	5984,6	0	0	-258439,14	1
issf_tci_20y_dmsavi_cool_semi_linear	10	516898,38	5984,7	0	0	-258439,19	1
issf_tci_prev_dmsavi_cool_exp1	9	516921,76	6008,1	0	0	-258451,88	1
issf_tci_prev_dmsavi_cool	8	516921,86	6008,2	0	0	-258452,93	1
issf_tci_20y_dmsavi_cool_exp1	9	516921,88	6008,2	0	0	-258451,94	1
issf_tci_20y_dmsavi_cool	8	516921,98	6008,3	0	0	-258452,99	1
issf_tpi3_prev_dmsavi_cool_exp2	11	516965,23	6051,6	0	0	-258471,61	1
issf_tpi3_20y_dmsavi_cool_exp2	11	516965,3	6051,7	0	0	-258471,65	1
issf_tpi3_prev_dmsavi_cool_semi_linear	10	516973,3	6059,7	0	0	-258476,65	1
issf_tpi3_20y_dmsavi_cool_semi_linear	10	516973,38	6059,7	0	0	-258476,69	1
issf_tpi3_prev_dmsavi_cool_exp1	9	516990,08	6076,4	0	0	-258486,04	1
issf_tpi3_20y_dmsavi_cool_exp1	9	516990,17	6076,5	0	0	-258486,08	1

issf_tpi3_prev_dmsavi_cool	8	516997,92	6084,3	0	0	-258490,96	1
issf_tpi3_20y_dmsavi_cool	8	516998,02	6084,4	0	0	-258491,01	1

A.17. Code to save the best ISSF model in CSV format

```
################################################
############ Save the ISSF models ##############
################################################

# We extract the rainy season ISSF model object and convert its summary to a
dataframe
rainy <- issf_tci7_20y_landsat_rainy_exp2$model
rainy_table <- as.data.frame(coef(summary(rainy)))

# We add a 'variable' column with row names and reorder columns for clarity
rainy_table$variable <- rownames(rainy_table)
names(rainy_table) <- c("coef","exp_coef","se_coef","z","p","variable")
rainy_table <- rainy_table[, c("variable","coef","exp_coef","se_coef","z","p")]

# We save the rainy season coefficients table to CSV
write.csv(rainy_table, "C:/Users/court/Documents/best_ISSF_csv/rainy.csv",
row.names = FALSE)



# We repeat the same process for the hot dry season model
hot_dry <- issf_tci7_20y_landsat_hot_exp2$model
hot_dry_table <- as.data.frame(coef(summary(hot_dry)))
hot_dry_table$variable <- rownames(hot_dry_table)
names(hot_dry_table) <- c("coef", "exp_coef", "se_coef", "z", "p", "variable")
hot_dry_table <- hot_dry_table[, c("variable", "coef", "exp_coef", "se_coef",
"z", "p")]
write.csv(hot_dry_table, "C:/Users/court/Documents/best_ISSF_csv/hot_dry.csv",
row.names = FALSE)



# We repeat the process for the cool dry season model
cool_dry <- issf_tci7_20y_landsat_cool_exp2$model
cool_dry_table <- as.data.frame(coef(summary(cool_dry)))
cool_dry_table$variable <- rownames(cool_dry_table)
names(cool_dry_table) <- c("coef", "exp_coef", "se_coef", "z", "p", "variable")
cool_dry_table <- cool_dry_table[, c("variable", "coef", "exp_coef", "se_coef",
"z", "p")]
write.csv(cool_dry_table,
"C:/Users/court/Documents/best_ISSF_csv/cool_dry.csv", row.names = FALSE)
```

A.18. Code for RSS creation

```
##############################################
############# RSS calculations ##############
##############################################

# Download the necessary packages
#install.packages("dplyr")
#install.packages("amt")
#install.packages("tidyr")
#install.packages("terra")


# Load the required libraries
library(amt)
library(ggplot2)
library(dplyr)
library(terra)

# We load the scaled covariate datasets for each season separately and combined
cov_scaled_cool <- readRDS(".../cov_cool_scaled.rds")
