Politecnico di Torino
Instituto Superior Técnico

Master of Science in Environmental and Land Engineering (Climate Change track)
A.a. 2023/2024
Graduation session: October 2024

Modelling the CO_2 sequestration potential of agricultural soils with a hybrid system

Supervisor:
Prof. Vincenzo Andrea Riggio

Co-supervisors:
Dr. Ricardo F.M. Teixeira
Prof. Tiago Domingos

Candidate:
Flavia Durelli S316910

Abstract

The increase of carbon dioxide (CO_2) concentration in the atmosphere and its consequences on the climate and the environment are widely discussed topics. Projections based on plausible emission scenarios have highlighted the need to take serious action against further increase in atmospheric greenhouse gas (GHG) concentration. The signing parties of the Paris Agreement of 2015 committed to keeping the temperature increase well below 2°C in 2100 with respect to pre-industrial conditions. It is unlikely that this goal will be met strictly with emissions reductions, which will require carbon offsetting, e.g. through soil carbon sequestration.

This thesis focuses on soil carbon sequestration in cultivated soils, which is one of the pathways to reducing the impacts of pastures and agricultural activities. The main goal of the thesis was to develop a model to quantify the dynamics of accumulation or depletion of Soil Organic Carbon (SOC) in pastures.

The model developed here had a hybrid configuration, combining a process-based relationship (theory-driven model) with machine learning (ML) techniques, more specifically using Artificial Neural Networks (ANN), for parameter calibration (data-driven model). The choice of a hybrid model was made to preserve the physical consistency and the interpretability of the process-based model, while integrating a large quantity of remote sensing (RS) data in complex patterns through the ML algorithm. This approach can overcome limitations of parameter rigidity in purely process-based models, as well as interpretability in purely data-based models. The data used to train the model were collected in 9 different farms (8 in Portugal and 1 in Spain). All plots surveyed in the farms had Sown Biodiverse Permanent Pastures Rich in Legumes (SBPPRL), a system known for its high potential for sequestering carbon in soils. The process-based model that was used as base for the study was a simple 0-dimension and 2-parameter equation describing the relationship between SOC content at a given instant and the value after a defined time interval. The two parameters of the model represent carbon inputs (from plants and livestock) and mineralization rate. Also, ANN were used to estimate the best values of such parameters. The research was focused on finding the model hyperparameters which allowed to obtain the best possible fit between measurements and modelled values.

This resulted in a model that described the dynamics of accumulation/depletion of SOC with good fitting accuracy (R^2 = 0.64 in the best configuration) compared to an alternative ordinary least squares (OLS) estimation of parameters (R^2 = 0.33). Moreover, it provided region-specific quantitative estimates of the parameters of the process-based model (carbon input K = 0.88 $kg_{SOM}/(100\ kg_{soil} \cdot y)$, mineralization rate α = 0.29 y^{-1}), and to assess the yearly potential for SOC accumulation in SBPPRL. The model can be used in the future to forecast SOC dynamics and carbon sequestration under several climatic scenarios, thus contributing to better inform strategies for CO_2 removal from the atmosphere for climate change mitigation. Future research should focus on improving the performance of the hybrid model through more extensive hyperparameter tuning, as well as trying the same approach with a more complex process-based model which explicitly models the different carbon pools present in soils.

Table of Contents

Table of figures

Table of tables

1. Introduction

Anthropic activity on Earth has caused multiple variations to the natural equilibriums of the planet, the most discussed of which is climate change. In the 6th Assessment Report of the Intergovernmental Panel on Climate Change (IPCC), it is stated with high confidence that "*human activities, principally through emissions of greenhouse gases, have unequivocally caused global warming"*, and that the emission of such GHG derived from *"different contributions, unequal in space and time, of activities such as unsustainable energy use, land use and other lifestyle and consumption patterns"* (IPCC, 2023). Due to the strong impact of GHG concentration increase on the environment and its side effects on many human activities, in the last decades many international organizations and governments have started working on strategies to either reduce the direct impacts related to GHG emissions, adapt society to climate change and its consequences, and to mitigate the present concentration levels. Currently, the most comprehensive of these agreements is the Paris Agreement of 2015, where 195 parties (194 countries and the European Union) committed to limit global warming in 2100 well below 2°C (and, possibly, below 1.5°C) with respect to pre-industrial levels.

In order to reach this goal, IPCC stated with high confidence that it is not sufficient to reduce current emissions, but it is absolutely necessary to achieve net zero emissions by 2050, by implementing Negative Emission Technologies (NET), including implementation of Carbon Capture and Storage (CCS) in energy production, transition to renewable energy resources, and some land related strategies, such as reforestation and reduced deforestation, agricultural land management and soil-related improvements (IPCC, 2023).

This thesis is focused on the last class of strategies, which is particularly relevant because land and land use are the basis of many anthropic activities, including food supply, energy production and a long list of ecosystem services, as well as potentially being both a source and a sink of GHGs: this is why IPCC dedicated a special report to the relationship between land use and climate change. In the report, there is a section specifically addressing sustainable land management, which is seen with very high confidence as an opportunity to "*prevent and reduce land degradation, maintain land productivity, and sometimes reverse the adverse impacts of climate change on land degradation"* (IPCC, 2020).

Land management is a fundamental issue because it is not equally addressed in all parts of the world. In developing countries, the strong population increase, and the lack of specific land-related regulations make the prospect of having high short-term productivity with unsustainable practices that increase soil degradation very profitable for companies. One the other hand, in Europe and other regions, soil is one of the mandatory carbon pools to be assessed to comply with the objectives of Kyoto protocol but, anyway, generally land-related strategies are privately managed, while a collective and global approach should be adopted (Lal et al., 2015).

Between the land-related strategies listed in the IPCC report, this thesis was mainly focused on SOC increase, whose impact on different aspects is described in this document: it is predicted to have a largely positive impact on all studied categories (mitigation, adaptation, desertification, land degradation and food security), though with different levels of confidence (IPCC, 2020). The numerous co-benefits of soil carbon sequestration require a more global approach, and a strong policy effort providing farmers with incentives to adopt more sustainable cultivation methods (Lal et al., 2015).

The main goal of this research is to create a reliable model to predict the evolution of SOC content in time, using the combination of some data from soil samples (such as sand, silt and clay content, pH, soil moisture and temperature) and radiometric data of reflectance in different bands from RS. The integration of data occurs by means of a hybrid model, where ML techniques (i.e., ANN) are applied to perform regression and obtain the parameters of a process-based model used to calculate SOC content after a determined time interval. The results of the process-based model with the calculated parameters are then compared to actual measurements in the considered samples, to assess the accuracy of the model.

This thesis is structured in seven main sections: section 1 is the introduction, with an overview of the rationale and goals of this thesis; section 2 has the description of the state of the art for SOM measurement, modelling and the role of hybrid models, and provides a theoretical framework for the work that has been carried out; in section 3, the objective of the research and its innovations with respect to the current state of the art are illustrated; section 4 shows the materials and methods that were used; in section 5, the results obtained through this procedure are illustrated and discussed; section 6 is dedicated to the analysis of the limitations observed in the approach and to suggestions of some potential future perspectives for improvement; section 7 briefly summarizes the conclusions that can be taken out of this research. The complete code used for the implementation of the model is presented in the Annex (section 9).

2. State of the art

2.1 The importance of carbon sequestration

SOC is defined as the carbon content of Soil Organic Matter (SOM), which is conventionally accounted with the van Bemmelen factor (SOM:SOC = 1.724, corresponding to a carbon content of 58% in organic matter), even though several studies challenge this assumption (Pribyl, 2010; Minasny et al., 2020), arguing that it would be a simplification that, in reality, can only be applied to a few types of soil, or to particular components of soil organic matter. More generally, the conversion factor of SOC in SOM should range between 1.4 and 2.5, corresponding to a percentage between 40% and 72% (Wiley, 1906), and a possibly more realistic value is suggested considering 50% of SOC in SOM (Pribyl, 2010).

Soil carbon represents a huge opportunity for carbon storage, since it constitutes one of the major C pools, together with oceanic, geologic and biotic, with a total of 2300 Pg of C storage potential at 1 m depth, of which 1550 Pg of SOC and 750 Pg of soil inorganic carbon (SIC) (Lal, 2003), which is around twice the amount found in the form of CO_2 in the atmosphere (Smith, 2012). Moreover, soil carbon brings further co-benefits other than reducing atmospheric concentrations, including the protection or increase of soil fertility, the maintenance of resilience to climate change, the reduction of soil erosion and habitat conversion (Bossio et al., 2020). This makes it not only an effective way to reduce atmospheric GHG concentrations, but also a strategy to reduce further emissions from agriculture due to increased productivity from higher soil quality. This is why both the conservation of existing soil carbon pools, and the restoration of depleted ones are fundamental to achieve the goals stated by international climate agreements (Bossio et al., 2020).

2.1.1 The carbon cycle

The main processes related to carbon cycle in soils are accumulation and mineralization. Accumulation is related to organic matter decomposition from macro- to micro-aggregates through rhizodeposition, earthworms' activity and root litter transformation, which incorporate carbon into the soil matrix allowing a good stabilization degree (Soussana et al., 2010). On the other hand, mineralization (also known as soil respiration) is the conversion of carbonaceous material to CO_2, which is related to erosion processes and soil microbial activities and could transform the stabilised carbon into a further GHG source. These processes are schematized in *Figure 1*.

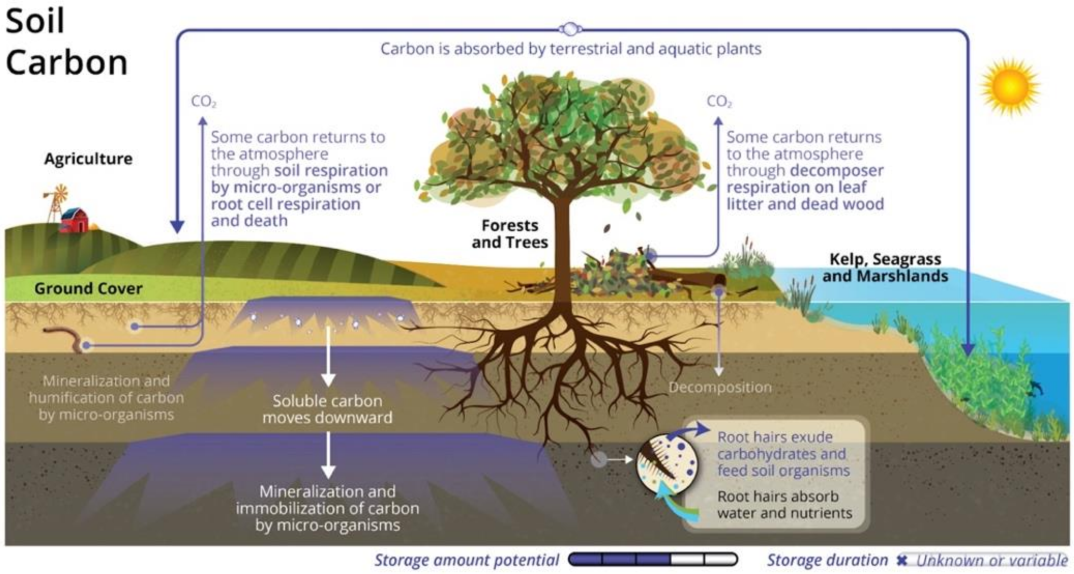
Soil Carbon
Carbon is absorbed by terrestrial and aquatic plants
CO₂
Some carbon returns to the atmosphere through soil respiration by micro-organisms or root cell respiration and death
CO₂
Some carbon returns to the atmosphere through decomposer respiration on leaf litter and dead wood
Agriculture
Forests and Trees
Kelp, Seagrass and Marshlands
Ground Cover
Mineralization and humification of carbon by micro-organisms
Soluble carbon moves downward
Decomposition
Root hairs exude carbohydrates and feed soil organisms
Root hairs absorb water and nutrients
Mineralization and immobilization of carbon by micro-organisms
Storage amount potential
Storage duration ✖ Unknown or variable

Figure 1 - Illustration of the soil carbon cycle, specifically observing the mechanisms of accumulation and mineralization. Adapted from Biotoken website (2023).

2.1.2 Carbon sequestration in soil

Carbon sequestration in soil can be defined as the "*process of transferring CO_2 from the atmosphere into the soil of a land unit through unit plants, plant residues and other organic solids, which are stored and retained in the unit as part of the soil organic matter*" (Lal et al., 2015).

The study conducted by Bossio et al. (2020), reports that soil carbon sequestration could potentially represent the 25% of Natural Climate Solutions (NCS), which is evaluated to be around 23.8 Gt_{CO2eq}/y in total. The computation was performed taking into account the land use constraints related to food security and biodiversity conservation. The relevance of carbon sequestration to NCS compared to the full environmental potential is variable depending on the type of ecosystem considered: it includes only the 9% of mitigation potential of forests (in which the highest fraction of carbon is sequestrated in the form of lignin of harvestable timber and other woody products), but has much higher significance in wetlands (72% of total mitigation potential) and in agriculture and grasslands, where it accounts for 47% of the total mitigation potential (Bossio et al., 2020). This potential can be achieved through conservation and restoration of soil carbon (avoided conversion of forested ecosystems to commercial agricultural uses, reforestation, protection and restoration of peatlands) and improved land management through the adoption of Recommended Management Practices (RMP), which include conservation agriculture, agroforestry, optimal grazing intensity and sowing of leguminous crops (Lal, 2007; Bossio et al., 2020).

On the other hand, other studies (Garnett et al., 2017) argue that the actual potential of soil carbon sequestration is not as relevant (ranging between 0.3 and 0.8 Gt_{CO2eq}/y) and that the computation of the sequestration potential is more commonly performed in local scales, while it is not possible to get a global estimate. However, as previously mentioned, it is also important to remember that land related solutions have multiple connections to land degradation, land productivity and mitigation purposes, which could make carbon sequestration in soil a valuable strategy in any case. There are several co-benefits associated to improved soil carbon content: the higher concentration of soil organic matter increase the productivity of soil and higher nutrient retention, which in agricultural fields represents better opportunities to achieve food security while reducing the need to use external nutrient inputs by fertilizers, improves soil erosion control and water retention capacity (leading to lower surface runoff and additional protection against its consequences), as well as off-setting anthropogenic emissions and reducing the net increase in concentration of atmospheric CO_2 (Lal et al., 2015).

The residence time of the absorbed carbon could be variable between a few instants (immediate remission) and a long-term storage up to millennia, which is why the development of methods for carbon sequestration in soil should not only focus on the quantity of carbon that is captured but on the storage stability as well, to ensure the removal of the GHG from the atmosphere for a significant time period. The stabilization of SOC in aggregates can occur through chemical (formation of soil carbonates) or biological (occlusion through formation of stable micro-aggregates and non-hydrolysable compounds) protection mechanisms, but these are very unstable conditions, which make the creation of SOC pools highly reversible (Lal et al., 2015). The main issue is the reactivity of SOC, which makes it extremely dynamic and vulnerable to land use and climate changes, and only with effective control of losses it is possible to reach a new equilibrium condition after a change in the environmental setting.

In general, land use change from grassland to cropland causes very fast losses of SOC (18% in temperate regions with dry climate and up to 29% in moist climate), due to lower return of biomass carbon, higher losses by erosion and mineralization, and stronger variations in soil temperature and moisture, and can take up to 20 years to be restored (Soussana et al., 2010). Moreover, agricultural practices such as ploughing tend to release a part of the already stocked SOC. The restoration of SOC pools occurs through specific management practices, including conservation agriculture, precision farming, integrated nutrient management and micro-irrigation (Soussana et al., 2010). The implementation of improved management systems (including fertilization and grazing intensity) has a positive impact in the ability of soil to sequestrate carbon. However, the evaluation of the impacts of change of agricultural practices still holds very high uncertainty (Morais et al. 2019).

Grasslands have higher sequestration potential than croplands, but the transformation of croplands into grasslands should primarily be done in depleted and abandoned fields, due to its impact in food security, because spontaneous grasslands have average lower food production (Lal et al., 2015). Applying specific management practices, including fertilization, improved grazing, sowing of legumes and grasses and irrigation, it is possible to obtain good SOC accumulation. (Lal et al., 2015).

As previously mentioned, climate change and its consequences are other factors which are expected to strongly affect C stocks in soil: even though the rise in atmospheric CO_2 decreases grassland sensitivity to drought and increases plant productivity, the negative impacts of increased temperature and reduced rainfall causing more frequent droughts are expected to turn temperate grasslands in carbon sources rather than sinks (Smith et al., 2008). Moreover, climate change is projected to have effects on plant distribution, with negative impacts on biodiversity at regional and global scale (Smith et al., 2008).

2.2 Soil carbon assessment

2.2.1 Direct or indirect measurement of soil carbon

One of the main issues in the implementation of carbon sequestration in soil strategies is the assessment of the storage potential of the soil under study and its possible evolutions. This is fundamental to understand and, later on, verify how much carbon could be stored in a site, how stable would the storage be, and which are the advised management practices to achieve the best results, which can be region-specific and depend on socio-economic context such as level of technological penetration of innovations in farming. Regardless of the model type that is used, all require field-level data, and therefore it is important to understand how data is collected in farms.

The first thing to do when trying to quantify the SOC stock potential in a site is to directly measure the baseline SOC values, which is not an easy task and is usually very expensive. The internationally accepted operational definition of SOC is "*the organic carbon present in the fraction of the soil that passes through the 2 mm sieve*" (FAO, 2019). The physical evaluation of its baseline value requires the quantification of fine earth (< 2mm) and coarse mineral fraction (> 2mm) in soil, the SOC content of the fine earth, and the soil bulk density.

An accurate procedure to achieve these results is the dry combustion method, which consists in air-drying the samples and pass them through a 2 mm sieve, to separate the fine earth from the coarse fraction; then, the dried soil is combusted in an elemental analyzer at high temperature in an atmosphere of pure of oxygen. This way, the carbon present in soil is converted into CO_2, which is then measured through a autoanalyzer. This method, however, only allows to compute the total carbon content of soil, so to evaluate SOC it is necessary to separately evaluate the fraction of SIC and then subtract it from the obtained result (FAO, 2019a). To obtain the actual dry matter weight of the fine earth, the weight of the residual water content needs to be subtracted from the < 2 mm fraction (FAO, 2019).

Direct measurement of SOC brings high sources of uncertainty (high spatial and temporal heterogeneity, changes related to sampling depth and the number of cores to extract to have a statistically meaningful sample), which may compromise the detection of stock changes and the identification of the most important factors responsible of such change (FAO, 2019). One of the major sources of uncertainty that comes up in the evaluation of SOC is the large spatial variability and the need to create site-specific models (Goidts et al., 2009). Further sources of uncertainty may arise from errors in the sampling and analytical procedures, including sampling depth, proper mixing of the composite samples and different climate conditions between natural and laboratory setting (FAO, 2019).

Because of cost and representativeness of samples, there have been proposals for ensuring good sampling of SOC. For example, the study conducted by Goidts et al. (2009) developed a sampling method which considers the SOC evaluation at different scales: the largest scale (regional scale) was subdivided into landscape units, which are based on the agricultural land use, the agricultural region and the soil type (LSU scale). Each LSU is divided into soil profiles (field scale), in each of which a composite sample should be collected, which includes five subsamples taken within a circle of 4 m radius from the centre of the soil profile (microsite scale and sample scale). Moreover, in each layer sampled an intact core should be extracted to measure the soil bulk density (Goidts et al., 2009). The description of this method can give an idea of the number of samples to extract to reduce sources of error in SOC estimation. Moreover, it is necessary to consider that SOC stock varies with depth, which means that deep soil sampling would be recommended. The collection of a sufficient number at a sufficient depth for a large-scale model would increase the cost of the investigation to the point of not being economically feasible. Therefore, finding alternative methods to direct measurement, such as the combination of direct sampling and modelling, would be a good strategy to achieve higher cost-effectiveness.

Another strategy for the evaluation of SOC stocks could be the indirect assessment by measuring the balance of its fluxes (net C storage) at system boundaries (Soussana et al., 2010). This technique provides a higher temporal resolution, as changes can be detected with a time span of one year, instead of direct measurements which take several years or even decades to be assessed, but it needs the measurement of many C fluxes (including exchange with atmosphere, organic C import/export, dissolution in water and erosion) to compute the mass balance. The fluxes are variable according to soil and climate conditions, and even though some of them can occasionally be neglected, it is not always simple to assess their value. Some of the factors influencing NCS would be grassland type, N fertilizer supply, drainage, burning and climatic variables including rainfall, temperature and radiation (Soussana et al., 2010). Another important influence on the carbon sequestration potential is represented by the process of evapotranspiration, which affects plant growth, biomass production and microbial activity in soil and, therefore, when it increases, the Net Ecosystem CO_2 Exchange (NEE) is increased as well (Zhang et al., 2022).

Furthermore, to quantify the possibility for carbon stock through flux balance, it is fundamental to consider all fluxes from all the main GHGs (i.e., N_2O and CH_4 other than CO_2), and to apply an integrated method to transform the effects of each gas in CO_2 equivalents through their Global Warming Potential (GWP). This is crucial because modifying grassland management systems to improve carbon sequestration could locally cause an increase in methane and nitrous oxide concentrations. For instance, N_2O is an intermediate product of both nitrification and denitrification processes in soil and is therefore emitted by both dynamics. The main regulators of these processes regard temperature, pH, soil moisture, C availability, and nitrogen availability, which means that the application of N-based fertilizers increases the rate of reaction and will stimulate the emission of the gas. N_2O emissions undergo significant temporal and spatial variations and, for this reason, are not easy to quantify (Soussana et al., 2010). CH_4 emissions from soil are more relevant in wet environment than in dry ones, because it is mainly formed in anaerobic conditions (Soussana et al., 2010), so it needs to be carefully monitored in the management of wetlands, where methane could potentially be released.

2.2.2 Spectral methods of assessing soil carbon

Both direct and indirect SOC measurement present, among others, the issue of it being too site-specific to extensively assess the possibility for carbon sequestration, as soil and environmental conditions are strongly variable both in space and time, and through these methods alone it is not possible to reach a reliable estimate or to make predictions on how the situation could evolve in the future. This is why upcoming technology is studying the combination of direct measurements with spectral methods that can survey large areas with minimum cost and acceptable spatial resolution. RS techniques, using either satellite, airborne platforms and Unmanned Aerial Vehicles (UAVs), are defined as rapid, cost-effective and non-destructive methods to assess soil properties, including SOC (Angelopoulou et al., 2019).

Spectral analysis is based on the observation of reflectance of light on soil in the near- or mid-infrared region (NIR/MIR) which, combined with previous knowledge collected in spectral libraries, can be interpreted and reconducted to soil-content specific characteristics. This method can be used in different RS application for the estimation of soil properties, and it is particularly promising because it shows no need to extract soil samples with complicated and expensive sampling campaigns, it allows to get data for large geographical extensions and for areas that may be inaccessible from land, and it is useful to assess several soil properties in a concise way.

However, data collection through RS has a few main downsides, including low spectral resolution, geometric and atmospheric distortions and low penetration depth. Moreover, when data are collected from satellites, meteorological conditions could affect soil visibility and, therefore, the possibility to acquire data (Angelopoulou et al., 2019). This is why the repetition of soil surveys at a national or subnational scale is a good strategy to provide trends on the evolution of SOC stocks overtime. However, this strategy is not effectively implemented in most countries and, since the first measurements were not performed with the intention of subsequent monitoring, it is not always possible to reconstruct the impacts of land use or climate change on SOC pools in the site (Smith et al., 2019).

There are many different studies on the RS of soil chemical and physical properties. For instance, in the review performed by Ge et al. (2011), several papers were analysed, and each is focused on different aspects of soil composition, such as the content of different chemical elements (e.g., phosphorus, potassium, calcium), the percentage of clay, silt and sand, and other physical properties such as the electrical conductivity (Ge et al., 2011). Moreover, the research performed by Padarian et al. (2022), highlights the possibility of using RS data to monitor land cover and land use changes, which are crucial for the estimation of SOC stocks (Padarian et al., 2022). These studies are particularly important in the development of precision agriculture, which aims at optimizing the efficiency of cultivation by carefully planning the amount of water and other inputs provided to the crops. Also, satellite observations can be used specifically for the estimation of

SOC (Morais et al., 2023). In this study, data were collected using Google Earth Engine (GEE), which is a cloud-based platform containing a large amount of geospatial NIR data (including the whole Sentinel2 database), and allows to track changes over time, map trends and quantify differences. These data provide explanatory variables to estimate SOC content exploiting the combination with ML techniques. The cited study is particularly relevant for this thesis, as it will be based on the same principle and data sources.

2.2.3 Soil process-based modelling

Process-based soil models (PBM) were developed to satisfy the need to describe physical, chemical and biological interactions in the system at different scales. This is necessary because of the complex relationship between the many components of soils, which affect the multiple ecosystem services it provides, including agricultural productivity, and to address the challenges to direct measurement that are created by global change. PBM are considered an appropriate way to approach management and decision-making related to environmental issues, and they should be developed following a balance between too little and too much detail (Cuddington et al., 2013).

Initially, soil models focused on physical (water and solute movement, heat flows and energy balance) and chemical (sorption models, interaction between phases in contaminated environments) processes, disregarding the biological component (microbial activity), but nowadays the improved knowledge on interactions between microorganisms and the higher possibilities to acquire data make it possible to create soil models at the level of pore scale and even smaller (Vereecken et al., 2016). These models can give an overall view on soil-related processes (formation, water and nutrient cycling, biological activity, salinization, erosion and compaction), the related ecosystem services (climate regulation, buffering and filtering, food, fibre and energy provision) and how are they impacted by external drivers.

As previously introduced, PBM give a good description of real conditions taking into consideration many factors and natural processes, so they can explain in a clear way complex mechanisms (such as the physical, chemical and biological interactions occurring in soils), making the results easily interpretable. Moreover, a very important feature of PBM is the possibility to insert projections of the required data to make predictions, which makes them fundamental for decision making tasks. However, these models are not able to fully capture the complexity of interactions in nature but need to be based on simplifying assumptions. In addition, usually PBM are not able to recreate the highly non-linear relationships that are observed in nature and require huge computational capacity to obtain results. (Reichstein et al., 2019)

Smith et al. (1997) compared nine different models and their ability to predict the evolution of SOM in a long-term interval, using different datasets. Between the models, it was observed the distinction between two groups, one performing simulation significantly better than the other. The main reasons for this difference in performance was identified to be the site-specific calibration used in the higher-performance group (including RothC, CANDY, DNDC, CENTURY, DAISY and NCSOIL). Moreover, the models in this group had sufficiently high performance in different types of land use, not only for the one they were developed for, while the other group (SOMM, ITE and Verberne) gave satisfactory results only in their own specific application (Smith et al., 1997). Another factor influencing the performance of soil models is the need to combine them with other sub-models, including the coupling with soil water and nitrogen and plant growth. Even models that perform well in the prediction of SOM values, if coupled with more complex interactions show problems.

Between these observed models, a particularly relevant case is the Rothamsted Carbon Model, or RothC (Coleman & Jenkinson, 1996b), which is a model that has been widely used since the review by Smith et al. (1997). The algorithm was developed to describe and simulate the turnover of organic carbon in non-waterlogged topsoil, which is influenced by the effects of soil type, temperature, moisture and plant cover. It was also applied with positive results to grassland and woodland and under different types of climates. RothC needs a few input values, including initial carbon input (from plant and animal source), average monthly characteristics of the atmospheric conditions in the area under study (rainfall, open pan evaporation, mean air temperature) and some site-specific parameters (clay content, soil cover, depth, monthly external inputs). The only parameter related to the land use type is the ratio between decomposable and resistant plant material (DPM/RPM), and it is the key factor to determine soil respiration (Coleman & Jenkinson, 1996b).

The model computes the quantity of SOC as the sum of five main components (carbon pools): decomposable plant material (DPM), resistant plant material (RPM), microbial biomass (BIO), humified organic matter (HUM) and inert organic matter (IOM). In these pools, carbon gets mineralized at different mineralization rates, which describe the dynamics of decay of each carbon pool. The only exception to this is the fraction of IOM, which is highly stable organic matter, and is resistant to decomposition, so it does not get a mineralization rate (Fasma et al., 2021). The scheme of the decomposition in these five components is showed in *Figure 2*.

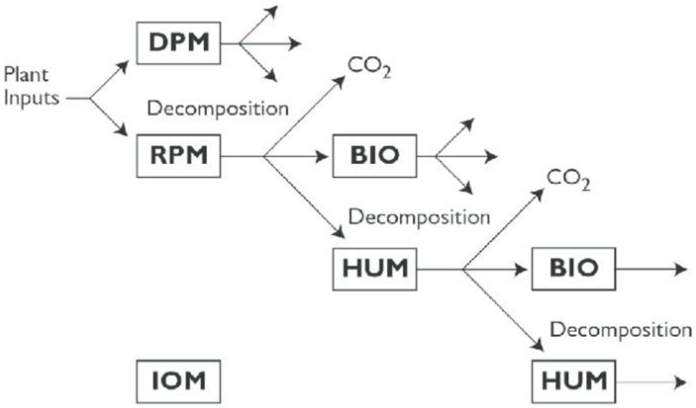

Figure 2 - Schematized representation of Soil Organic Carbon fragmentation in the model RothC. It is possible to observe that there is one non-decomposable pool (IOM), while the other fractions of SOC (decomposable plant material – DPM, resistant plant material – RPM, microbial biomass – BIO, humified organic matter – HUM) are decomposed at different rates. Decomposition is indicated by arrows in the figure (Skjemstad et al., 2004)

RothC has been used in many different applications both in its direct and inverse configuration. The direct configuration is used when the initial carbon input is known and gives as output the projection of SOM after a given time interval. For instance, the direct configuration was used to assess the proportion of pasture production as carbon input for SOC accumulation in different pasture types and management, to assess the best conditions to improve carbon fixation in soil (Liu et al., 2011).

The inverse configuration is used to evaluate the characteristics of the site under study without any direct sampling. For example, the inverse method was used to compute the optimal value of some parameters related to sown rainfed grasslands (root to shoot ratio, livestock unit, animal intake and DPM/RPM), to find the ideal conditions to optimize management practices with the aim to increase SOC content (Morais et al., 2018).

2.2.4 Hybrid models in Earth system science

Until recently, the only strategies used for geospatial assessments were the use of direct measurements and PBM, but advances in technology, specifically in artificial intelligence (AI) and ML techniques, open possibilities to integrate PBM and have a more powerful management of the large databases which characterise environmental studies through hybrid models (Willard et al., 2022). In the field of SOC improvement for CO_2 sequestration, the use of hybrid models could increase simulation accuracy at different spatial and temporal scales.

This type of modelling has been expanding in the last few years and applied to an increasing number of Earth system science (ESS) problems, especially when geospatial data is involved, in which both spatial and temporal context play a fundamental role. The structure of hybrid model combines a description of the physical processes (theory-driven model) and a deep learning algorithm (e.g., ANN, data-driven model) to describe spatial conditions in different circumstances and in different temporal instances, and even make predictions on possible future conditions. This is useful to merge the huge amount of geospatial data that is available and the possibility to process and interpret it in useful ways, while respecting the physical laws that describe the studied dynamics. In this study, a hybrid approach combining a theory-driven model and ANN will be applied.

ANN are the core of deep learning (DL), which is a branch of ML that aims to solve complex tasks. They were first introduced in the 1940s and experienced a swinging interest of the scientific community and, therefore, an alternate scheme of funding/non-funding. Recently, a new wave of interest in ANN has been developing, which is expected to be more permanent than the previous ones (Géron, 2019), and to lead to huge impact on technological progress and human lives because of the following reasons:

- The high performance of ANN with big amounts of data, which we have today.
- The increase of computing power (hardware advancement) in the last couple decades and the possibility for everyone to access databases and to store data in clouds.
- The improvement of training algorithms.

- The discovery that some of the limitations of ANN are not actually as strong as it was thought: for instance, they were believed to get stuck in local minima but the occurrence of this condition is not that frequent and, even when it happens, it still gives a good approximation of the absolute minimum.

ANN are a tool to design intelligent machines, modelled on the biological structure of human nervous system. Much like biological neurons, artificial neurons have a quite simple structure by themselves, and their strength is mainly represented by the big net of connections that they create between one another, through which they transmit the signals that make them able to function. From biological studies, it seems that neurons are organized in consecutive layers, and that is the same structure that is used to develop ANN (input layer – hidden layers – output layer). The basic structure of an ANN is schematized in *Figure 3*.

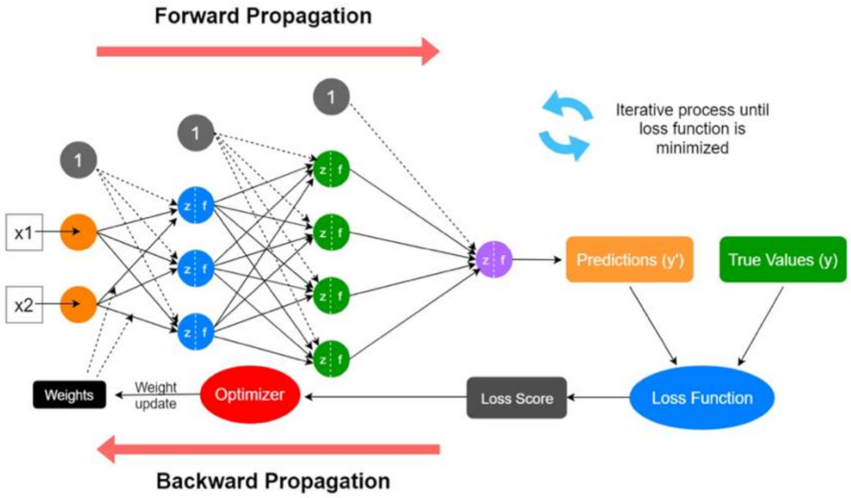

Figure 3 - Basic scheme of an ANN and its functioning process. More specifically, the scheme represents a network with two neurons in the input layer, two hidden layers (one containing three neurons and one with four neurons) and an output layer containing the results. (Pramoditha, 2022)

The simplest ANN feature is the perceptron, which is based on the threshold logic unit (TLU): inputs and outputs are numbers which are connected to each other through weighted connections. The output is computed through the weighted sum of inputs and a bias, which represents the minimum value for which the result is meaningful, and the application of an activation function to normalize the results (*Figure 3*). If the result of the step function exceeds a given threshold, the corresponding output is considered as the response to the task of the TLU. A single layer of TLUs creates a perceptron. If each TLU is connected to each input, the layer is called fully connected or dense (Géron, 2019).

Perceptrons are trained based on the theory of "*Cells that fire together, wire together*" (Hebb's rule): if two neurons tend to be triggered together, the relationship between them is stronger. Using this rule, perceptrons are trained by observing the links between input and output that reduce the error in the final output: these will have to grow stronger, while the weight of the connections that increase the error will be reduced. This algorithm resembles the stochastic gradient descent, according to which the weights are recalculated for each step according to their ability to minimize the error between correct answer and prediction (Géron, 2019).

The main limitations of a single perceptron can be solved by using a multi-layered structure (multilayer perceptron, MLP), composed of one input layer, one or more hidden layers and an output layer, each containing a defined number of neurons and a bias, all fully connected to each other. The best way to achieve MLP training is the backpropagation algorithm, based on stochastic gradient descent: this method computes the gradients of the error with respect to each parameter of the model, and states how the connection weights should be adjusted to minimize the error of computation, and repeats the process until converging to a final solution. The steps of this procedure are:

- Division of training dataset in mini batches (epochs).
- Forward pass: computation of model in each instance (element) of each mini-batch and storage of all connection weights between the perceptrons of different layers, which have to be adjusted later.
- Computation of prediction error with respect to expected result and measure of how much each connection contributes to this error.
- Backward pass: adjustment of each connection going back in the model according to the prediction error.
- Gradient descent until convergence to a minimum error.

The initialization of connection weights in the first step should be performed in a random way to avoid training failure (Géron, 2019).

ML methods (ANN in this specific case) are successful and useful in both classification (i.e., association of an input to its corresponding output in a range of possible choices) and regression problems (e.g., parameter optimization and property predictions from remotely sensed reflectance). In this project, ANN are used to solve a regression task and result particularly effective for the consideration of geospatial parameters in their dynamics instead of a static way.

However, the use of models entirely based on ML in ESS presents some fundamental challenges (Reichstein et al., 2019):

- Interpretability: ML algorithms are based on a black-box configuration, so the interpretation of results from methods only based on ML can be challenging because it is not possible to have a complete view of the intermediate steps of the process.
- Physical consistency: the model results need to be traced back to actual physical conditions in order to be meaningful. The use of algorithms purely based on ML does not take into account the laws of physics, possibly resulting in scientifically inaccurate results.
- Uncertainty of data: the performance of a ML strongly depends on the quality of the data provided, which in ESS may not always be as high as required due to missing observations or measurement inaccuracy. Moreover, the best performances are achieved with a high quantity of labelled data, which is not always available.

Considering the advantages and limitations of both PBM and ML algorithms, it is possible to observe that the development of hybrid models, combining theory-driven and data-driven approach, is a potential solution to obtain the best possible results. More specifically, hybrid models have the direct interpretability of PBM, are able to give constraints to results to make them physically meaningful and can be used to make future projections for predictive tasks and decision-making. On the other hand, the combination with data-driven algorithms gives the model the possibility to manage a very high quantity of data, to be very adaptable to different settings and to be able to identify unexpected patterns easily, while reducing the computational demand and the time needed for training and obtaining results (Reichstein et al., 2019). *Figure 4* highlights the strengths and the potential challenges and opportunities derived by the adoption of a hybrid approach to ESS.

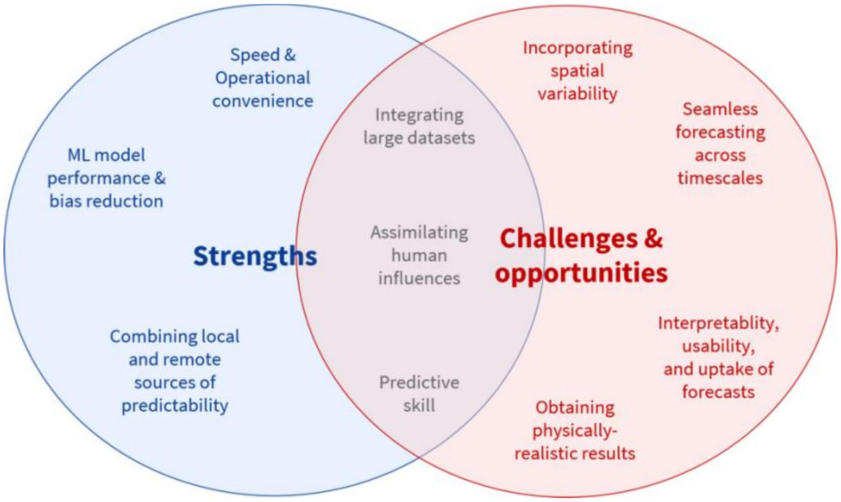
Speed & Operational convenience
ML model performance & bias reduction
Strengths
Combining local and remote sources of predictability
Integrating large datasets
Assimilating human influences
Predictive skill
Incorporating spatial variability
Seamless forecasting across timescales
Challenges & opportunities
Interpretablity, usability, and uptake of forecasts
Obtaining physically-realistic results

Figure 4 - Summary of strengths, challenges and opportunities of hybrid modelling in ESS and climate predictions (Slater et al., 2023).

There are different studies which aim at confirming the advantages brought by a hybrid approach in various fields of Earth System Science. For instance, in the field of rainfall prediction, where pure ML models were already proved to be sometimes more effective than physical models, the introduction of a hybrid system allowed to reach more accurate forecasts and to reduce uncertainty both at short and long time scales (Dotse et al., 2023).

For what concerns the specific field of soil carbon, a study on the quantification of SOC in sown pastures (Liu et al., 2023) was able to observe that the Knowledge-Guided Machine Learning model (KGML, i.e., the integration of ML and RS observations) outperformed the pure ML algorithm for any sample size, had lower sensitivity to the number of training samples and was also able to make reliable predictions in extreme conditions. The performance of such models can be improved by choosing a complete and well-suited process-based model to couple with the algorithm, and it can result in an accurate, cost-effective and high-resolution (both spatially and temporally) estimate of carbon budgets, crop yields and variation of SOC.

3. Objectives

The main goal of this study was to develop a hybrid dynamic model for SOC, which could combine the strengths of a simple process-based equation with a ML approach to integrate remotely sensed data. More specifically, a SOM dataset collected on the field in multiple pasture farms was used to calibrate the model, where the parameters were estimated using an ANN that had as main input parameters computed using remotely sensed data. Once the parameter estimation was performed and the performance of the model was assessed, the results were transformed into SOC using the van Bemmelen factor.

The study started from the manipulation of the initial dataset into a table containing couples of measurements of SOM in the same point, along with the time difference between observations, and a series of other radiometric, physical and locational features referred to the point itself.

The development of the model, which was completely performed in the Python environment (version 3.12.2, in the environments Jupyter and PyCharm), included the creation of two separate ANN to estimate the parameters of the process-based equation (carbon input and mineralization), and it consisted in finding the ideal hyperparameters that performed the best in terms of fitting the data while minimizing the overfitting effect, and selecting the variables to insert in the training of the two ANN from the list provided in the database according to the best performance obtainable.

The main difference between a standard application of ANN and this project lied in the error assessment. Typically, the training phase consists in computing the parameters of the model, and then backpropagating the obtained values to the input features, to adjust the weights of the connections between neurons, with the goal to find the parameter values that allow loss minimization. Only after this procedure, the estimated parameters are inserted in the model. In this case, the parameters were calculated and immediately inserted in the model during the training of the models. In this way, the loss function was evaluated as the difference between the calculated result using the ANN to compute parameters that are then entered in the process-based equation, and the measured values of SOM provided. This strategy allowed the models to combine complex data features in an effective way through the ML technique and, therefore, obtain the best possible values considering many different features, but at the same time the parameters should have physical meaning because they were calculated according to a process-based relationship which took into consideration the actual processes that were occurring (which the ML algorithm is not able to do by itself).

The result was a model whose parameters are specific for each combination of input variables, and therefore for each region. This fact improves the likeliness that the model can generalise different environmental conditions and calculate the possible evolution of SOC in different areas than the ones used in the training, at least the ones with similar climatic features. New site-specific parameters are calculated (using the ANN) every time the model is run for a new location without the need for inverting the model to match new observations. Moreover, the model should be able to simulate future projections, or at least give an idea on how the quantity of SOC could evolve by setting possible values for future conditions of soil (e.g., temperature and moisture).

4. Materials and methods

4.1 Case study description

The sites used for data collection were selected in a previous study (Morais et al., 2023), which was focused on modelling carbon concentration in soils of SBPPRL using only ML and remotely sensed data.

The choice of SBPPRL as reference land management practice was due to its potential to increase grassland productivity, and the many co-benefits that could arise from an effective implementation of this land use strategy, including high carbon sequestration.

The system was developed in Portugal in the 1970s as a win-win strategy from the economic and environmental point of view because it combined the private interests of farmers (improved production at low cost), and the ecosystem services provided by the engineered management of soil. Before the development of this technique, Portuguese agricultural areas had been experiencing both abandonment of agricultural areas and extensification of the remaining ones (due to labour price increase and selling price decrease, which is not sustainable for small farmers), which lead to the formation of shrub woody cover, that had strong consequences on soil structure and its ability to retain water, as well as resulting in an increase in risk of wildfires. Both these phenomena, combined with tillage, degraded the quality of soil by increasing the mineralization rate and, therefore, decreasing the natural soil carbon stock. Since spontaneous re-vegetation was proved to not be efficient in improving soil quality in abandoned agricultural fields, some kind of human management and intensification is required, and SBPPRL are an example of possible intensification, featuring a mix of different species tailored to the soil, climate and agricultural conditions of the studied field (Teixeira et al., 2015).

SBPPRL are installed by sowing a mix of up to 20 different species of grass and legumes. It is a strategy of agricultural and livestock production intensification in an engineered, sustainable way (Teixeira et al., 2015). If correctly maintained, the system could significantly improve ecosystem degradation in semi-arid and sub-humid climates. In this context, the role of legumes is specifically to improve nitrogen fixation from the atmosphere, making the area independent from external N fertilization. However, in the first settling years, legumes need generous applications of phosphate to grow, since Mediterranean soil is poor in this nutrient. Legumes cover over 50% of the agricultural field in the first years of development, but then grassland takes over in more mature conditions (legumes stabilize around 25/30% of extension after around 5 years of development) (Teixeira et al., 2015). The ratio between grass and legumes is fundamental to reach the best performance in terms of productivity and nitrogen fixation, and this can be achieved with different practices, including correct fertilization and grazing management. Grazing should be planned in such a way that, after summer, no excess of dry vegetation is found, to avoid problems in seed break down and germination. Therefore, in this period heavy grazing is suggested, while during flowering and maturation stages overgrazing is not advised. (Teixeira et al., 2015)

The system is expected to deliver several benefits for farmers, including higher yields and better-quality pastures; the replenishment of soil organic pools, which can act as a carbon sink; an improvement of soil structure; the decrease of surface runoff and the reduction of pyrophyte vegetation.

The studied effects of the installation of SBPPRL compared to semi-natural pastures (SNP) show, as previously mentioned, both economic and environmental advantages (Teixeira et al., 2015). Starting from the economic point of view, which is crucial to obtain the consent of farmers to apply the procedure, the main upside is the increased productivity, related to the selection of hard seeds, the ecological complementarity between the different species and a density increase of products (not size increase). Another economic benefit is the natural N fixation ability of legumes, which allows to cultivate without the use of synthetic N fertilizers. Improving agricultural productivity could result in some additional life-cycle effects, with both positive and negative features: increased animal grazing would lead to higher CH_4 emissions and limitations to wildlife biodiversity but would reduce the need for tillage since the increased livestock would autonomously manage the woody shrub growth, and less need for concentrated feeds, which have huge environmental impact and would compensate for the additional inputs required by SBPPRL (phosphorous fertilization, limestone application) (Teixeira et al., 2015).

From the environmental point of view, one key benefit, and main point of interest for this research, is the increase in the SOM pool of the soil, which brings additional co-benefits such as an increase of soil quality which reduces erosion risk, improved carbon sequestration and decreased surface runoff (which can also affect the receiving water bodies through eutrophication, silting and contamination). This is expected because SOM confers structure, stability and nutrients to soils, increases plant productivity without need for tillage and, therefore, decreases erosion, desertification and superficial water runoff.

Due to the establishment of those effects, between 2009 and 2014, the country financed carbon sequestered in these pastures through the Portuguese Carbon Fund (Teixeira et al., 2015). SOM was shown to increase by +0.21% per year in the first 10 years of establishment of SBPPRL, which is much higher than the results obtained for Natural Grasslands (NG) and Fertilized Natural Grasslands (FNG), for which the SOM increase was estimated to be around +0.08% per year (the results are more or less the same for NG and FNG because the fertilization does not have a strong impact on SOM content), starting from the same initial conditions (Teixeira et al., 2011). This result can be explained with the higher biomass production occurring in SBPPRL, which supports a much higher stocking rate. The balance in GHG emission/offset included the improved carbon sequestration ability of SBPPRL, as well as the emissions related to limestone application (for soil pH management), bacterial nitrification and livestock enteric fermentation and manure. All in all, SBPPRL can hold carbon sinks of 1.55-2.13 t_{CO_2} ha^{-1} y^{-1}, depending on the mineral bulk density of the site, which compared to the values computed for NG and FNG (0.53-0.75 t_{CO_2} ha^{-1} y^{-1}) represents a significant improvement (Teixeira, 2010).

In addition to this, SBPPRL have been observed to have higher resistance to extreme conditions and environmental pressure, thanks to the variety of species, which is fundamental because of the expected alterations and additional pressures related to climate change. Finally, the enhancement of N-pools in this type of pasture has been shown to be effective and the risk of N-losses is very low, but longer time series need to be observed before establishing effects in the long-term.

On the other hand, the main limitations expected are the dependency of legumes on phosphorous fertilization, which is required in the installation phase and sometimes for maintenance, and comes from non-renewable resources, and the unclear effects on wild biodiversity. There are still many uncertainties about this agricultural system which should be addressed: the implementation of increased monitoring and data time-series to assess more precisely the effects in long-term conditions, the management of main negative effects (including possible increase in non-CO_2 GHG emissions and, mostly, the use of phosphate fertilizer), and the possibility of upscaling to other semi-arid/sub-humid areas other than Portugal (Teixeira et al., 2015).

4.2 Data used

4.2.1 SOC measurements

The data used in this research were collected in eight different farms throughout Portugal (farms 1, 2, 3, 5, 6, 7, 8, 9), and one located in Spain (farm 4). The plots surveyed in those farms had variable areas (ranging between 26 and 42 ha), and spanned across latitudes and longitudes respectively between 37°50’ – 40°30’ N and 6°80’ – 8°30’ W. The soil types and the dominant parent materials were determined using the European Soil Database (Morais et al., 2023). The climate of all farms, according to the Köppen climate classification system, is in the hot-summer Mediterranean region (Csa), which is characterised by coldest month averaging above 0 °C (or −3 °C), at least one monthly average temperature above 22 °C, and at least four months averaging above 10 °C. At least three times as much precipitation in the wettest month of winter as in the driest month of summer is expected, and the driest month of summer receives less than 40 mm (Arnfield, 2024). The location of each farm is illustrated in *Figure 5*.