cov_scaled_rainy <- readRDS(".../cov_rainy_scaled.rds")
cov_scaled_hot <- readRDS(".../cov_hot_scaled.rds")
cov_scaled <- readRDS(".../cov_scaled.rds")

############# HUMAN ACTIVITY ##############

# 1. We retrieve the center and scale attributes used to standardize
'proba_human' in the combined dataset
center_human <- attr(cov_scaled$proba_human_scaled, "scaled:center")
scale_human <- attr(cov_scaled$proba_human_scaled, "scaled:scale")

# 2. We create a sequence of unscaled 'proba_human' values covering the central
98% of data
raw_seq_human <- seq(quantile(cov_scaled$proba_human, 0.01, na.rm=TRUE),
                     quantile(cov_scaled$proba_human, 0.99, na.rm=TRUE),
length.out=100)

# 3. We standardize this sequence using the previously retrieved center and
scale, to match model input scale
scaled_seq_human <- (raw_seq_human - center_human) / scale_human

# 4. We prepare a data frame 'x1_human' where 'proba_human_scaled' varies over
the sequence,
#    while other covariates are fixed at reference values (mostly zero or
median movement stats)
x1_human <- data.frame(
  TCI7_scaled = 0, `I(TCI7_scaled^2)` = 0, MSAVI_landsat_scaled = 0,
`I(MSAVI_landsat_scaled^2)` = 0,
  proba_human_scaled = scaled_seq_human, fire_20y_scaled = 0,
shrub_density_scaled = 0,
  `I(shrub_density_scaled^2)` = 0, sl_ = 550, log_sl_ = log(550), cos_ta_ = 0)

# 5. We define 'x2_human' as a reference covariate vector with all values fixed
at zero (reference level)
x2_human <- data.frame(
  proba_human_scaled = 0, MSAVI_landsat_scaled = 0, `I(MSAVI_landsat_scaled^2)`
= 0,
```

```
  TCI7_scaled = 0, `I(TCI7_scaled^2)` = 0, fire_20y_scaled = 0,
shrub_density_scaled = 0,
  `I(shrub_density_scaled^2)` = 0, sl_ = 550, log_sl_ = log(550), cos_ta_ = 0)

########## Rainy Season

# We load the fitted ISSF model for the rainy season with TCI7, fire_20y, and
landsat MSAVI
issf_tci7_20y_landsat_rainy_exp2 <-
readRDS(".../issf_tci7_20y_landsat_rainy_exp2.rds")

# We compute the log Relative Selection Strength (RSS) for the sequence of
human activity values,
# comparing x1_human vs reference x2_human, including standard error confidence
intervals at 95%
logRSS_raw_rainy_human <- log_rss(issf_tci7_20y_landsat_rainy_exp2, x1 =
x1_human,
                                  x2 = x2_human, ci = "se", ci_level = 0.95)

########## Hot Season

# We repeat the same loading and computation for the hot dry season
issf_tci7_20y_landsat_hot_exp2 <-
readRDS(".../issf_tci7_20y_landsat_hot_exp2.rds")

logRSS_raw_hot_human <- log_rss(issf_tci7_20y_landsat_hot_exp2, x1 = x1_human,
                                x2 = x2_human, ci = "se", ci_level = 0.95)

########## Cool Season

# Same for the cool dry season
issf_tci7_20y_landsat_cool_exp2 <-
readRDS(".../issf_tci7_20y_landsat_cool_exp2.rds")

logRSS_raw_cool_human <- log_rss(issf_tci7_20y_landsat_cool_exp2, x1 =
x1_human,
                                 x2 = x2_human, ci = "se", ci_level = 0.95)

########## Combine Hot, Rainy and Cool Seasons

# We add a season label and the raw unscaled sequence for plotting in each data
frame
df_hot_human <- logRSS_raw_hot_human$df %>% mutate(source = "hot",
raw_seq_human = raw_seq_human)
df_rainy_human <- logRSS_raw_rainy_human$df %>% mutate(source = "rainy",
raw_seq_human = raw_seq_human)
df_cool_human <- logRSS_raw_cool_human$df %>% mutate(source = "cool",
raw_seq_human = raw_seq_human)

# We combine all seasons' data for plotting
df_3_human <- bind_rows(df_hot_human, df_rainy_human, df_cool_human)

# We create a ggplot showing RSS (exp of logRSS) over the range of human
activity,
# with confidence ribbons and colored by season
human_3_seasons_ggplot <- ggplot(df_3_human, aes(x = raw_seq_human)) +
  geom_ribbon(aes(ymin = exp(lwr), ymax = exp(upr), fill = source), alpha =
```

```
0.2, color = NA) +
  geom_line(aes(y = exp(log_rss), color = source)) +
  geom_line(aes(y = exp(lwr), color = source), linetype = "dashed") +
  geom_line(aes(y = exp(upr), color = source), linetype = "dashed") +
  geom_hline(yintercept = 1, linetype = "dashed") +  # Reference RSS = 1 line
  scale_color_manual(values = c("hot" = "red", "rainy" = "blue", "cool" =
"green"),
                     labels = c("hot" = "hot dry season", "rainy" = "rainy
season", "cool" = "cool dry season")) +
  scale_fill_manual(values = c("hot" = "red", "rainy" = "blue", "cool" =
"green"),
                    labels = c("hot" = "hot dry season", "rainy" = "rainy
season", "cool" = "cool dry season")) +
  labs(x = "Probability of human activity", y = "RSS", color = "Legend", fill =
"Legend") +
  theme_classic()

# (Optional) We can save the figure with ggsave - adapt path as needed
# ggsave(".../RSS_human_unscaled.png", plot = human_3_seasons_ggplot, width =
8, height = 6, dpi = 300)


########### TCI ##############

# We repeat the same general steps for the Topographic Convergence Index (TCI)
covariate:

# Retrieve center and scale used to standardize TCI7
center_tci <- attr(cov_scaled$TCI7_scaled, "scaled:center")
scale_tci  <- attr(cov_scaled$TCI7_scaled, "scaled:scale")

# Define a sequence of unscaled TCI values spanning central 98%
raw_seq_tci <- seq(quantile(cov_scaled$TCI_7x7, 0.01, na.rm=TRUE),
                   quantile(cov_scaled$TCI_7x7, 0.99, na.rm=TRUE),
length.out=100)

# Standardize the sequence to match model input scale
scaled_seq_tci <- (raw_seq_tci - center_tci) / scale_tci

# Prepare data frame for varying TCI, other covariates fixed
x1_TCI <- data.frame(
  proba_human_scaled = 0, MSAVI_landsat_scaled = 0, `I(MSAVI_landsat_scaled^2)`
= 0,
  TCI7_scaled = scaled_seq_tci, `I(TCI7_scaled^2)` = scaled_seq_tci^2,
fire_20y_scaled = 0,
  shrub_density_scaled = 0, `I(shrub_density_scaled^2)` = 0, sl_ = 550, log_sl_
= log(550), cos_ta_ = 0)

# Reference covariates fixed at zero
x2_TCI <- data.frame(
  proba_human_scaled = 0, MSAVI_landsat_scaled = 0, `I(MSAVI_landsat_scaled^2)`
= 0,
  TCI7_scaled = 0, `I(TCI7_scaled^2)` = 0, fire_20y_scaled = 0,
shrub_density_scaled = 0,
  `I(shrub_density_scaled^2)` = 0, sl_ = 550, log_sl_ = log(550), cos_ta_ = 0)

# Load models and compute logRSS with confidence intervals for rainy, hot, and
```

```
cool seasons
issf_tci7_20y_landsat_rainy_exp2 <-
readRDS(".../issf_tci7_20y_landsat_rainy_exp2.rds")
logRSS_raw_rainy_tci <- log_rss(issf_tci7_20y_landsat_rainy_exp2, x1 = x1_TCI,