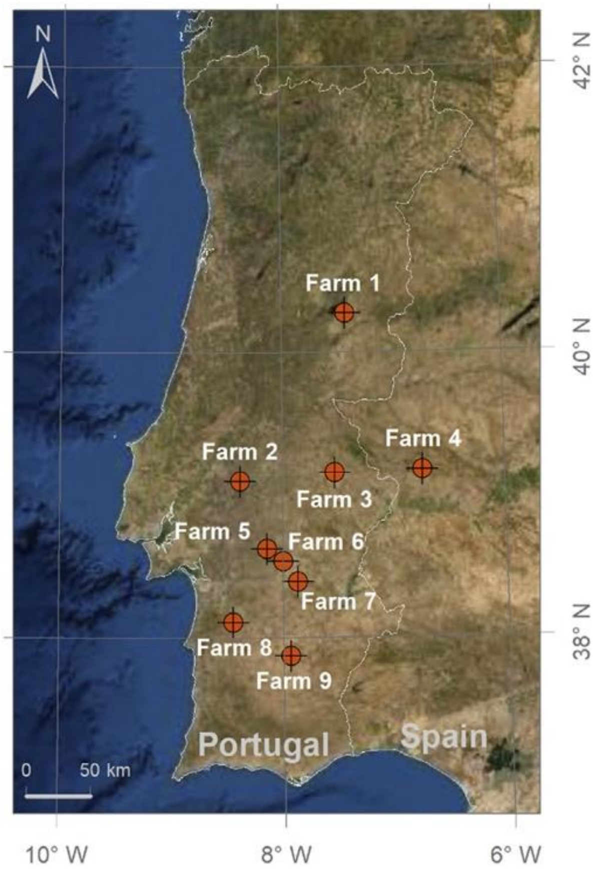

Figure 5 - Map showing the location of the farms where data were collected (Morais et al., 2023)

In each farm, different soil sample collection campaigns were held, in different periods ranging between four years of production (from 2017-2018 to 2020-2021), with the aim to measure the SOM content in the study area. Each sampling season started in September and ended in May and, in some cases, more than one sampling per season was performed. The locations were chosen with the goal to minimize the influence of rocks and trees on the measured values of SOM, but due to the high density of trees it was not always possible to collect the samples in equal number for each farm. *Table 1* shows the difference in each sampling campaign, which was variable not only between farms but also depending on the collection date. From *Table 1* it is also possible to observe that the time interval between two consecutive collections was strongly variable, and this had to be considered in the development of the model, in which couples of consecutive measurements were needed. A total of 1121 sampling points was collected from the farms throughout the years, but further considerations had to be made to build the final database used for the development of the model. The values of SOM were expressed in $kg_{SOM}/100\ kg_{soil}$.

Table 1 - Summary of all the sampling campaigns in the different farms: for each farm the date and number of samples collected is reported.

Farm Code	Production year	Collection date	Number of points	Total per year	Total per farm
1	2017-18	24/02/18	8	40	237
		16/04/18	24		
		17/05/18	8		
	2018-19	22/11/18	26	75	
		09/01/19	21		
		20/02/19	8		
		23/04/19	12		
		22/05/19	8		
	2019-20	09/12/19	34	58	
		19/02/20	12		
		14/04/20	12		
	2020-21	16/01/21	42	64	
		15/02/21	11		
		13/04/21	11		
2	2019-20	06/11/19	35	35	35
3	2017-18	09/04/18	24	32	203
		15/05/18	8		
	2018-19	21/01/19	47	71	
		13/02/19	6		
		16/04/19	12		
		17/05/19	6		
	2019-20	18/10/19	33	57	
		18/02/20	12		
		13/04/20	12		
	2020-21	04/03/21	31	43	
		12/04/21	12		
4	2018-19	04/02/19	12	24	24
		18/04/19	12		
5	2018-19	14/01/19	50	74	184
		11/02/19	6		
		08/04/19	12		
		13/05/19	6		
	2019-20	25/10/19	34	58	
		10/02/20	12		
		06/04/20	12		
	2020-21	08/02/21	12	52	
		03/03/21	28		

Farm Code	Production year	Collection date	Number of points	Total per year	Total per farm
		07/04/21	12		
6	2017-18	16/02/18	7	39	219
		02/04/18	24		
		14/05/18	8		
	2018-19	15/01/19	57	72	
		12/02/19	8		
		15/05/19	7		
	2019-20	24/10/19	33	57	
		12/02/20	12		
		08/04/20	12		
	2020-21	10/02/21	12	51	
		01/03/21	27		
		08/04/21	12		
7	2018-19	11/04/19	12	12	75
	2019-20	30/10/19	33	33	
	2020-21	02/03/21	30	30	
8	2018-19	19/02/19	8	28	132
		22/04/19	12		
		20/05/19	8		
	2019-20	29/10/19	29	51	
		06/02/20	12		
		02/05/20	10		
	2020-21	02/02/21	12	53	
		19/02/21	29		
		05/04/21	12		
9	2018-19	12/04/19	12	12	12
				Total points	1121

Since the model development implied verifying the change in time of the value of SOM, only the points with two or more measurements could be considered. For this reason, all the points from farm 2 and farm 9 had to be excluded from the database. Furthermore, a consideration of the single points was performed to observe which ones could be used in the remaining farms. In this phase, a study of the measurement process had to be taken into account: each soil sample was composed by four sub-samples that were pooled and mixed to achieve uniformity (Morais et al., 2023). In the considered farms, points with only one measurement were excluded from the database, and the remaining ones were sorted in chronological order. Moreover, the latest observation for each point was excluded, again because these would not have a following SOM measurement.

This way, each line of the database contained a value of SOM(t) and a value of SOM(t+Δt), where Δt is, as previously mentioned, a variable time interval. After this selection, the database contained 718 couples of points.

Finally, an outlier analysis was carried using SOM(t+Δt) – SOM(t) as variable of analysis, to exclude possible experimental errors which could give unrealistic values in SOM variation with respect to the time interval considered. As showed by *Figure 6,* the values seemed to be distributed mainly between the lower and upper whisker, and it was possible to identify only 11 outliers between the 718 in the dataset.

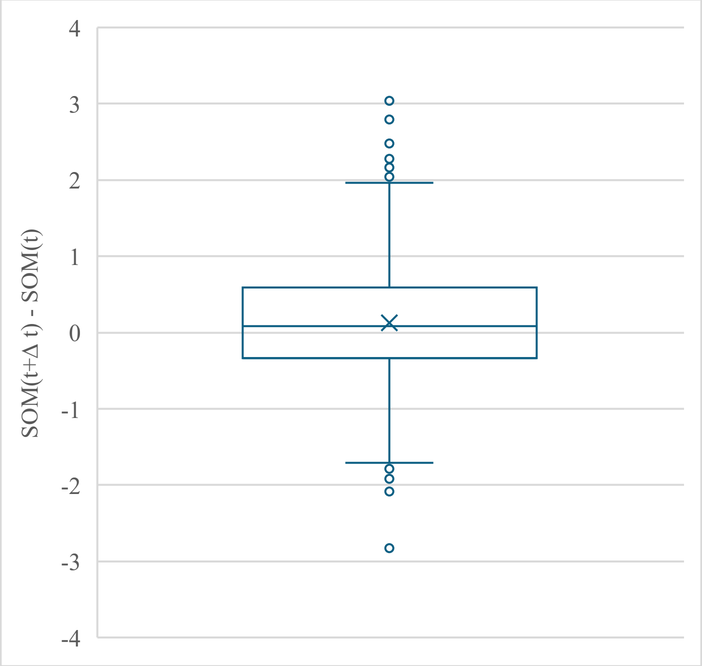

Figure 6 - Boxplot of the distribution of SOM(t+Δt) - SOM(t). The points outside the boundaries are the outliers, that were excluded from the final table.

Removing the outliers, the database showed a total of 707 lines, each containing couples of SOM measurements, which could be used for the model training, validation and testing.

4.2.2 Input variables used

After grouping the couples of measurements of SOM, the input variables for the model were compiled. A database was obtained collecting different parameters related to geospatial features of the points. As previously explained, the goal was to integrate data from samples and radiometric data (acquired through RS techniques), which could be used as explanatory variables to estimate SOM using ML. The complete set of data, containing a total of 49 parameters, referred to the same study from which the measurements of Soil Organic Matter were taken (Morais et al., 2023), and included radiometric data such as reflection bands and vegetation indices, as well as climatic, soil and terrain variables, and some auxiliary data. All values were collected using GEE, which is a cloud-based platform containing a large amount of geospatial data, and allows to track changes over time, map trends and quantify differences.

Since the final goal of the project was to construct a model able to make predictions in the future regarding the possible SOM content of a specific site given some changes in the climatic scenarios predicted for the area, some variables were considered both at the time of acquisition of the first sample (SOM(t)), and after the time interval of the consecutive measurement (SOM(t+Δt)). The variables chosen to be considered at both instants were the ones for which it was possible to make realistic future predictions, which were basically the soil moisture and soil temperature. All the other variables, including remotely sensed data, topographic indices, soil composition and pH were considered to remain constant through time, so there was no need to consider their values at t and t+Δt.

Table 2 provides a summary of the variables present in the original database that were selected for this study, together with their source database in GEE.

Table 2 - Summary of all the variables present in the original database (Morais et al., 2023) and used in the current project, with their sources.

	Type of variable	**Source database**	**Number of variables**
Satellite bands (August)	Radiometric	Sentinel-1, Sentinel-2	12
Satellite bands (closest to harvesting)	Radiometric	Sentinel-1, Sentinel-2	12
Soil composition	Climate/soil/terrain	SoilGrids	3
Soil water pH	Climate/soil/terrain	SoilGrids	1
Soil moisture	Climate/soil/terrain	GLDAS	2
Soil temperature	Climate/soil/terrain	GLDAS	2
Vegetation indices (August)	Radiometric	Computed from satellite bands	5
Vegetation indices (closest to harvesting)	Radiometric	Computed from satellite bands	5
Day (days since beginning of production)	Auxiliary	-	1
Lab	Auxiliary	-	1
Topographic indices	Climate/soil/terrain	NASA EOSDIS Land Processes DAAC	5
Location	Auxiliary	-	2

From *Table 2*, it is possible to observe the total number of variables which were considered as possible inputs to train the ANN to find the model parameters carbon input and mineralization rate. Collectively, the number of variables identified is 51, related to a total of 707 observations:

the ratio observations/variables deriving from this is 707/51 = 13.86. Studies on the relationship between number of events per variable in logistic regression (Peduzzi et al., 1996) state, as rule-of-thumb, that to ensure model accuracy and prevent overfitting the recommendation is to have at least 10 observations per variable. Even though the ratio previously computed satisfied this requirement, it was very close to the lower limit stated by Peduzzi et al. (1996), therefore a sort of variable selection, described in *Section 4.3.3.2*, was implemented, to reduce the number of variables and ensure higher accuracy.

4.2.2.1 Radiometric data

Radiometric data was acquired by joining information from the Sentinel-1 and Sentinel-2 missions, in which image resolution was variable depending on the satellite band considered, and there were three possible resolutions: 10 m for Blue, Green, Red and Near Infrared (NIR) bands; 20 m for the three Vegetation Red Edge bands, the Narrow NIR band, and the two Shortwave Infrared (SWIR) bands; and 60 m for Coastal Aerosol, Water Vapour and SWIR-Cirrus bands. All data was taken from Level-2A products, therefore acquired at the Bottom of Atmosphere (BOA). The considered images were acquired in two different periods of the year: the first (B1, B2, B3, B4, B5, B6, B7, B8, B8A, B9, B11, B12) was a composite frame of the available data in the period between the 1st and 31st of August, which was supposed to represent the bare soil condition and to capture the spectral reflectance of the soil, while the second (B1_close, …, B12_close) was a composite frame of images at the closest date to collection, representing the maximum vegetation cover to assess the influence of vegetation on SOM. The values of Bn and Bn_close collectively contributed to give 24 input variables for the evaluation of the parameters carbon input K and mineralization rate α, which were object of this study.

The presence and the contribution of vegetation was accounted using five different vegetation indices, always based on radiometric data and, specifically, some values of spectral reflectance in specific bands, variable depending on the index (Morais et al., 2023). Since they were also related to spectral reflectance, vegetation indices were accounted both in bare soil conditions and at closest date before collection (moment with highest vegetation density):

- NDVI (Normalized Difference Vegetation Index), used to understand vegetation density and plant health by quantifying vegetation greenness. It was computed as the ratio between the red (R) and the near infrared (NIR), namely

  $$NDVI = \frac{NIR-R}{NIR+R}. \tag{Eq. 1}$$

  NDVI is the most commonly used vegetation index, and the most suitable to track crop development dynamics, but it is quite sensitive to soil brightness and atmospheric effects, so it needs to be used in cooperation with other indicators (such as SAVI).
- NDWI (Normalized Difference Water Index), used to monitor the water content of leaves to mitigate soil brightness effect. It was evaluated through the ratio between the green (G) and the near infrared (NIR) radiometric values

$$NDWI = \frac{G-NIR}{G+NIR}. \tag{Eq. 2}$$

- SR (Simple Ratio), which provided a general indication of vegetation health (high values = healthy vegetation, low values = bare soil/water/ice). Computed simply as the ratio between NIR and R radiometric values

$$SR = \frac{NIR}{R}. \tag{Eq. 3}$$

- SAVI (Soil-Adjusted Vegetation Index), which is a vegetation index aimed at minimizing soil brightness influence, so it is particularly useful in arid areas where vegetation cover is low. Soil brightness influence was addressed using a correction factor L. In this application, L was set to 0.5, but it can vary between -1 and +1 depending on the site's vegetation cover (= 0 in highly vegetated areas, so that SAVI = NDVI, and = 1 in low vegetation zones)

$$SAVI = 1.5\frac{NIR-R}{NIR+R+0.5}. \tag{Eq. 4}$$

- OSAVI (Optimized Soil-Adjusted Vegetation Index), which used a different background adjustment factor to modify and optimize SAVI index. This allowed to get higher sensitivity to canopy cover greater than 50%

$$OSAVI = 1.16\frac{NIR\_R}{NIR+R+0.16}. \tag{Eq. 5}$$

Collectively, the vegetation indices at sampling date and at closest date to collection contributed with 10 additional variables to the evaluation of the parameters carbon input and mineralization rate.

4.2.2.2 Climate/soil/terrain data

Since the processes related to the carbon cycle (i.e., mineralization and accumulation of SOC) strongly depend on climatic and environmental features of the site, some data related to climate, soil and terrain was also collected. All the parameters were collected from GEE, but each from different databases specific for the variable. The considered parameters are the following:

- Soil temperature at 10 cm depth (in K). The measurement at current time was indicated as SoilTMP (t), while the consecutive measurement at the same point was indicated as SoilTMP (t+1), contributing with 2 additional variables to the evaluation of the parameters carbon input and mineralization rate.
- Soil moisture at 10 cm depth, expressed in percentage value. The measurement at current time was indicated as SoilMoi (t), while the consecutive measurement at the same point was indicated as SoilMoi (t+1), contributing with 2 additional variables to the evaluation of the parameters carbon input and mineralization rate.

- Soil composition (clay/sand/silt content in %). In the database, the percentage values were multiplied by 10 with respect to their original format, but in the code each parameter was normalized, therefore no modification was necessary. The values contributed to 3 input variables for the evaluation of the parameters carbon input and mineralization rate.
- Soil water pH: pH_{H2O} was used instead of measuring directly the soil's pH, because the first is easier to measure than the latter, and it is still well representative of soil conditions. In the database, the pH values were multiplied by 10 with respect to their original format, but in the code each parameter was normalized, therefore no modification was necessary. The values contributed to 1 input variable for the evaluation of the parameters carbon input and mineralization rate.
- Topographic indices, also called terrain variables, used to assess the effect of the position and the conformation of the territory on SOC:
  - Continuous heat-insolation load index (CHILI): used to assess the effects of insolation and topographic shading on evapotranspiration, by computing the insolation at early afternoon, at sun altitude equivalent to equinox. It was be used to identify warm, neutral, and cool areas of a landscape.
  - Digital elevation model (DEM): representation of the base ground topographic surface of the studied area, without accounting for trees, buildings or other surface objects.
  - LANDFORMS
  - Multi-scale topographic position index (mTPI): used to distinguish ridge from valley forms, it was calculated by the elevation at each location subtracted by the mean elevation within a neighbourhood.
  - Topographic Diversity (TopoDivers): represented the variety of temperature and moisture conditions available to species as local habitats (calculated combining mTPI and soil moisture).

  The topographic indices contributed to 5 input variables for the evaluation of the parameters carbon input and mineralization rate.

4.2.2.3 Auxiliary data

The third group of variables included auxiliary data, mostly related to the moment and the methodology of collection of the samples for the measurement of SOM, and included day, month and year of collection, days between the closest satellite image (for both Sentinel-1 and Sentinel-2) and sampling date, days since beginning of production year (31st August), laboratory of sample analysis and location of the sample through its coordinates (latitude and longitude). More specifically, the four auxiliary variables used in the database were:
- Number of days since the beginning of the production year (column indicated as *Days*)
- Laboratory of sample analysis (1 or 2), because the two laboratories used different analysis techniques (column indicated as *Lab*)
- Location of the sample in latitude and longitude (columns indicated as *lat* and *lon*).

4.3 Hybrid model development

The structure of the model combined the data previously described, including all the satellite information from RS and the auxiliary variables related to the structure of the site, and a simple process-based relationship for the computation of SOM given a time interval Δt, which was variable as the sampling campaigns were not held in a regular way. The process-based model used here was a simple, 0D, 2-parameter model introduced for quantifying the dynamics of SOM in Portuguese grasslands (Teixeira et al., 2011). It was a very simple model, based on only two parameters: the Carbon input (K), accounting for an increase of SOM related to vegetation and livestock presence, and the mineralization rate (α), describing the dynamics of SOM loss, according to

$$SOM(t + \Delta t) = \frac{K}{\alpha} \cdot (1 - e^{-\alpha \cdot \Delta t}) + e^{-\alpha \cdot \Delta t} \cdot SOM(t). \qquad \text{(Eq. 6)}$$

The model was based on a mass-balance principle: the dynamics of organic matter in soil depended on carbon external inputs (such as animal manure and plant residues), which were balanced by the mineralization rate. Given constant input and mineralisation rates, the model will asymptotically reach a steady state (with a time-dependent saturating exponential form), while a variation of land management practices could temporarily modify the SOM content trend, increasing the sequestration capacity of soil (Teixeira et al., 2011).

In the previous study (Teixeira et al., 2011), the values of the parameters were estimated using the OLS method and stepwise regression, based on SOM observations collected in the previous years. The present study aimed at increasing the prediction accuracy of the same model, by substituting the regression equations of OLS with ANN, which from the dataset previously described should be trained to determine the most realistic values of K and α. If the performance of the hybrid model is an improvement over the regression version, this work could be interpreted as a proof of concept showing the potential for this approach to improve existing soil models.

Before the development of the model with ANN, the parameters were estimated with a simple linear regression, to obtain a baseline performance value to compare with the results of the more complex procedure, which should produce a more accurate data fitting.

4.3.1 Linear regression

Since the used model to compute the value of SOM(t+Δt) presented a linear relationship with respect to the initial value of SOM(t), the first strategy that was applied was to use the basic linear regression to fit the data, to observe if the natural relationship between a measurement and the consecutive one was already compliant with the model. To achieve this, the expression of the linear relationship was used to find the values of the two parameters (K and α). The expressions used for this evaluation are

$$y = m \cdot x + q, \tag{Eq. 7}$$

$$m = e^{-\alpha \cdot \Delta t} \rightarrow \alpha = -\frac{\ln(m)}{\Delta t}, \tag{Eq. 8}$$

$$q = \frac{K}{\alpha} \cdot (1 - e^{-\alpha \cdot \Delta t}) \rightarrow K = \frac{q \cdot \alpha}{1 - e^{-\alpha \cdot \Delta t}}. \tag{Eq. 9}$$

In order to have a procedure and a number of observations comparable to the ones applied on the ANN model, before performing the linear regression the dataset was randomly split into a training (80% of observations) and a test (20% of observations) subset. The training dataset was interpolated to determine the values of m and q, and the formulas listed above were applied to obtain the final values of K and α. In this step, the mean value of Δt in the training set was used as fixed value. Once the parameters of the model were obtained, the values of SOM(t) in the test set were used to estimate SOM(t+Δt) through *Eq. 6*, and finally the performance was evaluated by calculating the coefficient of determination R^2 between measurements and modelled values, with the following relationship:

$$R^2 = 1 - \frac{\sum(\,SOM(t+1)_{test} - \overline{(SOM(t+1)_{test})}\,)^2}{\sum(\,SOM(t+1)_{test} - SOM(t+1)_{pred})^2}. \tag{Eq. 10}$$

The code used for this purpose can be consulted in *Section 9.1*.

4.3.2 Definition of the ANN

The goal of the study was to substitute the estimations of the two model parameters (K and α) with two separate ANN (in this case operating as regression tools, not as classification ones), able to fit all the input data into a generalized value for each parameter, which could be used to fit data from every site, and make predictions about future conditions of SOM.

Both ANN had the same structure, which was very simple: one input layer (with variable number of input features based on the variable assignment, as discussed in *Section 4.3.3.2*) fully connected to a single hidden layer (with variable number of neurons, which was the main hyperparameter to be tuned) though linear relationships, which was again linearly connected to the output layer, that had one single perceptron containing the result of the regression process.

After the first linear layer, a ReLU (Rectified Linear Unit) activation function was applied, to introduce some non-linearity in the model, which was fundamental to learn more complex patterns. Moreover, a layer with a dropout rate was introduced in the ANN: this function set randomly a percentage of the neurons to zero, avoiding the model to be too reliant on a single element and preventing overfitting. During the definition of the ANN, the variables that were defined included the number of input features, the number of neurons in the hidden layer, the dropout rate and the output activation function. The latter was introduced to satisfy the only requirement that was defined for the values of the two parameters: since the mineralization rate represented the fraction of mineralized SOM, it could not be greater than 1, since it would mean that the amount of carbon that is mineralized would be higher than the one actually available on site. This is why, on the model for the computation of α, the activation was set to a sigmoid function, commonly used to map any real number to a value between 0 and 1 (as showed in *Figure 7*)

$$\sigma(x) = \frac{1}{1+e^{-x}}. \qquad \text{(Eq. 11)}$$

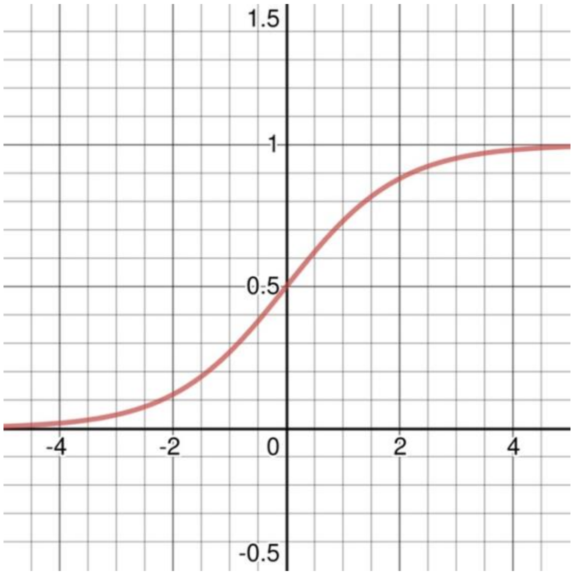

Figure 7 - Graphic representation of the sigmoid function (Activation Functions & Derivatives, n.d.)

Initially, a simple sigmoid activation function was applied for the computation of the mineralization, but observing previous studies (Teixeira et al., 2011), a further reduction was considered reasonable, and a customised activation was introduced to limit the maximum value of α to 0.3 (roughly doubling the values obtained for similar land uses).