x2 = x2_TCI, ci = "se", ci_level = 0.95)

issf_tci7_20y_landsat_hot_exp2 <-
readRDS(".../issf_tci7_20y_landsat_hot_exp2.rds")
logRSS_raw_hot_tci <- log_rss(issf_tci7_20y_landsat_hot_exp2, x1 = x1_TCI, x2 =
x2_TCI, ci = "se", ci_level = 0.95)

issf_tci7_20y_landsat_cool_exp2 <-
readRDS(".../issf_tci7_20y_landsat_cool_exp2.rds")
logRSS_raw_cool_tci <- log_rss(issf_tci7_20y_landsat_cool_exp2, x1 = x1_TCI, x2
= x2_TCI, ci = "se", ci_level = 0.95)

# Add season labels and combine for plotting
df_hot_tci <- logRSS_raw_hot_tci$df %>% mutate(source = "hot", raw_seq_tci =
raw_seq_tci)
df_rainy_tci <- logRSS_raw_rainy_tci$df %>% mutate(source = "rainy",
raw_seq_tci = raw_seq_tci)
df_cool_tci <- logRSS_raw_cool_tci$df %>% mutate(source = "cool", raw_seq_tci =
raw_seq_tci)
df_3_tci <- bind_rows(df_hot_tci, df_rainy_tci, df_cool_tci)

# Plot RSS against TCI for all seasons with confidence intervals and legend
tci_3_seasons_ggplot <- ggplot(df_3_tci, aes(x = raw_seq_tci)) +
  geom_ribbon(aes(ymin = exp(lwr), ymax = exp(upr), fill = source), alpha =
0.2, color = NA) +
  geom_line(aes(y = exp(log_rss), color = source)) +
  geom_line(aes(y = exp(lwr), color = source), linetype = "dashed") +
  geom_line(aes(y = exp(upr), color = source), linetype = "dashed") +
  geom_hline(yintercept = 1, linetype = "dashed") +
  scale_color_manual(values = c("hot" = "red", "rainy" = "blue", "cool" =
"green"),
                     labels = c("hot" = "hot dry season", "rainy" = "rainy
season", "cool" = "cool dry season")) +
  scale_fill_manual(values = c("hot" = "red", "rainy" = "blue", "cool" =
"green"),
                    labels = c("hot" = "hot dry season", "rainy" = "rainy
season", "cool" = "cool dry season")) +
  labs(x = "TCI", y = "RSS", color = "Legend", fill = "Legend") +
  theme_classic()

# Optional save figure code commented out
# ggsave(".../RSS_tci_unscaled.png", plot = tci_3_seasons_ggplot, width = 8,
height = 6, dpi = 300)



########### MSAVI ##############

# Repeat the process for MSAVI (vegetation index)

# Retrieve center and scale attributes for MSAVI_landsat_scaled
center_msavi <- attr(cov_scaled$MSAVI_landsat_scaled, "scaled:center")
scale_msavi <- attr(cov_scaled$MSAVI_landsat_scaled, "scaled:scale")
```

```
# Generate unscaled MSAVI sequence for plotting
raw_seq_msavi <- seq(quantile(cov_scaled$MSAVI, 0.01, na.rm=TRUE),
                     quantile(cov_scaled$MSAVI, 0.99, na.rm=TRUE),
length.out=100)

# Standardize MSAVI sequence
scaled_seq_msavi <- (raw_seq_msavi - center_msavi) / scale_msavi

# Define data frames for varying MSAVI and reference conditions
x1_MSAVI <- data.frame(proba_human_scaled = 0, MSAVI_landsat_scaled =
scaled_seq_msavi,
                       `I(MSAVI_landsat_scaled^2)` = scaled_seq_msavi^2,
TCI7_scaled = 0,
                       `I(TCI7_scaled^2)` = 0, fire_20y_scaled = 0,
shrub_density_scaled = 0,
                       `I(shrub_density_scaled^2)` = 0, sl_ = 550, log_sl_ =
log(550), cos_ta_ = 0)

x2_MSAVI <- data.frame(proba_human_scaled = 0, MSAVI_landsat_scaled = 0,
                       `I(MSAVI_landsat_scaled^2)` = 0, TCI7_scaled = 0,
`I(TCI7_scaled^2)` = 0,