Before the introduction of these activation functions, the values of mineralization were significantly higher than expected: without any output activation the model often resulted in values greater than 1, and even with the simple sigmoid the fitting gave values very close to 1 (around 0.9). After the introduction of the output activation, the values of α were reduced, but this resulted in a consequent reduction of the K parameter, which in some cases was found to be lower than zero. This was not physically possible, because negative C input would be an effect of no external inputs, which could not be the case since all the considered soils included both vegetation and livestock, which surely were sources of carbon, combined with strong mineralization, which should not be accounted in the K parameter because it was already considered in the mineralization rate. Therefore, the final version of the code was modified adding an output activation for the model computing the C input as well, to always give it values strictly greater than zero. The activation function that was chosen for this purpose is the softplus activation, a smooth approximation of the ReLU function, which avoids the discontinuity in the derivative that the latter has in 0. The function has a behaviour similar to linearity for positive values, and it asymptotically approaches 0 for negative results (as showed in *Figure 8*)

$$Softplus(x) = \ln(1 + e^x). \qquad \text{(Eq. 12)}$$

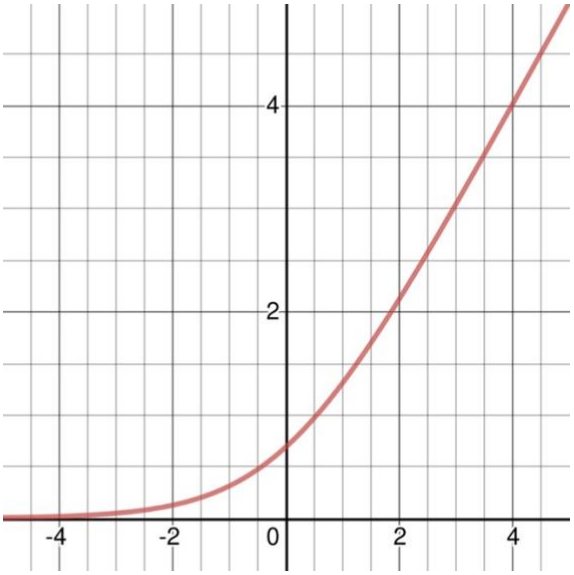

Figure 8 - Graphic representation of the Softplus function (Activation Functions & Derivatives, n.d.)

The ANN structure here defined, as previously mentioned, was applied in two separate instances for the computation of the two parameters K and α. The following steps were the training and validation of the network, which was connected to the hyperparameter tuning for the definition of the optimal number of neurons in the hidden layer, then finally the test phase, with the computation of the values of the parameters and the evaluation of the model performance through both a loss function and the R^2.
The final version of the model used to obtain the results can be consulted in *Section 9.3*.

4.3.3 Model development workflow

To give a clearer idea of the procedure that was followed and before going more into detail in the methods followed for the study, this section describes the workflow used to develop the final hybrid model.
First, the dataset was divided into three different subsets: the training set, containing 60% of the observations in the database, the validation and the test set, each containing 20% of the observations. This step was performed using the stratified splitting technique, to ensure that each subset had a regular distribution of the variable SOM(t+Δt), i.e., in each set the distributions of SOM(t+Δt) had a similar average and standard deviation.
After the dataset splitting, two separate ANN models, one for each parameter (called *model f* for carbon input and *model g* for mineralization rate), were defined. The two models corresponded to

two ANN with the structure defined in *Section 4.3.2*: each of them received a set of input variables from the database as input layer, had one single hidden layer, and an output layer which produced the estimates of the parameters. The number and type of input features was assigned randomly at each iteration, to reduce the number of variables with respect to the observations contained in the dataset. This was done to avoid any prior assignment of input variables to the models, which would require assumptions about the potential for each variable type to influence carbon input or mineralization. The ANN in their initial configuration were trained for 1000 epochs.

Hyperparamer tuning was performed first for the number of neurons in the hidden layer of each of the two models, which was carried out through a nested double loop at each iteration, to compute the loss function for each possible combination and choose the two sizes that minimized it. Then, the model was improved by optimizing additional hyperparameters i.e., the dropout rate and the learning rate scheduler. The dropout rate is a commonly used hyperparameter to prevent overfitting: it refers to dropping out (i.e., temporarily removing) hidden or visible units, along with their connections, from the network during training. The choice of the units to drop is random, and it leads to the creation of a "thinned" ANN, preventing it from co-adapting too much (Srivastava et al., 2014). The learning rate scheduler was used to track the performance of the model, and if it did not improve for a defined number of epochs (called patience), it multiplied the initial learning rate by a reduction factor. The tuning of these hyperparameters (dropout rate, patience and factor of the scheduler), as well as the number of training epochs, was achieved by setting a fixed variable configuration (the one with highest R^2 from the previous step) and modifying the values of the hyperparameters until reaching the best fit.

The model was run 100 times in this configuration, to find the random assignment of input variables that maximised the performance in terms of R^2, with random variable assignment and the previously tuned hyperparameters.

Each of these steps is explained in further detail in the next sub-sections.

4.3.3.1 Dataset splitting

The first step for the development of the model was to extract the needed columns from the dataset: the first column contained the measurement of the SOM in the first instant (t), the second referred to the values of SOM in the second measurement (t+Δt), and the third was the time interval between the two measurements (Δt). The other columns contained the 51 variables to insert in the ANN.

Once these preliminary operations were performed, the dataset was divided in three subsets: the training, the validation and the test sets. In this phase, the observations were separated through the stratified splitting technique, to ensure that the dataset was divided randomly, but each subset maintained the same distribution of a predefined target variable. In this case, SOM(t+Δt) was chosen as target. More specifically, in each of the three subsets, the distribution of the target variable maintained the same (or as close as possible) average and standard deviation.

To make this division possible, the target variable was discretized into quantiles, called bins, in which the target variable had more or less the same average and standard deviation, and the final division was performed maintaining the proportion of observations having similar features. The ideal number of bins could not be too high to avoid noise presence, which would lead to overfitting, and could not be too low to avoid information loss. This is why a search algorithm for the choice of the optimal number of bins was applied before the data splitting, in which different numbers of bins were evaluated based on a balance score. This balance score was computed as the variance of bin size across all bins, which as a quantification of the dispersion was able to compute whether the data was distributed in an even way. The identified best number of bins was then applied for dataset splitting. After this step, three datasets (training, validation and test) were obtained: the first containing 60% of the observations (424 lines), and the latter two splitting the remaining 40% in equal parts (141 observations for validation and 142 in the test set).

Each of these subsets was subjected to some preprocessing steps, namely:

- Normalization: if data have different scales of magnitude, the ones with higher ranges of values will tend to dominate the smaller ones, while normalizing all values in a common range each variable is able to be expressed in the model.
- Convergence: ML algorithms tend to converge faster or have better performance when their features are scaled.

More specifically, in the code the preprocessing is performed with the feature "StandardScaler", which computes the mean and standard deviation of each column in the dataset, and it transforms the values such that the mean is equal to zero and the standard deviation is equal to one using the formula

$$x_{scaled} = \frac{x - x_{mean}}{std(x)}. \qquad \text{(Eq. 13)}$$

At the end of these procedures, the three datasets contained balanced and normalised values, and were ready to be used in the next steps of the model.

The script used to perform dataset splitting can be consulted in *Section 9.2*.

4.3.3.2 Variable selection

As previously mentioned, the total features identified to be considered in the two ANN was 51, which was relatively high with respect to the number of valid measurement couples in the dataset. This is the reason why the relevance and contribution of these variables to the computation of the model parameters K and α had to be verified, implementing a well-thought variable selection that avoided overfitting while increasing the reliability of the model and giving good estimates of the two parameters.

On a first iteration, all 51 variables were assigned as inputs for the training of both parameters, and the observed results were, generally, overestimated and did not comply with the expectations of parameters values, and the overall performance of the model was not ideal. This was probably due to collinearity in the model regarding carbon inputs to soil and mineralization.

Then, a second attempt implied the development of a random assignment of the variables to the training of each parameter: the code was modified to read through the columns of the database and randomly assign each of them to one of the categories "goes in K", "goes in α", "goes in both", "goes in neither". During the final stages of experimentation, the model was run as many times as possible to test the best possible distribution of variables between the models.

4.3.3.3 Hyperparameter tuning

As introduced in *Section 4.3.*2, the introduction of some hyperparameters was performed in order to optimize the model performance and avoid overfitting.

Between these hyperparameters, a fundamental one to define was the number of training epochs, which represents the number of full training cycles through all the samples in the training dataset (Géron, 2019). This is important because it could affect the accuracy and computational efficiency of the process: a low number of training epochs may result in underfitting, while an excessive number of training epochs may result in overfitting. Moreover, the increase of training epochs increases the computational time needed.

Another important parameter was the learning rate (LR), which represents the width of the steps that the function took during the backpropagation phase: if the LR was too low, the risk would be getting a very slow convergence and, potentially, overfitting because the model was tuned in a very fine way and took into account noise in the data as well; while if it was too high the model would have poor generalization ability and it would be harder to learn the complex features that a deep learning model is expected to get. To avoid these effects, a LR scheduler, which is a tool designed to adjust the LR as the model is trained, was applied: starting from a higher value, if a certain metric was not verified the LR got reduced by a predefined factor. In this case, the function Reduce on Plateau was used and applied to the validation loss: therefore, if the validation loss did not decrease for a predefined number of epochs (called *patience*), the LR was multiplied by a factor to reduce its value. The initial LR was set to a moderate value, to ensure a good balance between stability and speed. Since usually typical values of LR are in the range between 0.01 and 0.0001, the intermediate 0.001 was chosen.

Finally, the other hyperparameter that was tuned in this procedure was the dropout rate, defined in *Section 4.3.3*.

During the tuning phase, each parameter was tested individually: first the dropout rate was tested fixing the others, and different values were tried until the R^2 started decreasing; then the patience was tested keeping the determined value of dropout rate and so on, until reaching the highest value of R^2, which indicated that the model reached its best performance.

Along with the parameters discussed above, hyperparameter tuning involved the number of neurons in the hidden layer (in the code called *hidden_size*) for both models: since the input features and the parameters to be calculated were different, the ideal value of *hidden_size* was supposed to be different for the two ANN. For this parameter, the research was performed based on 9 values, which are typical of ANN of different complexity levels: stating from the very basic and simple 8 neurons, the value was doubled until reaching 2048, which is typical of very complex

Convolutional ANN. The possible values considered were, therefore, 8, 16, 32, 64, 128, 256, 512, 1024, 2048.

The tuning algorithm used was a nested double loop, in which training and validation was performed fixing a value of *hidden_size* for the network computing K and iterating through all the possible values for the second network, and then updating the number of neurons of the first model and repeating the same process. The choice of the best values of *hidden_size* for each network was based on the combination which allowed to get the lowest validation loss.

4.3.3.4 Training and validation

The observation of training and validation loss trends gave insights not only on how the model performed at each iteration (and, therefore, how well it was fitting the given data), but also on the occurrence of underfitting or overfitting. This was a crucial point, since the goal was to find a model able to fit not just the data that were given, but that can also be generalized and applied in different contexts with different sets of data. If the model performed too well on the training data, it might become too context specific and may not be able to give more general results. This is why the dataset was not split only between training and test, but a validation set was also considered, to observe how the model is really performing. Generally, the performance on the training set was expected to be higher (lower training loss) than the validation, but they should follow a similar trend: a loss peak was expected to be observed at the beginning of the training, when the model had not learned the patterns yet, and then it should have decreased throughout the training epochs, until reaching a minimum. If the minimum was not reached, it probably meant that the model had not converged yet and that it should have been trained for longer (Géron, 2019).

The training and validation of the model consisted in running the ANN for the two datasets previously defined (corresponding respectively to the 60% and 20% of the complete dataset) and finding values for the two parameters (K and α). As stated in the objectives of the study, there was a fundamental difference between the usual approach with this kind of model and the one taken for this study: the training loss is usually computed by backpropagating the results to the initial inputs and adjusting the weights of the connections between neurons until a minimum (local or absolute) of the loss function is reached, while in this case the loss function was calculated relatively to the value of SOM predicted, inserting the computed parameters in the formula previously defined, which represented the process-based model. The most effective loss function was determined to be the Mean Squared Error function, applied between the measurement of SOM at the instant (t+Δt) and the predicted value with the ANN:

$$MSE = \frac{1}{n} * \sum_{i=1}^{n}\left(SOM(t+1)_i - SOM_{pred,i}\right)^2. \quad \text{(Eq. 14)}$$

Once the training loss was computed, the backpropagation was performed: an optimiser adjusted the weights of the connections between neurons with the aim to minimize the loss function, which was a measure of the distance between the measurements and the predicted values. In this case, the optimiser that was used is the Adaptive Moment Estimation (Adam), which is a very popular choice in deep learning algorithms.

After training, the same procedure was repeated with the validation dataset. This step was important to verify the actual performance of the model: using a different dataset compared to the training one, it was possible to observe if the fitting ability was similar or if it decreased. In the latter case, it would mean that the model was overfitting the training data, therefore obtaining good results for that dataset, but that it would not be able to produce generalized results. From the comparison between training and validation loss trends, information on the goodness of the model and the hyperparameter tuning could be extracted.

4.3.3.5 Computation of parameters

Once all the parameters were set and the model had its final configuration, after training and validation, the model could be applied to the test dataset (20% of initial database). The values of K and α were computed applying the model to each of the 142 observations contained in the test set, and their final value was defined as the average of these 142 results. Then, the final values of SOM were predicted using the formula previously defined (*Eq. 6*). As a simplification of notation, in the code the parameter referring to the carbon input (K in the equations in this thesis) was called "*a*", while the mineralization rate (α in the equations) was defined as "*b*", but in this text they were always referred to as K and α.

The carbon input parameter (K) is a variable expressed as a mass percentage of SOM in soil $kg_{SOM}/(100\ kg_{soil}\cdot y)$: it represented the external carbon inputs (from livestock and vegetation) introduced in the period Δt, as a mass percentage of the soil portion considered. It is important to remember that this value should always be strictly greater than zero, which is why the softplus output activation function was applied on the model. The unit of measure of K obtained could be transformed from $kg_{SOM}/100\ kg_{soil}$ into t_{SOM}/ha, to compare the result to other studies computing the SOM content of soils, with the relationships below

$$m_{SOM}\ [t] = m_{som}[kg]\cdot 10^{-3}, \qquad \text{(Eq. 15)}$$

$$m_{soil} = \rho_{soil}\cdot V_{soil} = \rho_{soil}\cdot h_{sample}\cdot A \rightarrow A\ [m^2] = \frac{m_{soil}}{\rho_{soil}\cdot h_{sample}}, \qquad \text{(Eq. 16)}$$

$$A[ha] = A[m^2]\cdot 10^{-4}. \qquad \text{(Eq. 17)}$$

Eq. 15 was used to transform the value of K from kg into tonnes, *Eq. 16* was the change of unit of the denominator (from 100 kg into an area), and *Eq. 17* was used to multiply the result of *Eq. 16* to obtain the area in hectares.

This result could be further transformed to express the tonnes of SOC effectively introduced yearly, by multiplying the result by the van Bemmelen factor

$$\frac{t_{SOC}}{ha \cdot y} = \frac{t_{SOM} \cdot 0.58}{ha \cdot y}. \qquad \text{(Eq. 18)}$$

The mineralization rate (α) is a parameter which expressed the percentage of SOM that was mineralized (i.e., converted into CO_2) with respect to the initial SOM content in the time interval Δt, therefore it had unit of measure [y^{-1}], and its value was always between 0 and 1. As previously mentioned, usual values of mineralization are generally much lower than 1, so a customized output activation function was applied to the model to limit the results to a maximum of 0.3.

This parameter could be used to assess the tonnes of SOC mineralized per year, by multiplying the average SOM content computed from the observations in the database and applying the same transformation factors defined for K.

$$SOM_{mineralized} = \overline{SOM} * \alpha. \qquad \text{(Eq. 19)}$$

The value obtained applying *Eq. 19* was transformed in SOC with the van Bemmelen factor, and then subtracted to the carbon input previously defined, thus obtaining the yearly SOC accumulated in the studied pasture type.

$$SOC_{accumulated} = SOC_{input} - SOC_{mineralized}. \qquad \text{(Eq. 20)}$$

4.3.3.6 Model performance

The assessment of the performance of the model was fundamental to give an idea of the reliability of the results and the possibility of improvement. During training and validation, the Mean Squared Error loss function was the metric considered to assess the goodness of the model and to decide the best configuration to apply to the test set.

In the test phase, the model's performance was measured in two main ways. The first metric of evaluation was the test loss, so the application of the same loss function applied during training and validation (Mean Squared Error function). At each iteration, the test loss was expected to be very similar or slightly lower than the validation loss.

The second metric to assess model performance, used in the test phase, was the computation of the coefficient of determination R^2, which is very a common way to observe the goodness of regression tasks thanks to its simple interpretation: it represents the fraction of variance of the dependent variable that is explained by the independent variables, and it can only get values between 0 (the variance of the dependent variable is not related to the independent variables, so there is no possibility of predicting it) and 1 (the dependent variable is perfectly explained by the independent variables). Getting values as close to 1 as possible is a sign of good fitting.

5. Results and discussion

The following paragraphs present the results obtained by following the workflow described in *Section 4.3.3*. Along with these results, a discussion on the meaning of the displayed tables and figures was performed.

5.1 Linear regression

Using the script in *Section 9.1*, the linear regression of the dataset was performed. From the training phase, the parameters of the linear model (slope m and intercept q) were computed, and the equations presented in *4.3.1* (*Eq. 7, 8,* 9) were applied to transform them into values of K and α. The slope of the linear regression was computed to be 0.755, while the angular coefficient resulted in the value 0.633.
Figure 9 shows the scatter plot of SOM(t+Δt) estimated versus SOM(t+Δt) measured, and the ideal linear trend using the computed parameters.

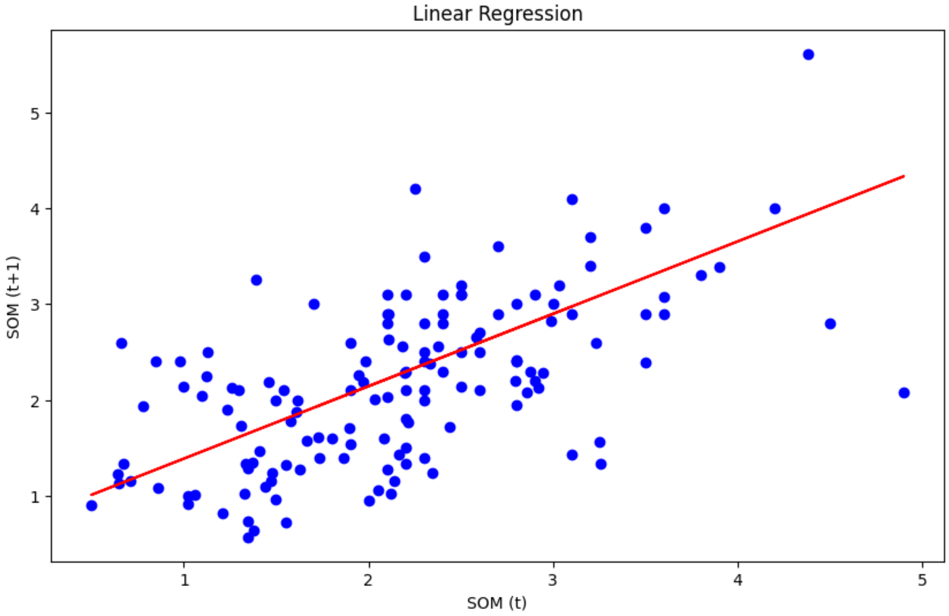

Figure 9 - Plot of the results of the linear regression: the scattered blue points represent the measurement of SOM(t+Δt) performed on field, while the red line represents the ideal values they would get if they followed the linear model.

From the application of this model and of the equations described in *Section 4.3.1* (*Eq 8, 9, 10*), the values that were obtained were:

$\alpha = -\frac{\ln(m)}{\Delta t} = 0.423,$

$$K = \frac{q \cdot \alpha}{1 - e^{-\alpha \cdot \Delta t}} = 1.091,$$

$$R^2 = 1 - \frac{\sum(SOM(t+1)_{test} - \overline{(SOM(t+1)_{test})}\,)^2}{\sum(SOM(t+1)_{test} - SOM(t+1)_{pred})^2} = 0.33.$$

It can be observed that the results of the linear regression are not consistent with real conditions: more specifically, the mineralization rate in this kind of soil is expected to be around 19% (Teixeira et al., 2011), while in this case it is evaluated at 42%, which is a unrealistic value. Moreover, the value of R^2 obtained through this procedure is relatively low compared to other studies related to SBPPRL (Teixeira et al., 2011), showing a limited linear correlation between SOM(t) and SOM(t+Δt). This is clearly linked to the more complex interactions that regulate the change in organic matter content of soil, including climate related factors, external inputs and natural processes, which a simple linear equation is not able to describe correctly. However, the application of linear regression gave a baseline value to assess whether the hybrid model was working correctly or not: expected valued would be significantly higher than the ones obtained with linear interpolation.

5.2 Variable selection

5.2.1 Before hyperparameter tuning

As described in *Section 4.3.3.2*, the selection of variables was performed using a random assignment function in two different phases of model development: the first without specific optimization hyperparameters and the second with the introduction of tuned hyperparameters.

The first stage of this research was aimed at finding a fixed input configuration to allow hyperparameter tuning, which was selected based on 100 iterations of the model. The results of the best iteration out of these 100 are showed in *Table 3*:

Table 3 - Results of the iteration with the highest value of R^2, which is the one that was used for the following tests on the other model parameters.

hidden size f	hidden size g	validation loss	test loss	R^2	K	α
64	128	0.282	0.491	0.381	0.506	0.249

From *Table 3*, it is possible to observe that, even without parameter tuning, the performance of the hybrid model is already higher than the one obtained with simple linear regression ($R^2 = 0.38$ compared to the one of linear regression which had $R^2 = 0.33$).

Out of the 100 iterations, the best configuration of parameters was the one summarized in *Table 4*:

Table 4 - Summary of the random variable assignment for the two models (model f for K, model g for α) in the best configuration found after the first 100 runs of the code, before the introduction of hyperparameters.

	Model f	**Model g**
B1 (t)		
B2 (t)	x	
B3 (t)	x	x
B4 (t)	x	x
B5 (t)	x	
B6 (t)		x
B7 (t)		
B8 (t)	x	x
B8A (t)		
B9 (t)		x
B11 (t)		x
B12 (t)	x	x
B1_close (t)	x	
B2_close (t)	x	
B3_close (t)		
B4_close (t)		x
B5_close (t)		x
B6_close (t)		
B7_close (t)		x
B8_close (t)	x	x
B8A_close (t)	x	x
B9_close (t)		x
B11_close (t)		
B12_close (t)		x
clay		x
sand	x	
silt	x	x
phh2o	x	x
SoilMoi0_10cm_inst (t)	x	x
SoilMoi (t+1)		x
SoilTMP0_10cm_inst		x
SoilTMP (t+1)	x	x
NDVI (t)		x
NDWI (t)		x
SR (t)		x

	Model f	**Model g**
SAVI (t)	x	x
OSAVI (t)		
NDVI_close (t)		
NDWI_close (t)	x	x
SR_close (t)	x	x
SAVI_close (t)	x	x
OSAVI_close (t)	x	
day	x	
lab		
chili	x	
dem	x	x
landforms		x
mTPI	x	x
topoDivers	x	
lon	x	
lat	x	x
TOTAL	27	32

5.2.2 After hyperparameter tuning

After the introduction and tuning of hyperparameters (described in *Section 5.3*), performed with the fixed variable configuration previously discussed (*Table 4*), the model was once again run for 100 times, reintegrating the random variable selection already implemented to find the configuration which could give the best results with a fixed set of hyperparameters, at least on the limited number of iterations that was performed. The result is a set of 100 different configurations, in which both the number and the type of selected variables for each model are changing. *Figure 10* shows how the number of input variables in each of the two models is distributed over the iterations.