                       fire_20y_scaled = 0, shrub_density_scaled = 0,
                       `I(shrub_density_scaled^2)` = 0, sl_ = 550, log_sl_ =
log(550), cos_ta_ = 0)

# Load models and compute logRSS with confidence intervals for all seasons
issf_tci7_20y_landsat_rainy_exp2 <-
readRDS(".../issf_tci7_20y_landsat_rainy_exp2.rds")
logRSS_raw_rainy_msavi <- log_rss(issf_tci7_20y_landsat_rainy_exp2, x1 =
x1_MSAVI, x2 = x2_MSAVI, ci = "se", ci_level = 0.95)

issf_tci7_20y_landsat_hot_exp2 <-
readRDS(".../issf_tci7_20y_landsat_hot_exp2.rds")
logRSS_raw_hot_msavi <- log_rss(issf_tci7_20y_landsat_hot_exp2, x1 = x1_MSAVI,
x2 = x2_MSAVI, ci = "se", ci_level = 0.95)

issf_tci7_20y_landsat_cool_exp2 <-
readRDS(".../issf_tci7_20y_landsat_cool_exp2.rds")
logRSS_raw_cool_msavi <- log_rss(issf_tci7_20y_landsat_cool_exp2, x1 =
x1_MSAVI, x2 = x2_MSAVI, ci = "se", ci_level = 0.95)

# Add season labels and combine for plotting
df_hot_msavi <- logRSS_raw_hot_msavi$df %>% mutate(source = "hot",
raw_seq_msavi = raw_seq_msavi)
df_rainy_msavi <- logRSS_raw_rainy_msavi$df %>% mutate(source = "rainy",
raw_seq_msavi = raw_seq_msavi)
df_cool_msavi <- logRSS_raw_cool_msavi$df %>% mutate(source = "cool",
raw_seq_msavi = raw_seq_msavi)
df_3_msavi <- bind_rows(df_hot_msavi, df_rainy_msavi, df_cool_msavi)

# Plot RSS vs MSAVI across seasons with ribbons and lines
MSAVI_3_seasons_ggplot <- ggplot(df_3_msavi, aes(x = raw_seq_msavi)) +
  geom_ribbon(aes(ymin = lwr, ymax = upr, fill = source), alpha = 0.2, color =
NA) +
  geom_line(aes(y = log_rss, color = source)) +
  geom_line(aes(y = lwr, color = source), linetype = "dashed") +
  geom_line(aes(y = upr, color = source), linetype = "dashed") +
```

```
  geom_hline(yintercept = 0, linetype = "dashed") +
  scale_color_manual(values = c("hot" = "red", "rainy" = "blue", "cool" =
"green"),
                     labels = c("hot" = "hot dry season", "rainy" = "rainy
season", "cool" = "cool dry season")) +
  scale_fill_manual(values = c("hot" = "red", "rainy" = "blue", "cool" =
"green"),
                    labels = c("hot" = "hot dry season", "rainy" = "rainy
season", "cool" = "cool dry season")) +
  labs(x = "MSAVI", y = "RSS", color = "Legend", fill = "Legend") +
  theme_classic()

# Optional save figure code commented out
# ggsave(".../RSS_MSAVI_unscaled.png", plot = MSAVI_3_seasons_ggplot, width =
8, height = 6, dpi = 300)

################## Fires ##################

# We retrieve the center and scale attributes used to standardize the
'fire_20y_scaled' covariate
center_fire <- attr(cov_scaled$fire_20y_scaled, "scaled:center")
scale_fire  <- attr(cov_scaled$fire_20y_scaled, "scaled:scale")

# We create a sequence of unscaled 'fire_20y' values between the 1st and 99th
percentiles
raw_seq_fire <- seq(quantile(cov_scaled$fire_20y, probs = 0.01, na.rm = TRUE),
                    quantile(cov_scaled$fire_20y, probs = 0.99, na.rm = TRUE),
                    length.out = 100)

# We standardize this sequence using the original center and scale to get
scaled values
scaled_seq_fire <- (raw_seq_fire - center_fire) / scale_fire

# We build a data frame x1_fire varying the fire covariate over the scaled
sequence while keeping others fixed at reference values