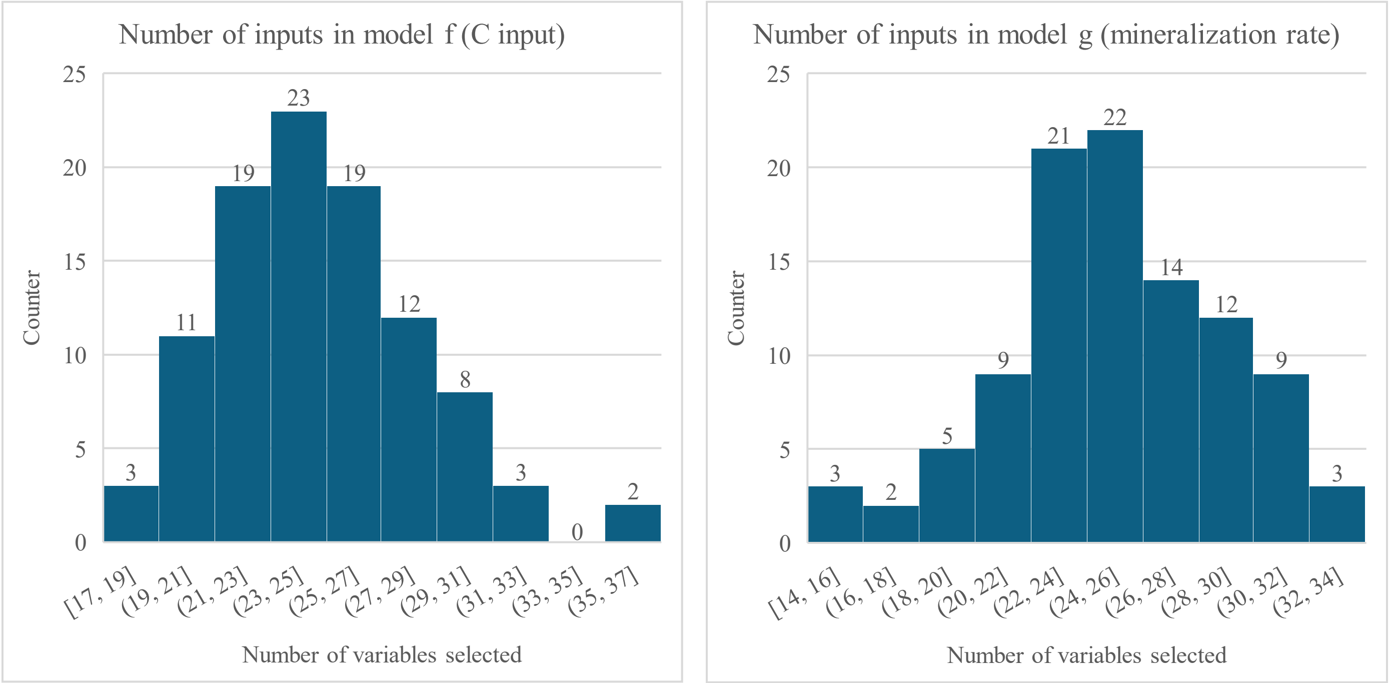

Figure 10 - Histograms representing the number of input variables selected for model f (left) and model g (right) over the 100 iterations of the code

Figure 10 shows that the distribution of the number of variables in the two models was quite similar: the minimum number of variables assigned to model f was 17, while for model g it was 14; the maximum number of variables assigned was 36 for model f and 34 for model g. In both distributions the majority of iterations assigned a number between 20 and 30 variables, with a peak in the middle (around 25). This shows that the random variable assignment was successful in the goal of reducing the amount of data to process in each model, which was an issue in the performance.

In *Figure 11*, it is possible to see the number of iterations in which each of the 51 variables in the database were assigned to each of the two ANNs.

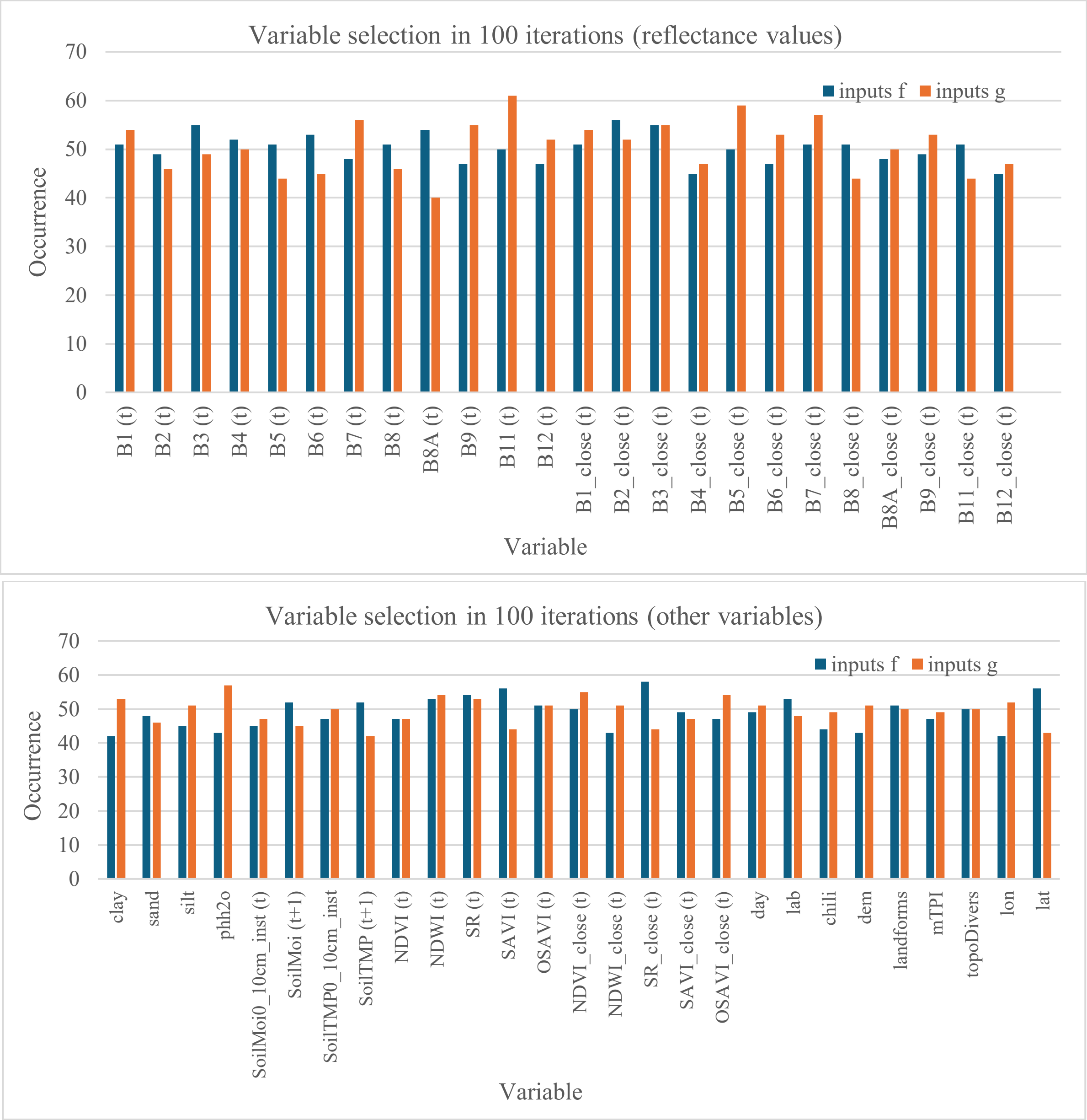

Figure 11 - Comparison between the assignment of each variable in model f (blue) and model g (orange). In the graph above, the histogram is related to the measurement of radiometric reflectance, while in the graph below all the other variables are considered.

The random selection of variables did not follow any theoretical and physical principle, as there were no specific indications that a variable could be more representative for the computation of any of the two parameters. As *Figure 11* shows, the variables were distributed in a regular way, in both ANNs: each of them was assigned at least in 40% of the iterations (which is the case of *B8A (t)* in model g), and at most in 61% of the cases (*B11(t)* in model g), and all values are around 50%.

In model f, the variables with lowest occurrence are *clay* and *lon*, with 42% of presence, while the most commonly used is *SR_close (t)*, which appears in 58% of the iterations.
After hyperparameter tuning and the following 100 final runs of the code, the distribution of variables which gives the best results is displayed in *Table 5*:

Table 5 - Summary of the random variable assignment for the two models (model f for K, model g for α) in the best configuration found after the final 100 runs of the code, after the introduction and tuning of hyperparameters.

	Model f	**Model g**
B1 (t)		
B2 (t)		x
B3 (t)	x	
B4 (t)		x
B5 (t)		x
B6 (t)	x	x
B7 (t)	x	x
B8 (t)		
B8A (t)		x
B9 (t)		
B11 (t)	x	x
B12 (t)	x	x
B1_close (t)		
B2_close (t)	x	x
B3_close (t)	x	
B4_close (t)		x
B5_close (t)		x
B6_close (t)	x	
B7_close (t)		x
B8_close (t)	x	
B8A_close (t)	x	x
B9_close (t)		x
B11_close (t)		
B12_close (t)	x	x
clay	x	
sand		
silt	x	x
phh2o		
SoilMoi0_10cm_inst (t)	x	
SoilMoi (t+1)		x
SoilTMP0_10cm_inst		

	Model f	Model g
SoilTMP (t+1)	x	
NDVI (t)		x
NDWI (t)	x	
SR (t)		x
SAVI (t)	x	x
OSAVI (t)		x
NDVI_close (t)		x
NDWI_close (t)	x	
SR_close (t)	x	
SAVI_close (t)	x	
OSAVI_close (t)		
day		x
lab	x	x
chili	x	x
dem		
landforms	x	
mTPI	x	
topoDivers		x
lon		
lat	x	
TOTAL	25	26

The configuration in *Table 5* represents the distribution of variables that can be fixed to apply the final model to a different dataset, or to make future projections of SOM content in the current dataset. The results obtained through this variable selection were more reliable compared to the assignment of the full list of variables contained in the database (as discussed in *Section 4.3.3.2*), but the method still has a few key downsides:

- The number of variables assigned to each parameter is still very high with respect to the measurements, increasing the risk of overfitting.
- The research of an ideal configuration of variable assignment is extremely long and complex, as the possible combinations of distribution of 51 variables over 4 categories (i.e., “goes in f”, “goes in g”, “goes in both”, “goes in neither”) are impossible to consider in their totality.

For future work, a more theoretical approach can be tried, or a more extensive application of the random selection which could give more comprehensive results is suggested. Another possible solution that could be considered would be to assign the variables according to theoretical principles and the correlation between variables, but the random assignment was still preferred for its easier implementation.

5.3 Hyperparameter tuning

The best parameters to apply to avoid overfitting (dropout rate, patience and factor of the scheduler and number of training epochs) are defined by trial-and-error procedure as described in *Section 4.3.3.3*, using the best variable configuration defined in the previous paragraph (*Table 5*). The results of this research are showed in *Table 6:*

Table 6 - Results of different code runs with the same variable distribution (defined in Table 5), to determine the optimal parameters to avoid overfitting. The final choice of hyperparameters, determined through the highest value of R^2, is highlighted.

dropout rate	patience	factor	epochs	hidden size f	hidden size g	validation loss	test loss	R^2	K	α
0.5	50	0.1	1000	64	8	0.391	0.441	0.396	0.649	0.227
0.6	50	0.1	1000	128	8	0.388	0.443	0.393	0.645	0.235
0.7	50	0.1	1000	128	8	0.388	0.428	0.413	0.633	0.239
0.8	50	0.1	1000	128	8	0.387	0.424	0.419	0.649	0.245
0.9	50	0.1	1000	128	1024	0.391	0.403	0.448	0.670	0.254
0.9	50	0.2	1000	2048	8	0.398	0.456	0.375	0.667	0.196
0.9	50	0.3	1000	2048	8	0.394	0.477	0.346	0.667	0.191
0.9	50	0.4	1000	512	8	0.395	0.448	0.386	0.623	0.207
0.9	50	0.5	1000	512	8	0.394	0.456	0.374	0.640	0.199
0.9	50	0.1	1000	1024	2048	0.393	0.408	0.441	0.668	0.262
0.9	20	0.1	1000	2048	2048	0.388	0.394	0.459	0.743	0.279
0.9	40	0.1	1000	2048	2048	0.390	0.397	0.456	0.692	0.272
0.9	60	0.1	1000	512	8	0.394	0.461	0.367	0.650	0.187
0.9	80	0.1	1000	1024	16	0.388	0.468	0.359	0.630	0.216
0.9	30	0.1	1000	256	1024	0.392	0.404	0.446	0.677	0.250
0.9	20	0.1	1000	2048	2048	0.381	0.388	0.468	0.774	0.285
0.9	20	0.1	2000	2048	2048	0.386	0.391	0.464	0.745	0.285
0.9	20	0.1	3000	2048	2048	0.386	0.390	0.466	0.772	0.284
0.9	20	0.1	4000	2048	2048	0.384	0.387	0.470	0.759	0.283
0.9	20	0.1	5000	512	2048	0.383	0.392	0.462	0.720	0.270
0.9	20	0.1	6000	1024	2048	0.386	0.394	0.459	0.727	0.271

It is possible to observe the improvement derived by this parameter tuning: the values of R^2 in the first tuning iterations were lower than 0.4, while after the tuning they arrived up to 0.47.

During the testing of the parameters, the values of training and validation loss for each epoch were plotted to observe the evolution of the model and the effect of changing hidden layer size on the fitting. As it can be observed in *Table 6*, the best combination of hyperparameters included a dropout rate of 0.9, a LR reduction factor of 0.1, which is applied with a patience of 20 training epochs. The number of training epochs that gives the best result was 4000.

As introduced in *Section 4.3.3.3*, the selection of the number of neurons contained in the hidden layers of the two ANNs was the main action of hyperparameter tuning performed in the final iterations of the model: the other hyperparameters (dropout rate, number of epochs, factor and patience of the LR scheduler) were tuned separately and then fixed.

The number of neurons in the hidden layer, instead, was studied at each iteration, since it is one of the most influential parameters on the performance of a ANN. Another possibility to increase the performance would have been increasing the number of intermediate layers, but for simplicity it was decided to keep both models with a single hidden layer. As previously introduced, the choice of the number of neurons was made based on the combination of *hidden_size_f* and *hidden_size_g* which allowed to get the lowest validation loss, and then were applied on the test set fixing the selected dimensions. The results of the 100 final iterations are showed in *Figure 12*:

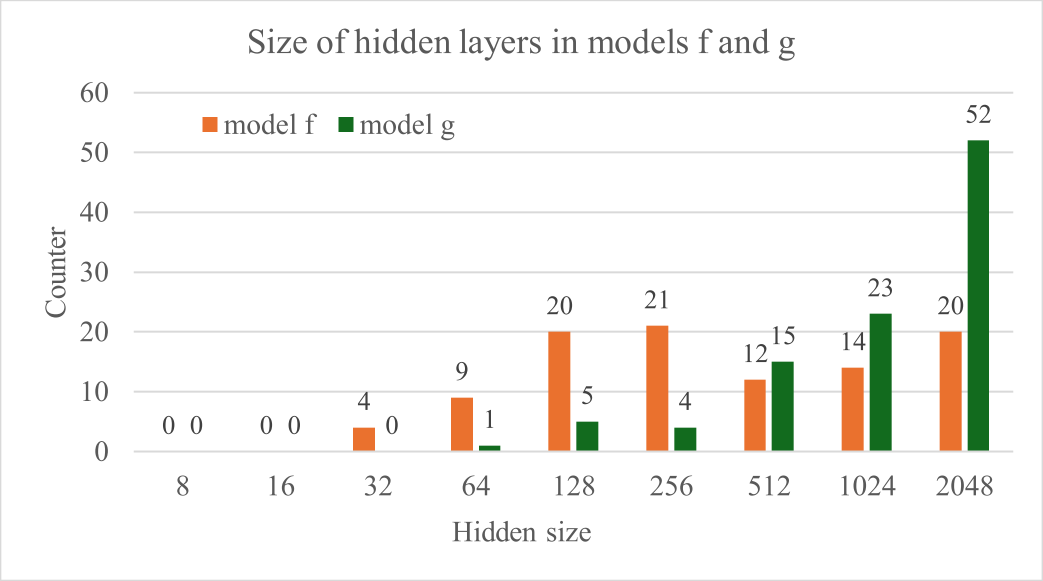

Figure 12 - Comparison between the hidden layer sizes chosen for model f (orange) and model g (green) over the 100 iterations

Results depicted in *Figure 12* and the values of hidden size obtained during the hyperparameter tuning (*Table 6*) show that, at the beginning of the tuning, the best results were obtained for very simple models with few neurons: the most common values were 64 and 128 for model f, while for model g the best number of neurons resulted almost always to be 8, which was a sign that the model was underfitting and, therefore, was obtaining worse results. Then, the lowest validation loss started being reached setting both models to very high number of neurons (sometimes both were set at 2048 neurons in the hidden layer), which was likely a sign of overfitting.

After the tuning and the final 100 iterations, it was possible to observe that the simplest configurations were always discarded for both models (there are no cases where the ANN have 8 or 16 neurons in the hidden layer, and very few cases in which 32 or 64 neurons are considered). Moreover, the values for model f stabilised in the intermediate values (even though it is quite evenly distributed between all the considered sized between 128 and 2048 neurons), while model g was set at 2048 neurons for over 50% of the iterations, and anyway in 90% of the cases the hidden layer of model g was set to a high value (from 512 neurons).

5.4 Training and validation

The trends of training and validation during the choice of the best number of neurons for the hidden layer were plotted and saved at each iteration, to observe how the distance between the two varies at different model configurations and how hyperparameter tuning affects fitting. The parameters observed, as introduced in the *Section 4.3.2*, were the dropout rate, the patience and factor of the LR scheduler and the number of training epochs. The research of the best hyperparameters was performed with a fixed configuration of variable selection (*Table 4*) and started from standard values (dropout rate = 0.5, patience = 50, factor = 0.1, epochs = 1000) which were varied until the highest value of R^2 in the test set was found. The figures below can be used to make observations about how changing these parameters the trends of training and validation loss are modified.
Figure 13 and *Figure 14* show respectively the plots referred to the initial condition that was considered during tuning (dropout rate = 0.5, LR factor = 0.1, LR patience = 50, training epochs = 1000) and the final condition at the end of the tuning process (dropout rate = 0.9, LR factor = 0.1, LR patience = 20, training epochs = 4000).

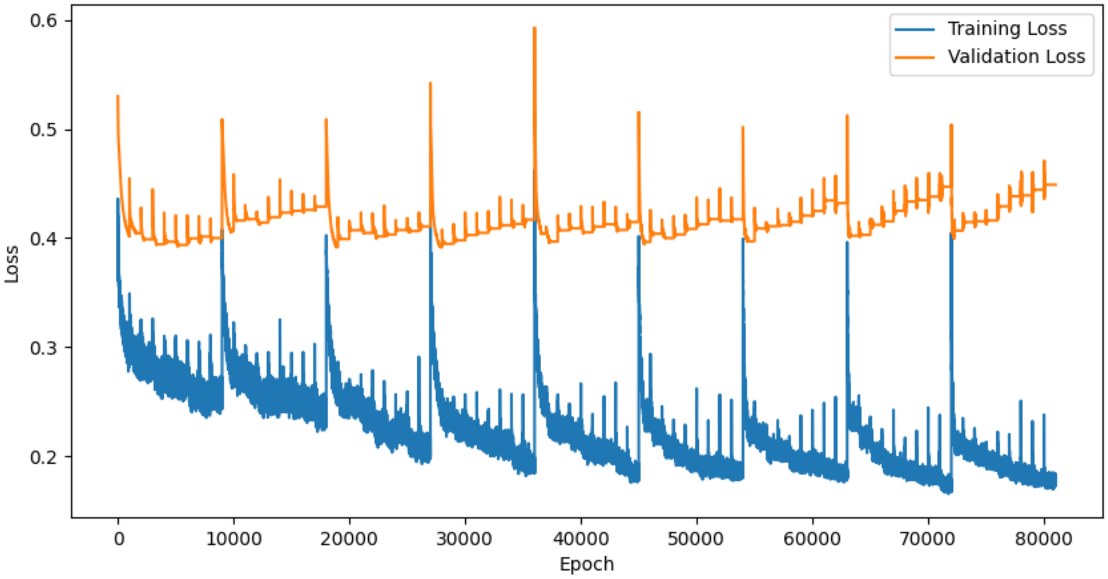

Figure 13 - Plot of training and validation loss at the initial conditions of hyperparameter tuning.

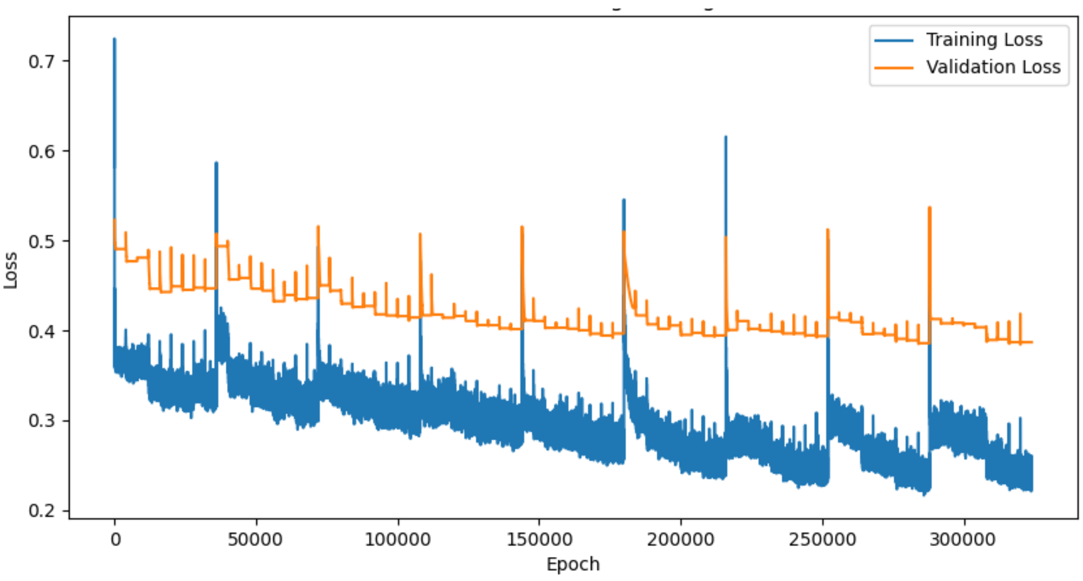

Figure 14 - Plot of training and validation loss at the final condition of hyperparameter tuning.

Comparing the pictures, the improvement of model fitting was clear. In *Figure 13*, the training loss followed a descending trajectory, and it reached very low values of loss (lower than 0.2), indicating good performance, but the validation loss was not only much higher, but it also seemed to be increasing as the training epochs increased. This highlighted two aspects: the number of training epochs was not high enough, because the validation loss did not reach a stability, and model was clearly overfitting the data, because it was able to reach very good fitting on the training set but with different data the model had a much lower performance.

After the hyperparameter tuning (*Figure 14*), there still was a gap between training and validation loss, but the two followed a more similar, descending trend. The performance during training was slightly lower than the one observed with initial conditions (the minimum is higher than 0.2), but the lower distance between training and validation and the regular trend indicated the ability of the model to keep adequate performance with different datasets without overfitting.

5.5 Computation of parameters

5.5.1 Carbon input

As previously described, the carbon input K was evaluated at each observation in the test set (142 observations). Then, for each iteration, the average value between the 142 was computed, to obtain the parameter used in the equation.

The results related to all the values obtained and their averages after the final 100 iterations are displayed in the following table and figures.

Table 7 - Summary of the main features of the values assigned to the parameter K after 100 iterations.

K max	4.277
K min	3.9E-17
K average	0.769
standard deviation	0.687
K < 0.1 (count)	2792
K < 0.1 (%)	19.662
K > 1 (count)	4239
K > 1 (%)	29.852

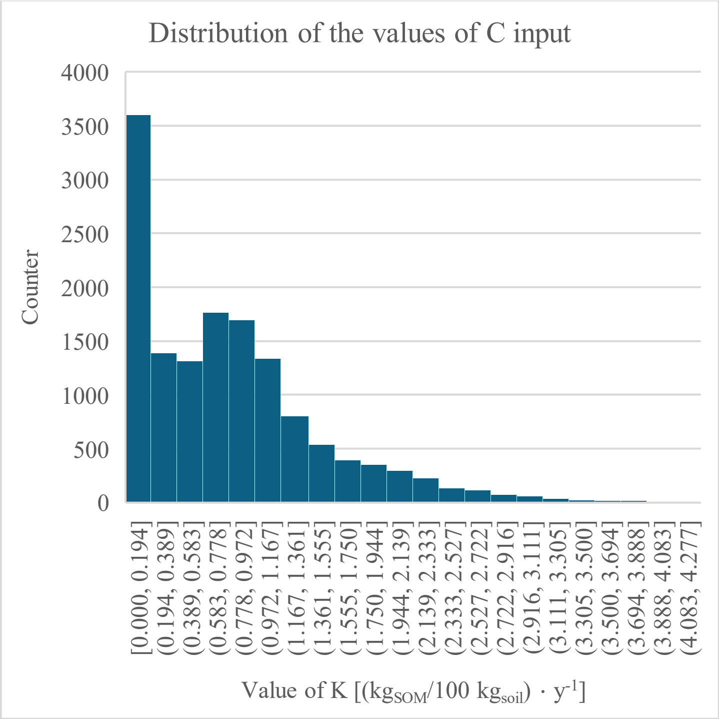

Figure 15 - Histogram containing information on all the values to which the parameter K was fit during the test phase (142·100 values in total)

In *Table 7* and *Figure 15* it was possible to observe a few properties of the values assigned to the parameter K during the testing phase, at each observation of each iteration (for a total of 14200 estimations of K). The possible values, are ranging in a fairly wide interval, with minimum very close to zero (3.9·10^{-17} $kg_{SOM}/(100\ kg_{soil}\cdot y)$) and maximum equal to 4.277 $kg_{SOM}/(100\ kg_{soil}\cdot y)$. However, it can be clearly deduced from the histogram that in the majority of cases, the values of K are well below 1 (around 20% of all the obtained values is even lower than 0.1). This is also confirmed by the fact that, even though the maximum is much higher than 1, the average value still stays below (0.7694 $kg_{SOM}/(100\ kg_{soil} \cdot y)$). The standard deviation indicates a moderate level of spreading around the mean value. *Figure 16* shows more clearly these described features through a box-and-whiskers plot.