x1_fire <- data.frame(
  proba_human_scaled = 0, TCI7_scaled = 0, `I(TCI7_scaled^2)` = 0,
  fire_20y_scaled = scaled_seq_fire, MSAVI_landsat_scaled = 0,
  `I(MSAVI_landsat_scaled^2)` = 0, shrub_density_scaled = 0,
  `I(shrub_density_scaled^2)` = 0, sl_ = 550, log_sl_ = log(550), cos_ta_ = 0
)

# We build a reference data frame x2_fire with all covariates fixed at
reference values (including fire at zero)
x2_fire <- data.frame(
  proba_human_scaled = 0, MSAVI_landsat_scaled = 0, `I(MSAVI_landsat_scaled^2)`
= 0,
  TCI7_scaled = 0, `I(TCI7_scaled^2)` = 0, fire_20y_scaled = 0,
shrub_density_scaled = 0,
  `I(shrub_density_scaled^2)` = 0, sl_ = 550, log_sl_ = log(550), cos_ta_ = 0
)



########## Rainy Season

# We load the fitted ISSF model for the rainy season
```

```
issf_tci7_20y_landsat_rainy_exp2 <-
readRDS("C:/Users/court/Documents/ISSF/issf_tci7_20y_landsat_rainy_exp2.rds")

# We compute the log relative selection strength (logRSS) and confidence
intervals comparing x1_fire vs x2_fire
logRSS_raw_rainy_fire <- log_rss(issf_tci7_20y_landsat_rainy_exp2, x1 =
x1_fire,
  x2 = x2_fire, ci = "se", ci_level = 0.95)



########## Hot Season

# We load the fitted ISSF model for the hot dry season
issf_tci7_20y_landsat_hot_exp2 <-
readRDS("C:/Users/court/Documents/ISSF/issf_tci7_20y_landsat_hot_exp2.rds")

# We compute logRSS and confidence intervals for the hot season similarly
logRSS_raw_hot_fire <- log_rss(issf_tci7_20y_landsat_hot_exp2, x1 = x1_fire,
  x2 = x2_fire, ci = "se", ci_level = 0.95)



########## Cool Season

# We load the fitted ISSF model for the cool dry season
issf_tci7_20y_landsat_cool_exp2 <-
readRDS("C:/Users/court/Documents/ISSF/issf_tci7_20y_landsat_cool_exp2.rds")

# We compute logRSS and confidence intervals for the cool season
logRSS_raw_cool_fire <- log_rss(issf_tci7_20y_landsat_cool_exp2, x1 = x1_fire,
  x2 = x2_fire, ci = "se", ci_level = 0.95)



########## Combine 3 seasons

# We add season labels and original (unscaled) fire counts to each season's
dataframe
df_hot_fire   <- logRSS_raw_hot_fire$df   %>% mutate(source = "hot",
raw_seq_fire = raw_seq_fire)
df_rainy_fire <- logRSS_raw_rainy_fire$df %>% mutate(source = "rainy",
raw_seq_fire = raw_seq_fire)
df_cool_fire  <- logRSS_raw_cool_fire$df  %>% mutate(source = "cool",
raw_seq_fire = raw_seq_fire)

# We combine all three season datasets into one for plotting
df_3_fire <- bind_rows(df_hot_fire, df_rainy_fire, df_cool_fire)

# We plot RSS against the number of fires with confidence ribbons and lines,
colored by season
fire_3_seasons_ggplot <- ggplot(df_3_fire, aes(x = raw_seq_fire)) +
  geom_ribbon(aes(ymin = exp(lwr), ymax = exp(upr), fill = source),
              alpha = 0.2, color = NA) +
  geom_line(aes(y = exp(log_rss), color = source)) +
  geom_line(aes(y = exp(lwr), color = source), linetype = "dashed") +
  geom_line(aes(y = exp(upr), color = source), linetype = "dashed") +
  geom_hline(yintercept = 1, linetype = "dashed") +  # baseline RSS=1 reference
  scale_color_manual(
```

```
    values = c("hot" = "red", "rainy" = "blue", "cool" = "green"),
    labels = c("hot" = "hot dry season", "rainy" = "rainy season", "cool" =
"cool dry season")
  ) +
  scale_fill_manual(
    values = c("hot" = "red", "rainy" = "blue", "cool" = "green"),
    labels = c("hot" = "hot dry season", "rainy" = "rainy season", "cool" =
"cool dry season")
  ) +
  labs(
    x = "number of fires for the period 2004-2023",
    y = "RSS",
    color = "Legend",
    fill  = "Legend"
  ) +
  theme_classic()

# Optional: We can save the plot with ggsave (path adapted to our system)
# ggsave("C:/Users/court/Documents/RSS/figures/RSS_fire_unscaled.png", plot =
fire_3_seasons_ggplot, width = 8, height = 6, dpi = 300)



############# Woody vegetation ##############

# We retrieve the center and scale attributes used to standardize
shrub_density_scaled
center_shrub <- attr(cov_scaled$shrub_density_scaled, "scaled:center")
scale_shrub <- attr(cov_scaled$shrub_density_scaled, "scaled:scale")

# We create a sequence of unscaled shrub density values between the 1st and
99th percentiles
raw_seq_shrub <- seq(quantile(cov_scaled$shrub_density, probs = 0.01, na.rm =
TRUE),
                     quantile(cov_scaled$shrub_density, probs = 0.99, na.rm =
TRUE),
                     length.out = 100)

# We standardize this sequence using the original center and scale
scaled_seq_shrub <- (raw_seq_shrub - center_shrub) / scale_shrub

# We build a data frame x1_shrub varying shrub_density_scaled over the
sequence, others fixed at references
x1_shrub <- data.frame(
  proba_human_scaled = 0, MSAVI_landsat_scaled = 0, `I(MSAVI_landsat_scaled^2)`
= 0,
  TCI7_scaled = 0, `I(TCI7_scaled^2)` = 0, fire_20y_scaled = 0,
  shrub_density_scaled = scaled_seq_shrub, `I(shrub_density_scaled^2)` =
scaled_seq_shrub^2,
  sl_ = 550, log_sl_ = log(550), cos_ta_ = 0
)

# We build a reference data frame x2_shrub with all covariates fixed at
reference values including shrub density at zero
x2_shrub <- data.frame(
  proba_human_scaled = 0, MSAVI_landsat_scaled = 0, `I(MSAVI_landsat_scaled^2)`
```

```
= 0,
  TCI7_scaled = 0, `I(TCI7_scaled^2)` = 0, fire_20y_scaled = 0,
  shrub_density_scaled = 0, `I(shrub_density_scaled^2)` = 0,
  sl_ = 550, log_sl_ = log(550), cos_ta_ = 0
)


########### RAINY

# We load the fitted ISSF model for rainy season
issf_tci7_20y_landsat_rainy_exp2 <-
readRDS("C:/Users/court/Documents/ISSF/issf_tci7_20y_landsat_rainy_exp2.rds")

# We compute logRSS and confidence intervals for shrub density variation
logRSS_raw_rainy_shrub <- log_rss(issf_tci7_20y_landsat_rainy_exp2, x1 =
x1_shrub,
  x2 = x2_shrub, ci = "se", ci_level = 0.95)

# We plot RSS against unstandardized shrub density with confidence ribbon for
rainy season
shrub_rainy <- ggplot(logRSS_raw_rainy_shrub$df, aes(x = raw_seq_shrub, y =
exp(log_rss), ymin = exp(lwr), ymax = exp(upr))) +
  geom_ribbon(fill = "grey80", alpha = .5) +
  geom_line(color = "steelblue", linewidth = 1) +
  geom_hline(yintercept = 1, linetype = "dashed") +
  labs(
    x = "perennial vegetation density",
    y = "RSS"
  ) +
  theme_bw()


########### HOT

# We load the fitted ISSF model for hot dry season
issf_tci7_20y_landsat_hot_exp2 <-
readRDS("C:/Users/court/Documents/ISSF/issf_tci7_20y_landsat_hot_exp2.rds")

# We compute logRSS and confidence intervals for shrub density variation in hot
season