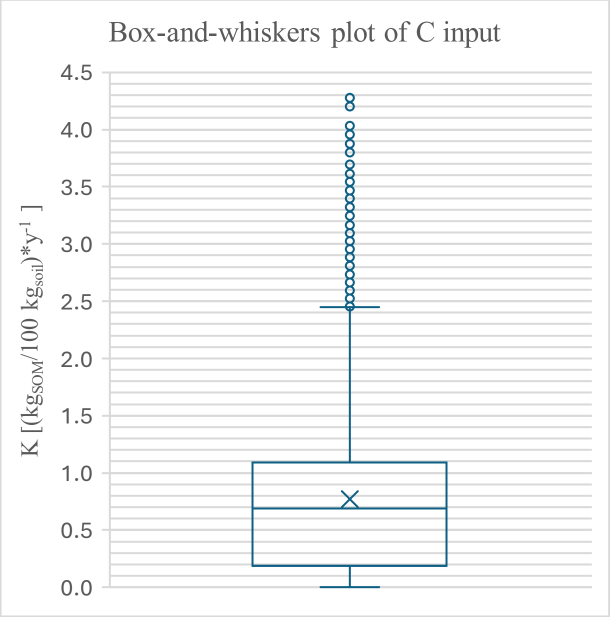

Figure 16 - Box-and-whiskers plot containing information on all the values to which the parameter K was fit during the test phase

The plot in *Figure 16* confirmed the previous observations: the Interquartile Range (IQR), containing the 50% of the observations, was between the values 0.3 and 1.1, with the median at 0.7. The lower whisker is extended until 0, while the upper one ended at around 2.45. All the values above were considered outliers.

Finally, *Figure 17* represents the plot of the values of K obtained by averaging the 142 test values for each of the 100 iterations.

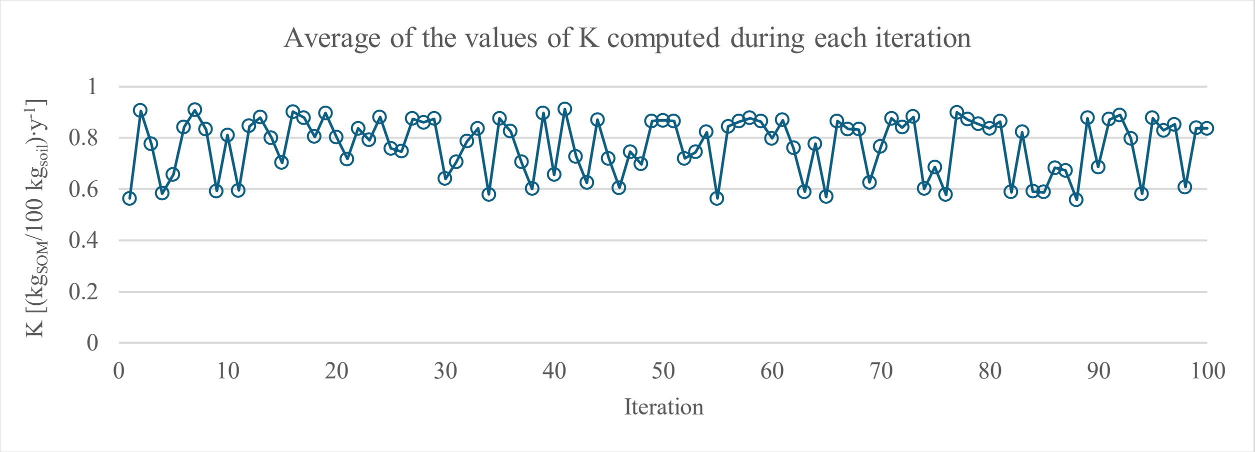

Figure 17 - Plot of the average value of K computed in the test phase during the 100 iterations.

Figure 17 is a final confirmation of the previous observations: averaging the 142 values obtained in each iteration, the final value of K is always included between 0.55 $kg_{SOM}/(100\ kg_{soil} \cdot y)$and 0.9 $kg_{SOM}/(100\ kg_{soil} \cdot y)$, never going over the unit. The value of K in the iteration with the highest R^2 (best performance) is 0.8797 $kg_{SOM}/(100\ kg_{soil} \cdot y)$.

Considering the value of K obtained in the best iteration and the equations defined in *Section 4.3.3.5*, a few observations of physical interpretation can be made.

Every year, the soil receives an external C input from vegetation and livestock of 0.8797 kg$_{SOM}$ every 100 kg$_{soil}$. Assuming the following statements:

- The bulk density (ρ_{soil}) of cropland soil in Europe is estimated to be 1200 kg/m^3 (rounding the average value obtained in Panagos et al. (2024)),
- The sampling depth (h$_{sample}$) in the campaigns from which data was taken is 0-20 cm (Morais et al., 2023), according to the reference depth used for other soil datasets (i.e., LUCAS database), as described in Orgiazzi et al. (2017). An average of this depth interval (10 cm) can be used for the transformation of unit of measure (*Eq. 16*), the C input value con be transformed into tonnes per hectare (per year) as follows, according to *Eq. 15, 16, 17* and *18*:

$$0.8797\ kg_{SOM} = 0.8797 \cdot 10^{-3}\ t_{SOM},$$

$$A = \frac{m_{soil}}{\rho_{soil} \cdot h_{sample}} = \frac{100\ kg}{1200\frac{kg}{m^3} \cdot 0.1\ m} = 0.83\ m^2 = 8.3 \cdot 10^{-5}\ ha,$$

$$0.8797 \frac{kg_{SOM}}{100\ kg_{soil}} = \frac{0.8797 \cdot 10^{-3}}{8.3 \cdot 10^{-5}} \frac{t_{SOM}}{ha} = 10.6 \frac{t_{SOM}}{ha},$$

$$t_{SOC} = 10.6 \frac{t_{SOM}}{ha} \cdot 0.58 = 6.1 \frac{t_{SOC}}{ha}.$$

The obtained value of annual C input per hectare can be compared to the value computed in Teixeira et al. (2011), which stated that the annual C input for SBPPRL would be 0.64 pp (i.e. 0.64 kg_{SOM}/100 kg_{soil}). Applying the same transformations (*Eq. 15, 16, 17, 18*):

$$0.64\ kg_{SOM} = 0.64 \cdot 10^{-3}\ t_{SOM},$$

$$A = \frac{m_{soil}}{\rho_{soil} \cdot h_{sample}} = \frac{100\ kg}{1200 \frac{kg}{m^3} \cdot 0.1\ m} = 0.83\ m^2 = 8.3 \cdot 10^{-5}\ ha,$$

$$0.64 \frac{kg_{SOM}}{100\ kg_{soil}} = \frac{0.64 \cdot 10^{-3}}{8.3 \cdot 10^{-5}} \frac{t_{SOM}}{ha} = 7.71 \frac{t_{SOM}}{ha},$$

$$t_{SOC} = 7.71 \frac{t_{SOM}}{ha} \cdot 0.58 = 4.47 \frac{t_{SOC}}{ha}.$$

The value of C input obtained in this study was higher than the one obtained by Teixeira et al. (2011) and, in general, than the expected input values in grasslands, which were estimated to a maximum of 5 t_{SOC}/ha·y (Soussana et al., 2004). This could be due to a combination of high C input related to the type of pasture considered and a parameter overestimation from the hybrid model.

5.5.2 Mineralization rate

As well as what was done for carbon input, the mineralization rate α was evaluated at each observation in the test set (142 observations). Then, for each iteration, the average value between the 142 obtained was computed, to obtain the parameter used in the equation.

The results related all the values obtained and their averages after the final 100 iterations are displayed in the following table and figures.

Table 8 - Summary of the main features of the values assigned to the parameter α after 100 iterations

α max	0.3
α min	0.0045
α average	0.2685
Standard deviation	0.0574
α < 0.1 (count)	564
α < 0.1 (%)	3.9718
α > 0.29 (count)	8169
α > 0.29 (%)	57.5282

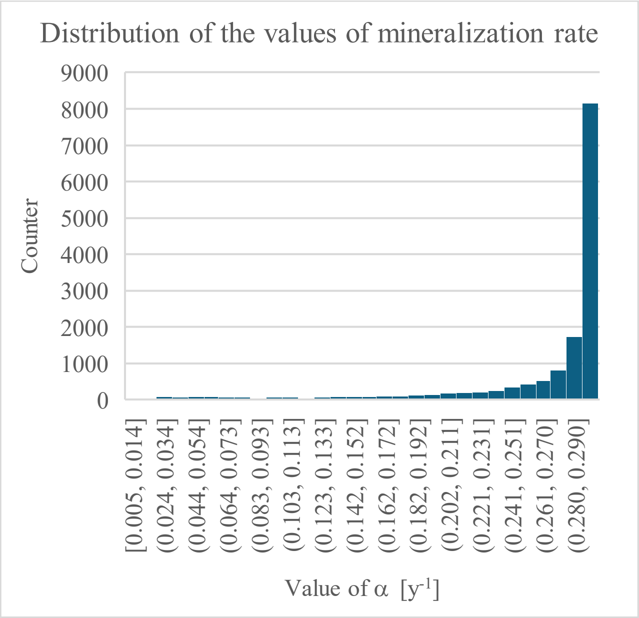

Figure 18 - Histogram containing information on all the values to which the parameter α was fit during the test phase (142·100 values in total)

As showed by *Table 8* and *Figure 18*, the fitting of α usually gave values closer to the upper limit (0.3), with almost 60% of the 14200 values being greater than 0.29 y^{-1}. The maximum value assigned was exactly the upper limit 0.3, while the minimum was close to zero (0.0045 y^{-1}). The average, as expected, was close to the upper limit as well (0.2685 y^{-1}), and the standard deviation is quite low, which meant that most results were clustered around the average. The box-and-whiskers plot in *Figure 19* confirmed these observations.

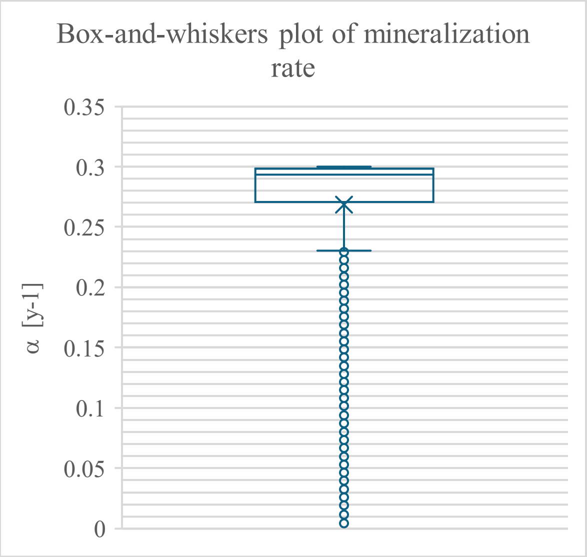

Figure 19 - Box-and-whiskers plot containing information on all the values to which the parameter α was fit during the test phase

Figure 19 shows that the Interquartile Range, containing the 50% of observations, was included between the values of 0.27 y^{-1} and very close to 0.3 y^{-1}, with the median falling above 0.29 y^{-1}. The average was outside the IQR, meaning that there were several values lower than the first interquartile, but still the majority of values fell above the average. This is showed by the fact that the lower whisker is extended up to 0.22 y^{-1}, but there were still many outliers below this point.

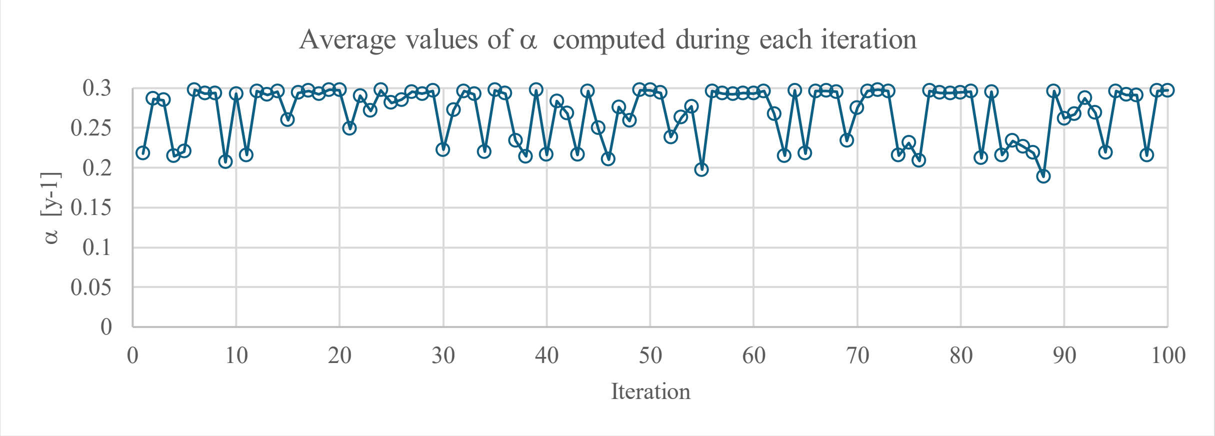

Figure 20 - Plot of the average value of α computed in the test phase during the 100 iterations.

Figure 20 further confirmed the observations written above: in most cases, the average α was very close to 0.3 y^{-1}, there were very few cases in which it fell below 0.25 y^{-1} and only in a couple iterations it was evaluated slightly below 0.2 y^{-1}. The value of α in the iteration with the highest R^2 is 0.2918 y^{-1}.

For the physical interpretation of mineralization rate, it can be observed that the standard values of this parameter are much lower than the ones obtained through the model: natural rates are estimated to be between 1 and 5%, and up to 10-15% in highly controlled environments (Teixeira et al., 2011), while the obtained average is 26% and most values go up to 30%. This is probably related to both the simplicity of the model, which tends to overestimate, and the relatively low number of observations in the dataset. However, it can also be justified by the fact that the SBPPRL considered in the study contain very labile plants, therefore allowing a very fast carbon turnover, and by the fact that the C input parameter seems to be slightly overestimated as well.

Making the same considerations performed for the parameter K, the quantity of SOC mineralized yearly can be computed, as defined in *Section 4.3.3.*5.

The average SOM content measured through the sampling campaigns is 2.13 kg$_{SOM}$/100 kg$_{soil}$, and considering the best mineralization rate of 0.2918 the total mineralized content would be (*Eq. 19*):

$$SOM_{mineralized} = \overline{SOM} \cdot \alpha = 2.13 \frac{kg_{SOM}}{100\ kg_{soil}} \cdot 0.292 = 0.622 \frac{kg_{SOM}}{100\ kg_{soil}},$$

The transformation of unit is the same performed for K (*Eq. 15, 16, 17, 18*):

$$0.621\ kg_{SOM} = 0.621 \cdot 10^{-3}\ t_{SOM},$$

$$A = \frac{m_{soil}}{\rho_{soil} \cdot h_{sample}} = \frac{100\ kg}{1200 \frac{kg}{m^3} \cdot 0.1\ m} = 0.83\ m^2 = 8.3 \cdot 10^{-5}\ ha,$$

$$0.621 \frac{kg_{SOM}}{100\ kg_{soil}} = \frac{0.621 \cdot 10^{-3}}{8.3 \cdot 10^{-5}} \frac{t_{SOM}}{ha} = 7.49 \frac{t_{SOM}}{ha},$$

$$t_{SOC} = 7.49 \frac{t_{SOM}}{ha} \cdot 0.58 = 4.3 \frac{t_{SOC}}{ha}.$$

Considering that the annual SOC input, computed in the previous paragraph, is 6.1 t_{SOC}/ha, the yearly accumulation of SOC in this type of pasture resulted (*Eq. 20*):

$$SOC_{acc} = SOC_{input} - SOC_{mineralized} = 6.1 \frac{t_{SOC}}{ha} - 4.3 \frac{t_{SOC}}{ha} = 1.8 \frac{t_{SOC}}{ha}.$$

The obtained value of yearly carbon accumulation is very similar to the average increase in Sown Biodiverse Pastures, which was estimated to be 1.78 t_C/ha·y (Teixeira et al., 2011). It is important to note that this result is consistent with the ones obtained in the previous study, even though both parameters (K and α) were estimated to be higher. This indicates a tendency of the model to overestimate the parameters, but in a balanced way, which allows to get acceptable fitting results.

5.6 Model performance

Figure 21 shows the trends of validation and test loss observed during the 100 iterations performed.

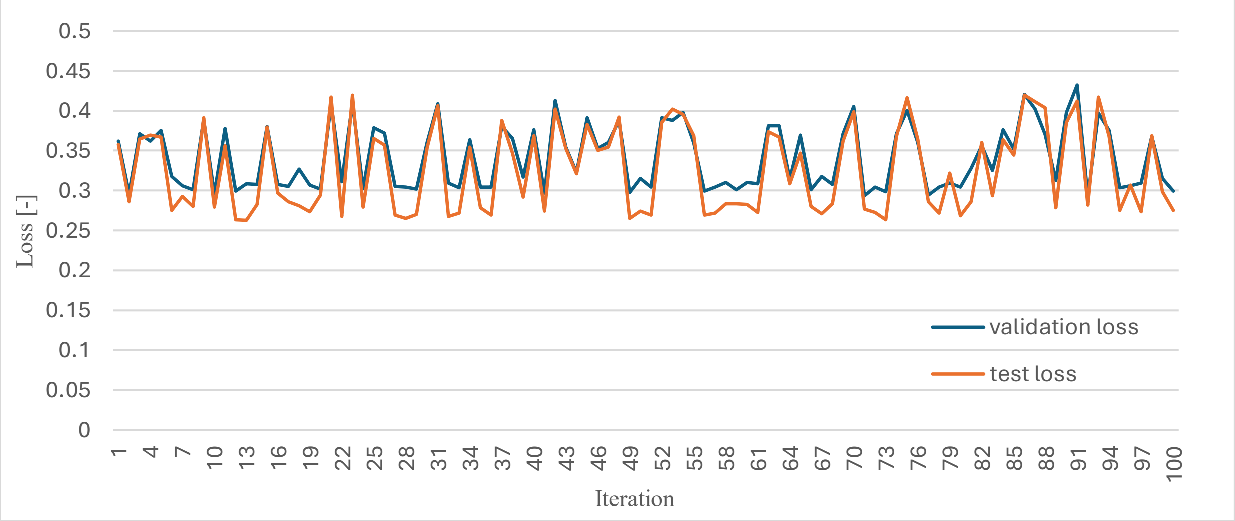

Figure 21 - Plot of the values of best validation loss and test loss observed during the 100 iterations.

As predicted, the test loss (plotted in orange in *Figure 21*), followed a very similar trend to the one observed for the best value of validation loss (the value according to which the size of the hidden layers of the ANN were defined), the first being always almost equal or slightly than the latter. This is a sign of proper training, without any significant underfitting or overfitting.
After 100 iterations, *Table 9* and *Figure 22* summarized the main properties of the obtained values of R^2:

Table 9 - Summary of the main features of R^2 after 100 iterations.

R^2 max	0.64
R^2 min	0.423
Average	0.554
Standard deviation	0.072
$R^2 < 0.5$	30
$R^2 > 0.6$	45

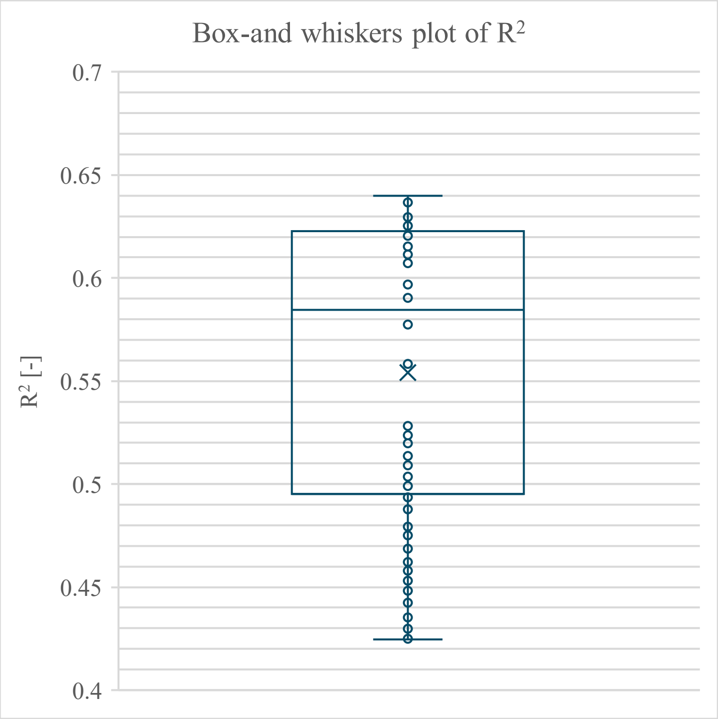

Figure 22 - Box-and-whiskers plot containing information on all the values assumed by R^2 (in each iteration) during the test phase.

From *Table 9* it is possible to observe that the minimum R^2 obtained corresponded to 0.4247, while the maximum is equal to 0.6398, with an average of 0.5541. The standard deviation is fairly low, indicating that the results were mostly clustered around the average, suggesting that the model performs in a consistent way. Out of 100 iterations, R^2 was found to be lower than 0.5, which could be considered the low performance threshold, on the 30% of cases, which is not a low frequency but it's important to notice that, anyway, the coefficient never goes much below 0.5. Moreover, in 45% of the cases the R^2 appeared to be over 0.6.

Figure 23 confirms these observations: the IQR includes values between 0.49 and 0.62, indicating that over 50% of the iterations have an R^2 of 0.5. All the values are contained between the upper and the lower whisker, which affirm the statement that the model performance was quite consistent throughout the iterations. More detail on the values assumed by R^2 during each iteration were plotted in *Figure 23*:

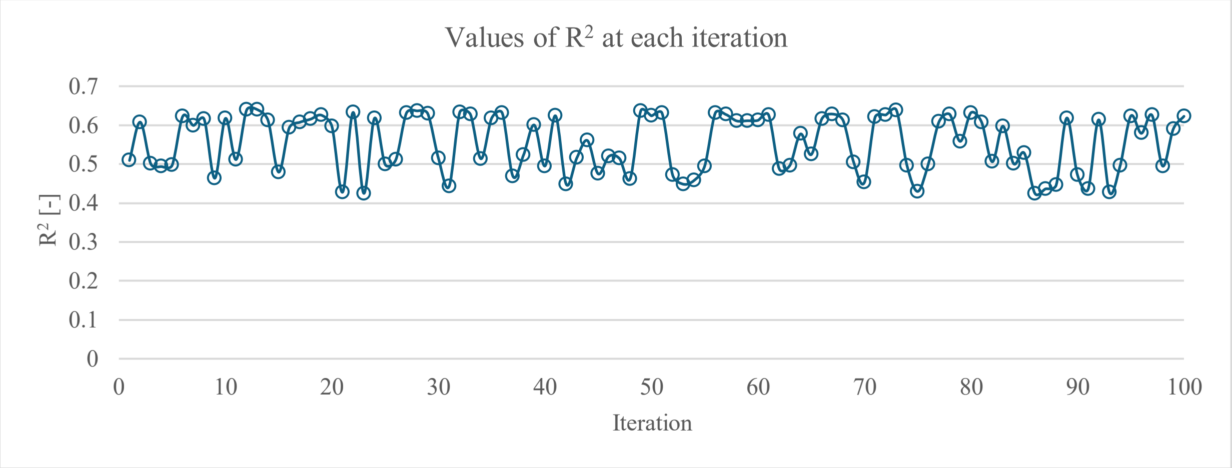

Figure 23 - Plot of the values of R^2 computed in the test phase for each of the 100 iterations.

Figure 23 clearly showed that, in most cases, R^2 resulted to be over 0.6, with several exceptions that went even below 0.5, but in general evenly distributed around the mean of 0.55. In any case, comparing the values obtained with the hybrid model to the linear regression, it was possible to state that the application of this approach was a significant improvement, which allowed to reach performances close to the double of the ones obtained with linear regression (0.33 versus 0.64 in the best run).

The influence of the R^2 could be observed by creating the scatter plot of SOM(t) versus the predicted values of SOM(t+Δt) through the model, and comparing it to the theoretical line in which the modelled SOM(t+Δt) perfectly corresponded to the measurement: a low value of coefficient of determination corresponded to a cloud-like distribution of data around the linear approximation, while as R^2 increases the points of the scatter plot tend to align along the model. The representation of the scatter plots in the worst ($R^2 = 0.425$) and best ($R^2 = 0.64$) iterations were showed in *Figure 24* and *Figure 25*.

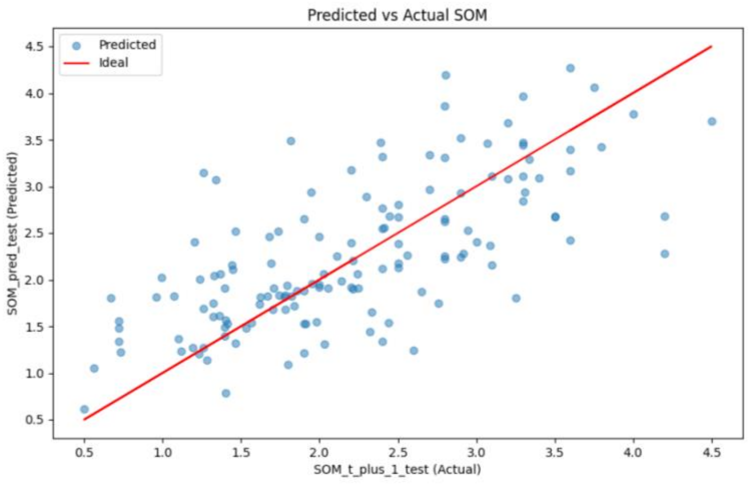

Figure 24 - Scatter plot representing the values of measured SOM(t+Δt) versus modelled SOM(t+Δt). The red line represents the ideal distribution obtained with the model. Worst case scenario (R^2 = 0.423)

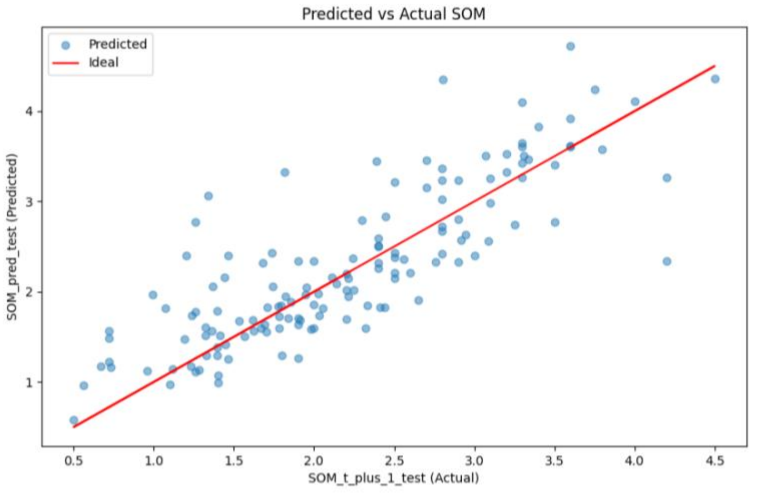

Figure 25 - Scatter plot representing the values of measured SOM(t+Δt) versus modelled SOM(t+Δt). The red line represents the ideal distribution obtained with the model. Best case scenario (R^2 = 0.64)

Comparing the two pictures, it can clearly be observed that, in both cases, the points do not follow perfectly the ideal line. However, in *Figure 24*, the points are much more scattered further from the line, while in *Figure 25* most of the observations are very close to the ideal line, and there are fewer outliers as well, so clearly the second plot shows a higher fitting performance than the first one.

The performance of the model can be compared to previous research aiming to estimate SOC. There are several studies on the application of ANN for SOC estimation and digital soil mapping, as reviewed in Lamichhane et al. (2019). For instance, Were et al. (2015) affirmed that ANN are one of the most suited types of model to estimate SOC, and obtained a value of R^2 = 0.6. Song et al. (2017) obtained a maximum R^2 = 0.49 for ANN, while Dai et al. (2014) obtained R^2 = 0.69.

These studies, however, were usually referred to an instantaneous estimate of SOC, while other studies, considered a dynamic approach, but they are usually developed on process-based models. In conclusion to this section, it can be stated that the model performance that was obtained can be considered moderately good. There still is room for improvement, which is briefly treated in the following section, but obtaining a $R^2 = 0.64$ with a simple, 0-D, two-parameter model for the description of natural complex interactions such as the carbon trapping mechanisms in soil can be considered a satisfactory result for this study.

6. Limitations and future work

The model in its best configuration, according to the coefficient of determination, can explain approximately 64% of the variance of the dependent variable through the independent variables, and the estimated carbon sequestration potential is consistent with previous research that used other data sources for the same pasture system. However, the study had some limitations that present several opportunities for further research and improvement:

- First, the number of explanatory variables introduced in the ANN was very high compared to the observations present in the dataset. In this thesis the technique of random variable assignment was implemented, but there are other possibilities, such as a study of the theoretical value of each of the two parameters to define which of the variables is more significant to their definition. Another opportunity is the study of the correlation matrix between each couple of variables, and to couple the ones with very high correlation, with the aim to reduce the number of available explanatory variables. Principle component analysis could also be used towards the same end. A third strategy that could be adopted is maintaining the random selection, which anyway performed quite effectively for the task, but more extensively, trying to cover as many combinations as possible (i.e., performing not only 100 iterations but running the code thousands of times). It is possible that one of those configurations that were not explored here could provide improved results.
- Another limitation of this study is the hyperparameter tuning, which could be implemented more effectively using a grid search algorithm instead of manually changing the parameters (dropout rate, LR reduction factor and patience, number of training epochs) to explore all possible combinations and consider more values. Moreover, the dimensions of the hidden layers of the ANN were limited to nine possible values for each model (which were selected between the most standard values applied to ANN), while more possibilities with different values can be explored. It would be possible to extend this number at the expense of added computational time. Furthermore, the possibility of increasing the complexity of the ANN could be explored, by changing the architecture of the deep learning model and introducing more hidden layers with a lower number of perceptrons instead of using a single, large intermediate layer.
- The main factor influencing the results of this study is, however, the simplicity of the process-based model that was applied. Here, we used a straightforward two-parameter, 0-D model as proof of concept that hybridization in soil models could be a promising new development for estimating soil carbon. In the future, it will be preferable to use a process-based model that depicts more complex interactions involved in soil and the carbon cycle. For instance, the application of the same procedure on the RothC model could be more effective. RothC has more parameters, as C input is not considered as a single aggregate but is divided into five fractions, corresponding to the fine soil carbon pools (DPM, RPM, IOM, BIO, HUM). Out of these five C inputs, the IOM pool remains unaltered (which is why it is called inert), while the

> other four pools mineralize carbon at four different mineralization rates. This setup provides a more complete description of the interactions of carbon in soil, increasing the number of parameters of the model. Substituting each of these parameters with its own ANN could result challenging and computationally demanding, but even the estimation of a few of them through the hybridization process could improve the performance and allow the creation of a model able to generalize the trends of evolution of SOC in different geographic areas and under different land use configurations and soil management systems. This could allow the optimization of soil management techniques to improve carbon sequestration, increasing the mitigation potential of this system as well as bringing the many co-benefits previously discussed.

Finally, one last opportunity for future research is represented by the possibility to apply the obtained model to perform projections of the SOM content trend. At first, the model is applied knowing both SOM(t) and SOM(t+Δt), to estimate its parameters, but once they are defined it is possible to fix the parameters and apply the model in its direct configuration to estimate SOM(t+Δt), defining a specific Δt. This is possible due to the presence of some variables in the database that are considered in a dynamic over time: as *Table 2* shows, the variables related to radiometric features, soil composition and topography are treated as constant throughout time, but climatic conditions (i.e., soil temperature and moisture) are considered both in at the instant (t) and at (t+Δt), suggesting the possibility of a variation in such conditions.

By creating projections of possible values of soil temperature and moisture in the defined time interval, the new observations could be inserted in the model, to observe how the values of SOM(t+Δt) are expected to change over time and, therefore, to define trends of soil carbon sequestration potential in the future.

7. Conclusions

The development of strategies for Carbon Capture and Storage is fundamental for the achievement of the Net Zero Emission goal, necessary to comply with the objectives of the Paris Agreement to avoid extreme climate change and all its consequences. This thesis focused on carbon sequestration in pasture soil as a tool for climate change mitigation that can bring some significant co-benefits to agriculture, food security and environmental factors. The study aimed at defining a hybrid model, combining a theory-driven, process-based model and a data-driven, deep learning algorithm (specifically ANN) to model carbon sequestration in pasture soils.

The result of this study is represented by the model found through the iteration with highest coefficient of determination ($R^2 = 0.64$), which is defined by a set of fixed weights for two ANN that replace the two parameters (carbon input and mineralization rate) of the process-based model, and a list of input variables for the two ANN. From these parameters, it was possible to estimate the annual potential carbon accumulation in this type of farms (1.8 t_{SOC}/ha·y), which is aligned with the values in the literature for SBPPRL.

The use of the hybridization approach provided a better fit to the data compared to linear regression ($R^2 = 0.33$), serving as proof of concept that this approach can, in the future, be extended to more complex soil models. The model can also be used to make projections for the future, which would make it an effective tool to define strategies to improve soil quality and increase carbon sequestration.

8. Bibliography

Activation Functions & Derivatives. (n.d.). https://jc-progjava.github.io/Building-Neural-Networks-From-Scratch/activation.html

Angelopoulou, T., Tziolas, N., Balafoutis, A., Zalidis, G., & Bochtis, D. (2019). Remote Sensing Techniques for Soil Organic Carbon Estimation: A review. *Remote Sensing*, *11*(6), 676. https://doi.org/10.3390/rs11060676

Arnfield, A. J. (2024, September 13). *Koppen climate classification | Definition, System, & Map*. Encyclopedia Britannica. https://www.britannica.com/science/Koppen-climate-classification

Biotoken. (2023, August 1). *Raising the flag on soil carbon credits*. Biotoken. https://biotoken.world/en/raising-the-flag-on-soil-carbon-credits/#.

Bolinder, M. A., Kätterer, T., Andrén, O., & Parent, L. E. (2012). Estimating carbon inputs to soil in forage-based crop rotations and modeling the effects on soil carbon dynamics in a Swedish long-term field experiment. *Canadian Journal of Soil Science*, *92*(6), 821–833. https://doi.org/10.4141/cjss2012-036

Bossio, D. A., Cook-Patton, S. C., Ellis, P. W., Fargione, J., Sanderman, J., Smith, P., Wood, S., Zomer, R. J., Von Unger, M., Emmer, I. M., & Griscom, B. W. (2020). The role of soil carbon in natural climate solutions. *Nature Sustainability*, *3*(5), 391–398. https://doi.org/10.1038/s41893-020-0491-z

Calvin, K., Dasgupta, D., Krinner, G., Mukherji, A., Thorne, P. W., Trisos, C., Romero, J., Aldunce, P., Barrett, K., Blanco, G., Cheung, W. W., Connors, S., Denton, F., Diongue-Niang, A., Dodman, D., Garschagen, M., Geden, O., Hayward, B., Jones, C., . . . Ha, M. (2023). *IPCC, 2023: Climate Change 2023: Synthesis Report. Contribution of Working Groups I, II and III to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change [Core Writing Team, H. Lee and J. Romero (eds.)]. IPCC, Geneva, Switzerland.* https://doi.org/10.59327/ipcc/ar6-9789291691647

Coleman, K., & Jenkinson, D. S. (1996a). RothC-26.3 - A Model for the turnover of carbon in soil. In *Springer eBooks* (pp. 237–246). https://doi.org/10.1007/978-3-642-61094-3_17

Coleman, K., & Jenkinson, D. S. (1996b). RothC-26.3 - A Model for the turnover of carbon in soil. In *Springer eBooks* (pp. 237–246). https://doi.org/10.1007/978-3-642-61094-3_17

Cuddington, K., Fortin, M., Gerber, L. R., Hastings, A., Liebhold, A., O'Connor, M., & Ray, C. (2013). Process-based models are required to manage ecological systems in a changing world. *Ecosphere*, *4*(2), 1–12. https://doi.org/10.1890/es12-00178.1

Dai, F., Zhou, Q., Lv, Z., Wang, X., & Liu, G. (2014). Spatial prediction of soil organic matter content integrating artificial neural network and ordinary kriging in Tibetan Plateau. *Ecological Indicators*, *45*, 184-194. https://doi.org/10.1016/j.ecolind.2014.04.003

Diele, F., Marangi, C., & Martiradonna, A. (2021). Non-Standard Discrete ROTHC models for soil carbon dynamics. *Axioms*, *10*(2), 56. https://doi.org/10.3390/axioms10020056

Dotse, S., Larbi, I., Limantol, A. M., & De Silva, L. C. (2023). A review of the application of hybrid machine learning models to improve rainfall prediction. *Modeling Earth Systems and Environment*, *10*(1), 19–44. https://doi.org/10.1007/s40808-023-01835-x

Falloon, P. D., & Smith, P. (2002). Simulating SOC changes in long-term experiments with RothC and CENTURY: model evaluation for a regional scale application. *Soil Use and Management*, *18*(2), 101–111. https://doi.org/10.1111/j.1475-2743.2002.tb00227.x

FAO. (2019a). *Standard operating procedure for soil total carbon: Dumas dry combustion method*. https://openknowledge.fao.org/server/api/core/bitstreams/d0f932cf-88f5-4e1d-b3a7-ee2535761f49/content

FAO. (2019b). *Biodiversity and the livestock sector - Guidelines for quantitative assessment: Version 1*. Food & Agriculture Organization of the United Nations.

Garnett, T., Godde, C., Muller, A., Röös, E., Smith, P., De Boer, I., Ermgassen, E. Z., Herrero, M., Van Middelaar, C., Schader, C., & Van Zanten, H. (2017). *Grazed and confused? : Ruminating on cattle, grazing systems, methane, nitrous oxide, the soil carbon sequestration question - and what it all means for greenhouse gas emissions*. https://library.wur.nl/WebQuery/wurpubs/fulltext/427016

Ge, Y., Thomasson, J. A., & Sui, R. (2011). Remote sensing of soil properties in precision agriculture: A review. *Frontiers of Earth Science*. https://doi.org/10.1007/s11707-011-0175-0

Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media.

Goidts, E., Van Wesemael, B., & Crucifix, M. (2009). Magnitude and sources of uncertainties in soil organic carbon (SOC) stock assessments at various scales. *European Journal of Soil Science*, *60*(5), 723–739. https://doi.org/10.1111/j.1365-2389.2009.01157.x

Lal, R. (2003). Global potential of soil carbon sequestration to mitigate the greenhouse effect. *Critical Reviews in Plant Sciences*, *22*(2), 151–184. https://doi.org/10.1080/713610854

Lal, R. (2007). Carbon sequestration. *Philosophical Transactions of the Royal Society B Biological Sciences*, *363*(1492), 815–830. https://doi.org/10.1098/rstb.2007.2185

Lal, R., Negassa, W., & Lorenz, K. (2015). Carbon sequestration in soil. *Current Opinion in Environmental Sustainability*, *15*, 79–86. https://doi.org/10.1016/j.cosust.2015.09.002

Lamichhane, S., Kumar, L., & Wilson, B. (2019). Digital soil mapping algorithms and covariates for soil organic carbon mapping and their implications: A review. *Geoderma*, *352*, 395–413. https://doi.org/10.1016/j.geoderma.2019.05.031

Liu, D. L., Chan, K. Y., Conyers, M. K., Li, G., & Poile, G. J. (2011). Simulation of soil organic carbon dynamics under different pasture managements using the RothC carbon model. *Geoderma*, *165*(1), 69–77. https://doi.org/10.1016/j.geoderma.2011.07.005

Liu, L., Zhou, W., Guan, K., Peng, B., Xu, S., Tang, J., Zhu, Q., Till, J., Jia, X., Jiang, C., Wang, S., Qin, Z., Kong, H., Grant, R., Mezbahuddin, S., Kumar, V., & Jin, Z. (2024). Knowledge-guided machine learning can improve carbon cycle quantification in agroecosystems. *Nature Communications*, *15*(1). https://doi.org/10.1038/s41467-023-43860-5

Medium. (n.d.). Medium. https://medium.com/data-science-365/overview-of-a-neural-networks-learning-process-61690a502fa.

Minasny, B., McBratney, A. B., Wadoux, A. M., Akoeb, E. N., & Sabrina, T. (2020). Precocious 19th century soil carbon science. *Geoderma Regional*, *22*, e00306. https://doi.org/10.1016/j.geodrs.2020.e00306

Morais, T. G., Jongen, M., Tufik, C., Rodrigues, N. R., Gama, I., Serrano, J., Gonçalves, M. C., Mano, R., Domingos, T., & Teixeira, R. F. M. (2023). Satellite-based estimation of soil organic carbon in Portuguese grasslands. *Frontiers in Environmental Science*, *11*. https://doi.org/10.3389/fenvs.2023.1240106

Morais, T. G., Teixeira, R. F., & Domingos, T. (2019). Detailed global modelling of soil organic carbon in cropland, grassland and forest soils. *PLoS ONE*, *14*(9), e0222604. https://doi.org/10.1371/journal.pone.0222604

Morais, T. G., Teixeira, R. F. M., Rodrigues, N. R., & Domingos, T. (2018). Characterizing livestock production in Portuguese sown rainfed grasslands: Applying the inverse approach to a Process-Based model. *Sustainability*, *10*(12), 4437. https://doi.org/10.3390/su10124437

Orgiazzi, A., Ballabio, C., Panagos, P., Jones, A., & Fernández-Ugalde, O. (2017). LUCAS Soil, the largest expandable soil dataset for Europe: a review. *European Journal of Soil Science*, *69*(1), 140–153. https://doi.org/10.1111/ejss.12499

Padarian, J., Stockmann, U., Minasny, B., & McBratney, A. (2022). Monitoring changes in global soil organic carbon stocks from space. *Remote Sensing of Environment*, *281*, 113260. https://doi.org/10.1016/j.rse.2022.113260

Panagos, P., De Rosa, D., Liakos, L., Labouyrie, M., Borrelli, P., & Ballabio, C. (2024). Soil bulk density assessment in Europe. *Agriculture Ecosystems & Environment*, *364*, 108907. https://doi.org/10.1016/j.agee.2024.108907

Peduzzi, P., Concato, J., Kemper, E., Holford, T. R., & Feinstein, A. R. (1996). A simulation study of the number of events per variable in logistic regression analysis. *Journal of Clinical Epidemiology*, *49*(12), 1373–1379. https://doi.org/10.1016/s0895-4356(96)00236-3

Pribyl, D. W. (2010). A critical review of the conventional SOC to SOM conversion factor. *Geoderma*, *156*(3–4), 75–83. https://doi.org/10.1016/j.geoderma.2010.02.003

Reichstein, M., Camps-Valls, G., Stevens, B., Jung, M., Denzler, J., Carvalhais, N., & Prabhat, N. (2019). Deep learning and process understanding for data-driven Earth system science. *Nature*, *566*(7743), 195–204. https://doi.org/10.1038/s41586-019-0912-1

Shukla, P., Skea, J., Buendia, E. C., Masson-Delmotte, Pörtner, H., Roberts, D., Zhai, P., Slade, R., Connors, S., Van Diemen, R., Ferrat, M., Haughey, E., Luz, S., Neogi, S., Pathak, M., Petzold, J., Pereira, J. P., Vyas, P., Huntley, E., . . . Malley, J. (2019). *IPCC, 2019: Climate Change and Land: an IPCC special report on climate change, desertification,* *land degradation, sustainable land management, food security, and greenhouse gas fluxes in terrestrial ecosystems*. https://doi.org/10.25561/76618

Skjemstad, J. O., Spouncer, L. R., Cowie, B., & Swift, R. S. (2004). Calibration of the Rothamsted organic carbon turnover model (RothC ver. 26.3), using measurable soil organic carbon pools. *Soil Research*, *42*(1), 79. https://doi.org/10.1071/sr03013

Slater, L. J., Arnal, L., Boucher, M., Chang, A. Y., Moulds, S., Murphy, C., Nearing, G., Shalev, G., Shen, C., Speight, L., Villarini, G., Wilby, R. L., Wood, A., & Zappa, M. (2023). Hybrid forecasting: blending climate predictions with AI models. *Hydrology and Earth System Sciences*, *27*(9), 1865–1889. https://doi.org/10.5194/hess-27-1865-2023

Smith, P. (2012). Soils and climate change. *Current Opinion in Environmental Sustainability*, *4*(5), 539–544. https://doi.org/10.1016/j.cosust.2012.06.005

Smith, P., Fang, C., Dawson, J. J., & Moncrieff, J. B. (2008). Impact of global warming on soil organic carbon. In *Advances in agronomy* (pp. 1–43). https://doi.org/10.1016/s0065-2113(07)00001-6

Smith, P., Smith, J., Powlson, D., McGill, W., Arah, J., Chertov, O., Coleman, K., Franko, U., Frolking, S., Jenkinson, D., Jensen, L., Kelly, R., Klein-Gunnewiek, H., Komarov, A., Li, C., Molina, J., Mueller, T., Parton, W., Thornley, J., & Whitmore, A. (1997). A comparison of the performance of nine soil organic matter models using datasets from seven long-term experiments. *Geoderma*, *81*(1–2), 153–225. https://doi.org/10.1016/s0016-7061(97)00087-6

Smith, P., Soussana, J., Angers, D., Schipper, L., Chenu, C., Rasse, D. P., Batjes, N. H., Van Egmond, F., Mcneill, S., Kuhnert, M., Navarro, C. A., Olesen, J. E., Chirinda, N., Fornara, D., Wollenberg, E., Sanz-cobena, A., & Klumpp, K. (2019). How to measure, report and verify soil carbon change to realize the potential of soil carbon sequestration for atmospheric greenhouse gas removal. *Global Change Biology*, *26*(1), 219–241. https://doi.org/10.1111/gcb.14815

Song, Y., Yang, L., Li, B., Hu, Y., Wang, A., Zhou, W., Cui, X., & Liu, Y. (2017). Spatial prediction of soil organic matter using a hybrid geostatistical model of an extreme learning machine and ordinary kriging. *Sustainability*, *9*(5), 754. https://doi.org/10.3390/su9050754

Soussana, J., Tallec, T., & Blanfort, V. (2010). Mitigating the greenhouse gas balance of ruminant production systems through carbon sequestration in grasslands. *Animal*, *4*(3), 334–350. https://doi.org/10.1017/s1751731109990784

Soussana, J., Loiseau, P., Vuichard, N., Ceschia, E., Balesdant, J., Chevallier, T., & Arrouays, D. (2004). Carbon cycling and sequestration opportunities in temperate grasslands. *Soil Use and Management*, *20*(2), 219–230. https://doi.org/10.1079/sum2003234

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, *15*(1), 1929–1958. https://jmlr.csail.mit.edu/papers/volume15/srivastava14a/srivastava14a.pdf

Teagasc. (n.d.). *Soil carbon Sequestration - TEAGASC | Agriculture and Food Development Authority*. https://www.teagasc.ie/about/farm-advisory/advisory-regions/cork-east/farm-advice/soil-carbon-sequestration/

Teixeira, R., Domingos, T., Costa, A., Oliveira, R., Farropas, L., Calouro, F., Barradas, A., & Carneiro, J. (2011). Soil organic matter dynamics in Portuguese natural and sown rainfed grasslands. *Ecological Modelling*, *222*(4), 993–1001. https://doi.org/10.1016/j.ecolmodel.2010.11.013

Teixeira, R. F. (2010). Sustainable land uses and carbon sequestration: the case of sown biodiverse permanent pastures rich in legumes. *ResearchGate*. https://www.researchgate.net/publication/267765149

Teixeira, R. F., Proença, V., Crespo, D., Valada, T., & Domingos, T. (2015). A conceptual framework for the analysis of engineered biodiverse pastures. *Ecological Engineering*, *77*, 85–97. https://doi.org/10.1016/j.ecoleng.2015.01.002

Vereecken, H., Schnepf, A., Hopmans, J. W., Javaux, M., Or, D., Roose, D. O. T., Vanderborght, J., Young, M. H., Amelung, W., Aitkenhead, M., Allison, S. D., Assouline, S., Baveye, P., Berli, M., Bruggemann, N., Finke, P., Flury, M., Gaiser, T., Govers, G., . . . Young, I. M. (2016). Modeling soil Processes: review, key challenges, and new perspectives. *Vadose Zone Journal*, *15*(5), 1–57. https://doi.org/10.2136/vzj2015.09.0131