logRSS_raw_hot_shrub <- log_rss(issf_tci7_20y_landsat_hot_exp2, x1 = x1_shrub,
  x2 = x2_shrub, ci = "se", ci_level = 0.95)

# We plot RSS against unstandardized shrub density for hot season
shrub_hot <- ggplot(logRSS_raw_hot_shrub$df, aes(x = raw_seq_shrub, y =
exp(log_rss), ymin = exp(lwr), ymax = exp(upr))) +
  geom_ribbon(fill = "grey80", alpha = .5) +
  geom_line(color = "steelblue", linewidth = 1) +
  geom_hline(yintercept = 1, linetype = "dashed") +
  labs(
    x = "perennial vegetation density",
    y = "RSS"
  ) +
  theme_bw()
```

```
########### COOL

# We load the fitted ISSF model for cool dry season
issf_tci7_20y_landsat_cool_exp2 <-
readRDS("C:/Users/court/Documents/ISSF/issf_tci7_20y_landsat_cool_exp2.rds")

# We compute logRSS and confidence intervals for shrub density variation in
cool season
logRSS_raw_cool_shrub <- log_rss(issf_tci7_20y_landsat_cool_exp2, x1 =
x1_shrub,
  x2 = x2_shrub, ci = "se", ci_level = 0.95)

# We plot RSS against unstandardized shrub density for cool season
shrub_cool <- ggplot(logRSS_raw_cool_shrub$df, aes(x = raw_seq_shrub, y =
exp(log_rss), ymin = exp(lwr), ymax = exp(upr))) +
  geom_ribbon(fill = "grey80", alpha = .5) +
  geom_line(color = "steelblue", linewidth = 1) +
  geom_hline(yintercept = 1, linetype = "dashed") +
  labs(
    x = "perennial vegetation density",
    y = "RSS"
  ) +
  theme_bw()


########## Combine 3 seasons

# We add season labels and original shrub density values to each dataframe
df_hot_shrub <- logRSS_raw_hot_shrub$df   %>% mutate(source = "hot",
raw_seq_shrub = raw_seq_shrub)
df_rainy_shrub <- logRSS_raw_rainy_shrub$df %>% mutate(source = "rainy",
raw_seq_shrub = raw_seq_shrub)
df_cool_shrub <- logRSS_raw_cool_shrub$df %>% mutate(source = "cool",
raw_seq_shrub = raw_seq_shrub)

# We combine all three seasons for shrub density for plotting
df_3_shrub <- bind_rows(df_hot_shrub, df_rainy_shrub, df_cool_shrub)

# We plot RSS vs shrub density for all seasons with ribbons and lines, colored
by season
shrub_3_seasons_ggplot <- ggplot(df_3_shrub, aes(x = raw_seq_shrub)) +
  geom_ribbon(aes(ymin = exp(lwr), ymax = exp(upr), fill = source),
              alpha = 0.2, color = NA) +
  geom_line(aes(y = exp(log_rss), color = source)) +
  geom_line(aes(y = exp(lwr), color = source), linetype = "dashed") +
  geom_line(aes(y = exp(upr), color = source), linetype = "dashed") +
  geom_hline(yintercept = 1, linetype = "dashed") +  # baseline reference RSS=1
  scale_color_manual(
    values = c("hot" = "red", "rainy" = "blue", "cool" = "green"),
    labels = c("hot" = "hot dry season", "rainy" = "rainy season", "cool" =
"cool dry season")
  ) +
  scale_fill_manual(
    values = c("hot" = "red", "rainy" = "blue", "cool" = "green"),
    labels = c("hot" = "hot dry season", "rainy" = "rainy season", "cool" =
"cool dry season")
```

```
  ) +
  labs(
    x = "indicator of woody vegetation density",
    y = "RSS",
    color = "Legend",
    fill  = "Legend"
  ) +
  theme_classic()

# Optional: save the shrub density plot
# ggsave("C:/Users/court/Documents/RSS/figures/RSS_shrub_unscaled.png", plot =
shrub_3_seasons_ggplot, width = 8, height = 6, dpi = 300)
```