Were, K., Bui, D. T., Dick, Ø. B., & Singh, B. R. (2015). A comparative assessment of support vector regression, artificial neural networks, and random forests for predicting and mapping soil organic carbon stocks across an Afromontane landscape. *Ecological Indicators*, *52*, 394–403. https://doi.org/10.1016/j.ecolind.2014.12.028

Wiley, H. W. (1906). *Principles and practice of agricultural analysis; a manual for the study of soils, fertilizers, and agricultural products; for the use of analysists, teachers, and students of agricultural chemistry*. https://doi.org/10.5962/bhl.title.31806

Willard, J., Jia, X., Xu, S., Steinbach, M., & Kumar, V. (2022). Integrating Scientific Knowledge with Machine Learning for Engineering and Environmental Systems. *ACM Computing Surveys*, *55*(4), 1–37. https://doi.org/10.1145/3514228

Zhang, Y., Guo, X., Pei, H., Min, L., Liu, F., & Shen, Y. (2022). Evapotranspiration and carbon exchange of the main agroecosystems and their responses to agricultural land use change in North China Plain. *Agriculture Ecosystems & Environment*, *338*, 108103. https://doi.org/10.1016/j.agee.2022.108103

9. Annex: codes used for the development of the model

9.1 Linear regression

```
import pandas as pd
from sklearn.model_selection import train_test_split
import torch
import numpy as np
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
import os

def create_folder(folder_path):
    if not os.path.exists(folder_path):
        os.makedirs(folder_path)

LinReg_folder = 'LinReg_plots'
create_folder(LinReg_folder)

# Load the data
file_path = 'Database_no_t1.xlsx'
data = pd.read_excel(file_path)

SOM_t = data.iloc[:,0].values.reshape(-1,1)
SOM_t_plus_1 = data.iloc[:,1].values.reshape(-1, 1)
delta_t = data.iloc[:,2].values.reshape(-1, 1)

SOM_t_tensor = torch.tensor(SOM_t, dtype=torch.float32)
SOM_t_plus_1_tensor = torch.tensor(SOM_t_plus_1, dtype=torch.float32)
delta_t_tensor = torch.tensor(delta_t, dtype=torch.float32)

# Split the dataset into training and test sets (e.g., 80% for training,
20% for test)
SOM_t_train, SOM_t_test, SOM_t_plus_1_train, SOM_t_plus_1_test,
delta_t_train, delta_t_test = train_test_split(SOM_t, SOM_t_plus_1,
delta_t, test_size=0.2, random_state=42)

# Compute average delta t, to use in the linear regression formula for
training and test set
delta_t_train = np.array(delta_t_train)
t_avg_train = np.mean(delta_t_train)

delta_t_test = np.array(delta_t_test)
t_avg_test = np.mean(delta_t_test)
```

```
# Create a linear regression model and fit data
model = LinearRegression()
model.fit(SOM_t_train, SOM_t_plus_1_train)

# Get the slope (m) and intercept (q)
m = model.coef_[0]
q = model.intercept_

# Print the coefficients
print("Slope (m):", m)
print("Intercept (q):", q)

# Make predictions
SOM_t_plus_1_pred = model.predict(SOM_t_train)

# Plotting (training set)
plt.scatter(SOM_t_train, SOM_t_plus_1_train, color='blue')  # Original
data points
plt.plot(SOM_t_train, SOM_t_plus_1_pred, color='red')  # Regression line
plt.xlabel('SOM (t)')
plt.ylabel('SOM (t+1)')
plt.title('Linear Regression')
plt.show()

# Evaluate parameters of the model
b = (-np.log(m))/t_avg_train
a = (q·b)/(1-np.exp(-b·t_avg_train))

# Compute SOM in the test dataset
SOM_t_plus_1_pred_test = m · SOM_t_test + q

# Plotting (test set)
plt.figure(figsize=(10, 6))
plt.scatter(SOM_t_test, SOM_t_plus_1_test, color='blue')  # Original data
points
plt.plot(SOM_t_test, SOM_t_plus_1_pred_test, color='red')  # Regression
line
plt.xlabel('SOM (t)')
plt.ylabel('SOM (t+1)')
plt.title('Linear Regression')
plt.show()

# Save the plot with a unique filename for each iteration
plot_filename = os.path.join(LinReg_folder,
f'SOM_plot_LinearRegression.png')
plt.savefig(plot_filename)
```

```
# Compute R^2 manually
ss_total = np.sum((SOM_t_plus_1_test - np.mean(SOM_t_plus_1_test))··2)
ss_residual = np.sum((SOM_t_plus_1_test - SOM_t_plus_1_pred_test)··2)
r2_manual = 1 - (ss_residual / ss_total)

print("R^2 computed manually:", r2_manual)
```

9.2 Dataset splitting

```
import pandas as pd
from sklearn.model_selection import train_test_split
import numpy as np
import torch

# Load the dataset
file_path = 'Database_no_t1.xlsx'
data = pd.read_excel(file_path)

# Display the first few rows to understand the structure of the dataset
data.head()

# Calculate the overall mean and standard deviation of 'SOM (t+1)'
mean_som = data['SOM (t+1)'].mean()
std_som = data['SOM (t+1)'].std()

# Display the overall statistics for 'SOM (t+1)'
mean_som, std_som

# Function for the research of the best number of bins
def grid_search_bins(data, bins_range, target_column):
    best_bins = None
    best_balance_score = float('inf')  # Initialize to a high value

    for bins in bins_range:
        # Bin the target column
        data['bin'] = pd.qcut(data[target_column], q=bins, labels=False,
duplicates='drop')

        # Split the data into training (60%) and temp (40%) subsets,
stratified by the 'bin' column
        train_data, temp_data = train_test_split(
            data, test_size=0.4, stratify=data['bin'], random_state=42)

        # Then, split the temp data into validation (20%) and test (20%)
sets
```

```
        val_data, test_data = train_test_split(
            temp_data, test_size=0.5, stratify=temp_data['bin'], 
random_state=42)

        # Calculate the balance score (variance of bin sizes) for training 
data
        bin_sizes_train = train_data['bin'].value_counts()
        balance_score = bin_sizes_train.var()  # Lower variance is better

        # Update the best bins if the current configuration is better
        if balance_score < best_balance_score:
            best_bins = bins
            best_balance_score = balance_score

    return best_bins

# Define best number of bins
bins_range = range(5, 31)
best_bins = grid_search_bins(data, bins_range, 'SOM (t+1)')
print(f"Best number of bins: {best_bins}")

# Create bins for 'SOM (t+1)' to group similar values
data['SOM_bin'] = pd.qcut(data['SOM (t+1)'], q=best_bins, labels=False)

# First, split the data into training (60%) and temp (40%) subsets, 
stratified by the 'SOM_bin' column
train_data, temp_data = train_test_split(
    data, test_size=0.4, stratify=data['SOM_bin'], random_state=42)
# Then, split the temp data into validation (20%) and test (20%) sets
val_data, test_data = train_test_split(
    temp_data, test_size=0.5, stratify=temp_data['SOM_bin'], 
random_state=42)

# Drop the 'SOM_bin' column from the final subsets
train_data = train_data.drop(columns=['SOM_bin'])
val_data = val_data.drop(columns=['SOM_bin'])
test_data = test_data.drop(columns=['SOM_bin'])

# Verify the means and standard deviations for each subset
train_mean_std = (train_data['SOM (t+1)'].mean(), train_data['SOM 
(t+1)'].std())
val_mean_std = (val_data['SOM (t+1)'].mean(), val_data['SOM (t+1)'].std())
test_mean_std = (test_data['SOM (t+1)'].mean(), test_data['SOM 
(t+1)'].std())

train_mean_std, val_mean_std, test_mean_std
```

```
# Export obtained datasets to excel
train_data.to_excel('training_set.xlsx', index=False)
val_data.to_excel('validation_set.xlsx', index=False)
test_data.to_excel('test_set.xlsx', index=False)
```

9.3 Model

```
import torch
import torch.nn as nn
import torch.nn.functional as F
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import r2_score
import matplotlib.pyplot as plt
import os


# Define the ANN model
class ANN(nn.Module):
    def __init__(self, input_size, hidden_size, output_activation=None,
dropout_rate=0.9):
        super(ANN, self).__init__()
        self.layers = nn.Sequential(
            nn.Linear(input_size, hidden_size),
            nn.ReLU(),
            nn.Dropout(dropout_rate),
            nn.Linear(hidden_size, 1)
        )
        self.output_activation = output_activation

    def forward(self, x):
        x = self.layers(x)
        if self.output_activation:
            x = self.output_activation(x)
        return x


# Function to log results to a file
def log_variable_selection(file_path, inputs_f_columns, inputs_g_columns):
    with open(file_path, 'a') as file:
        file.write(f'Iteration: {iteration + 1}\n')
        file.write(f'inputs_f: {inputs_f_columns}\n')
        file.write(f'inputs_g: {inputs_g_columns}\n\n')
```

```
# Define a function to log results to a file
def log_results(file_path, size_f, size_g, val_loss, loss, r2, a, b):
    with open(file_path, 'a') as file:
        file.write(f'Iteration: {iteration + 1}\n')
        file.write(f'Hidden size f: {size_f}, Hidden size g: {size_g},
Best validation loss: {val_loss}'
                   f'Test loss: {loss}, R^2: {r2}, a_pred: {a}, b_pred:
{b}\n\n')


# Define a function to log the complete vectors a and b to a file
def log_parameters(file_path, a_vec, b_vec):
    with open(file_path, 'a') as file:
        file.write(f'Iteration: {iteration + 1}\n')
        file.write(f'Values of a: {a_vec}\n')
        file.write(f'Values of b: {b_vec}\n')


# Define a function to create a folder
def create_folder(folder_path):
    if not os.path.exists(folder_path):
        os.makedirs(folder_path)


# Define a custom actiation function for the output
def custom_activation(x):
    return 0.3 · torch.sigmoid(x)


# Path to the log files
log_variables_file_path = 'column_assignments_log.txt'
log_results_file_path = 'best_values_log.txt'
log_parameters_file_path = 'parameters_log.txt'

# Path to the folder where you want to save the plots (train-val loss)
loss_plots_folder = 'loss_plots'
create_folder(loss_plots_folder)
# Path to the folder where you want to save the plots (scatter plots)
scatter_plots_folder = 'SOM_plots'
create_folder(scatter_plots_folder)

# Load the data
file_path_tr = 'training_set.xlsx'
data_train = pd.read_excel(file_path_tr)

file_path_v = 'validation_set.xlsx'
data_val = pd.read_excel(file_path_v)
```

```
file_path_te = 'test_set.xlsx'
data_test = pd.read_excel(file_path_te)

# Slice the DataFrame to start from the fourth column
data_sliced_train = data_train.iloc[:, 3:]
data_sliced_val = data_val.iloc[:, 3:]
data_sliced_test = data_test.iloc[:, 3:]

# Create SOM(t), SOM(t+1) and dt vectors
SOM_t_train = data_train.iloc[:, 0].values.reshape(-1, 1)
SOM_t_plus_1_train = data_train.iloc[:, 1].values.reshape(-1, 1)
delta_t_train = data_train.iloc[:, 2].values.reshape(-1, 1)

SOM_t_val = data_val.iloc[:, 0].values.reshape(-1, 1)
SOM_t_plus_1_val = data_val.iloc[:, 1].values.reshape(-1, 1)
delta_t_val = data_val.iloc[:, 2].values.reshape(-1, 1)

SOM_t_test = data_test.iloc[:, 0].values.reshape(-1, 1)
SOM_t_plus_1_test = data_test.iloc[:, 1].values.reshape(-1, 1)
delta_t_test = data_test.iloc[:, 2].values.reshape(-1, 1)

# Get the number of columns in the data
num_columns = len(data_sliced_train.columns)

# Define the number of iterations to train the code
num_iterations = 100

for iteration in range(num_iterations):

    # Randomly assign each row to inputs_f, inputs_g, both, or none
    random_assignment = np.random.choice(['inputs_f', 'inputs_g', 'both',
'none'], size=num_columns)

    # Initialize inputs_f and inputs_g
    inputs_f_columns = []
    inputs_g_columns = []
    both_columns = []
    none_columns = []

    for column_name, assignment in zip(data_sliced_train.columns,
random_assignment):
        if assignment == 'inputs_f' or assignment == 'both':
            inputs_f_columns.append(column_name)
        if assignment == 'inputs_g' or assignment == 'both':
            inputs_g_columns.append(column_name)
        if assignment == 'both':
            both_columns.append(column_name)
        if assignment == 'none':
            none_columns.append(column_name)

    # Print the number of columns assigned to each category
    print(f'Number of columns assigned to inputs_f:
{len(inputs_f_columns)}')
    print(f'Number of columns assigned to inputs_g:
{len(inputs_g_columns)}')

    n_inputs_f = len(inputs_f_columns)
    n_inputs_g = len(inputs_g_columns)

    # Print the variables going into each category
    print("\nVariables going into inputs_f:")
    print(inputs_f_columns)

    print("\nVariables going into inputs_g:")
    print(inputs_g_columns)

    # Log results to file
    log_variable_selection(log_variables_file_path, inputs_f_columns,
inputs_g_columns)

    # Extract the data for inputs_f and inputs_g
    inputs_f_train = data_sliced_train[inputs_f_columns].values
    inputs_g_train = data_sliced_train[inputs_g_columns].values

    # Extract the data for inputs_f and inputs_g
    inputs_f_val = data_sliced_val[inputs_f_columns].values
    inputs_g_val = data_sliced_val[inputs_g_columns].values

    # Extract the data for inputs_f and inputs_g
    inputs_f_test = data_sliced_test[inputs_f_columns].values
    inputs_g_test = data_sliced_test[inputs_g_columns].values

    # Preprocess the data
    scaler_f = StandardScaler()
    scaler_g = StandardScaler()

    inputs_f_train_scaled = scaler_f.fit_transform(inputs_f_train)
    inputs_g_train_scaled = scaler_g.fit_transform(inputs_g_train)
    inputs_f_val_scaled = scaler_f.transform(inputs_f_val)
    inputs_g_val_scaled = scaler_g.transform(inputs_g_val)
    inputs_f_test_scaled = scaler_f.transform(inputs_f_test)
    inputs_g_test_scaled = scaler_g.transform(inputs_g_test)
    inputs_f_train = torch.tensor(inputs_f_train_scaled,
dtype=torch.float32)
    inputs_g_train = torch.tensor(inputs_g_train_scaled,
dtype=torch.float32)
    SOM_t_train = torch.tensor(SOM_t_train, dtype=torch.float32)
    SOM_t_plus_1_train = torch.tensor(SOM_t_plus_1_train,
dtype=torch.float32)
    delta_t_train = torch.tensor(delta_t_train, dtype=torch.float32)

    inputs_f_val = torch.tensor(inputs_f_val_scaled, dtype=torch.float32)
    inputs_g_val = torch.tensor(inputs_g_val_scaled, dtype=torch.float32)
    SOM_t_val = torch.tensor(SOM_t_val, dtype=torch.float32)
    SOM_t_plus_1_val = torch.tensor(SOM_t_plus_1_val, dtype=torch.float32)
    delta_t_val = torch.tensor(delta_t_val, dtype=torch.float32)

    inputs_f_test = torch.tensor(inputs_f_test_scaled,
dtype=torch.float32)
    inputs_g_test = torch.tensor(inputs_g_test_scaled,
dtype=torch.float32)
    SOM_t_test = torch.tensor(SOM_t_test, dtype=torch.float32)
    SOM_t_plus_1_test = torch.tensor(SOM_t_plus_1_test,
dtype=torch.float32)
    delta_t_test = torch.tensor(delta_t_test, dtype=torch.float32)

    # Define the directory to save the model
    model_dir = 'D:/models/'  # Adjust this path

    # Check if the directory exists, and create it if it doesn't
    if not os.path.exists(model_dir):
        os.makedirs(model_dir)

    model_f_filename = 'model_f.pth'
    model_g_filename = 'model_g.pth'

    model_f_path = os.path.join(model_dir, model_f_filename)
    model_g_path = os.path.join(model_dir, model_g_filename)

    # Initialize models
    best_val_loss = np.inf
    inner_best_val_loss = np.inf
    best_hidden_size = None
    best_hidden_size_g = None
    hidden_sizes = [8, 16, 32, 64, 128, 256, 512, 1024, 2048]

    # Lists to store loss values
    train_losses = []
    val_losses = []
    train_r2_scores = []
    val_r2_scores = []

    for hidden_size in hidden_sizes:
        model_f = ANN(input_size=n_inputs_f, hidden_size=hidden_size,
output_activation=nn.Softplus())
        print(f'#################### Hidden size of f: {hidden_size}')

        for hidden_size_g in hidden_sizes:
            model_g = ANN(input_size=n_inputs_g,
hidden_size=hidden_size_g, output_activation=custom_activation)
            print(f'#################### Hidden size of g:
{hidden_size_g}')
            optimizer = torch.optim.Adam(list(model_f.parameters()) +
list(model_g.parameters()), lr=0.001)
            loss_function = nn.MSELoss()

            scheduler =
torch.optim.lr_scheduler.ReduceLROnPlateau(optimizer, 'min', patience=20,
factor=0.1)

            for epoch in range(4000):
                model_f.train()
                model_g.train()
                optimizer.zero_grad()
                a_pred = model_f(inputs_f_train)
                b_pred = model_g(inputs_g_train)
                SOM_pred = (a_pred / b_pred) · (1 - torch.exp(-b_pred ·
delta_t_train)) + (
                    torch.exp(-b_pred · delta_t_train)) · SOM_t_train

                # Ensure tensors have the same shape
                if SOM_t_plus_1_train.dim() == 1:
                    SOM_t_plus_1_train = SOM_t_plus_1_train.view(-1, 1)
                if SOM_pred.dim() == 1:
                    SOM_pred = SOM_pred.view(-1, 1)

                loss_train = loss_function(SOM_pred, SOM_t_plus_1_train)
                loss_train.backward()
                optimizer.step()

                model_f.eval()
                model_g.eval()

                with torch.no_grad():
                    a_pred_val = model_f(inputs_f_val)
                    b_pred_val = model_g(inputs_g_val)
                    SOM_pred_val = (a_pred_val / b_pred_val) · (1 -
torch.exp(-b_pred_val · delta_t_val)) + (
                        torch.exp(-b_pred_val · delta_t_val)) · SOM_t_val

                    # Ensure tensors have the same shape
                    if SOM_t_plus_1_val.dim() == 1:
                        SOM_t_plus_1_val = SOM_t_plus_1_val.view(-1, 1)
                    if SOM_pred_val.dim() == 1:
                        SOM_pred_val = SOM_pred_val.view(-1, 1)

                    loss_val = loss_function(SOM_pred_val,
SOM_t_plus_1_val)

                    # Calculate R^2 scores
                    r2_train = r2_score(SOM_t_plus_1_train.numpy(),
SOM_pred.numpy())
                    r2_val = r2_score(SOM_t_plus_1_val.numpy(),
SOM_pred_val.numpy())
                    train_r2_scores.append(r2_train)
                    val_r2_scores.append(r2_val)

                    if loss_val < best_val_loss:
                        best_val_loss = loss_val
                        best_hidden_size = hidden_size
                        best_hidden_size_g = hidden_size_g
                        best_state_dict_f = model_f.state_dict()
                        best_state_dict_g = model_g.state_dict()

                    scheduler.step(loss_val)

                train_losses.append(loss_train.item())
                val_losses.append(loss_val.item())

    # Save the best models
    torch.save({
        'hidden_size': best_hidden_size,
        'state_dict': best_state_dict_f
    }, model_f_path)

    torch.save({
        'hidden_size': best_hidden_size_g,
        'state_dict': best_state_dict_g
    }, model_g_path)

    print(f'Best hidden size: {best_hidden_size}')
    print(f'Best hidden size g: {best_hidden_size_g}')
    print(f'Best Validation loss: {best_val_loss}')
    # Plotting the loss trends
    plt.figure(figsize=(10, 5))
    plt.plot(train_losses, label='Training Loss')
    plt.plot(val_losses, label='Validation Loss')
    plt.xlabel('Epoch')
    plt.ylabel('Loss')
    plt.title('Loss Trend During Training')
    plt.legend()
    # plt.show()

    # Save the plot with a unique filename for each iteration
    plot_filename = os.path.join(loss_plots_folder,
f'loss_plot_iteration_{iteration + 1}.png')
    plt.savefig(plot_filename)
    plt.close()  # Close the plot to avoid display in subsequent
iterations

    print(f"Plot saved: {plot_filename}")

    # Load the best models
    checkpoint_f = torch.load(model_f_path)
    checkpoint_g = torch.load(model_g_path)

    best_hidden_size = checkpoint_f['hidden_size']
    best_hidden_size_g = checkpoint_g['hidden_size']

    model_f = ANN(input_size=n_inputs_f, hidden_size=best_hidden_size,
output_activation=nn.Softplus())
    model_g = ANN(input_size=n_inputs_g, hidden_size=best_hidden_size_g,
output_activation=custom_activation)

    model_f.load_state_dict(checkpoint_f['state_dict'])
    model_g.load_state_dict(checkpoint_g['state_dict'])

    model_f.eval()
    model_g.eval()

    # applied to test set to measure error
    with torch.no_grad():
        a_pred_test = model_f(inputs_f_test)
        b_pred_test = model_g(inputs_g_test)
        SOM_pred_test = (a_pred_test / b_pred_test) · (1 - torch.exp(-
b_pred_test · delta_t_test)) + (
            torch.exp(-b_pred_test · delta_t_test)) · SOM_t_test

        # Ensure tensors have the same shape
        if SOM_t_plus_1_test.dim() == 1:
            SOM_t_plus_1_test = SOM_t_plus_1_test.view(-1, 1)
        if SOM_pred_test.dim() == 1:
            SOM_pred_test = SOM_pred_test.view(-1, 1)

        loss_test = loss_function(SOM_pred_test, SOM_t_plus_1_test)
        r2_test = r2_score(SOM_t_plus_1_test.numpy(),
SOM_pred_test.numpy())
        print(f'Test Loss: {loss_test.item()}')
        print(f'Test R^2: {r2_test}')

    # Displaying the estimated values of a_pred and b_pred for the best
run
    print("Estimated values for the best run:")
    print(f"a_pred_test (mean): {a_pred_test.mean()}")
    print(f"b_pred_test (mean): {b_pred_test.mean()}")

    # Log all values of parameters
    log_parameters(log_parameters_file_path, a_pred_test, b_pred_test)

    # Log results for the current iteration
    log_results(log_results_file_path, best_hidden_size,
best_hidden_size_g, best_val_loss, loss_test, r2_test,
                a_pred_test.mean(), b_pred_test.mean())

    # Plotting SOM_pred_test vs SOM_t_plus_1_test
    plt.figure(figsize=(10, 6))
    plt.scatter(SOM_t_plus_1_test, SOM_pred_test, alpha=0.5,
label='Predicted')
    plt.plot(SOM_t_plus_1_test, SOM_t_plus_1_test, color='red',
label='Ideal')  # Ideal line for reference
    plt.xlabel('SOM_t_plus_1_test (Actual)')
    plt.ylabel('SOM_pred_test (Predicted)')
    plt.title('Predicted vs Actual SOM')
    plt.legend()
    # plt.show()

    # Save the plot with a unique filename for each iteration
    plot_filename = os.path.join(scatter_plots_folder,
f'SOM_plot_iteration_{iteration + 1}.png')
    plt.savefig(plot_filename)
    plt.close()  # Close the plot to avoid display in subsequent
iterations

    print(f"Plot saved: {plot_filename}")
```

Acknowledgements

At the end of these five years of academic journey, I would like to express my most sincere gratitude to all the people that were a part of it.

First, I want to thank my supervisor, Prof. Vincenzo Riggio, for his support during the preparation of this thesis, despite the physical distance. I would also like to thank my co-supervisors at IST, Dr. Ricardo F.M. Teixeira and Prof. Tiago Domingos, for their precious guidance and insights during this project. Their mentorship has given me the chance to explore new topics and develop key skills that will undoubtedly benefit my professional future.

Of course, my deepest gratitude has to go to my whole family, my parents, my sister and my grandparents, for their unconditional love and support and for always believing in me, every step of the way.

I am extremely grateful to Matteo, for always being by my side, for his help through every difficulty, and for the calmness that he was able to give me every time I felt lost.

Then, a heartfelt thanks needs to go to all the friends that I have known for a lifetime, for growing with me and always being a fundamental source of support. Also, I am deeply thankful to all the people that I have met during these five years in Torino, for sharing this journey with me and for making it a little easier every single day.

Finally, I need to thank Lisbon, and all the people that I had the pleasure to meet during my semester abroad. It has been the perfect conclusion to a wonderful period of my life, it helped me grow personally and I will always remember it as one of the greatest experiences that I ever had.