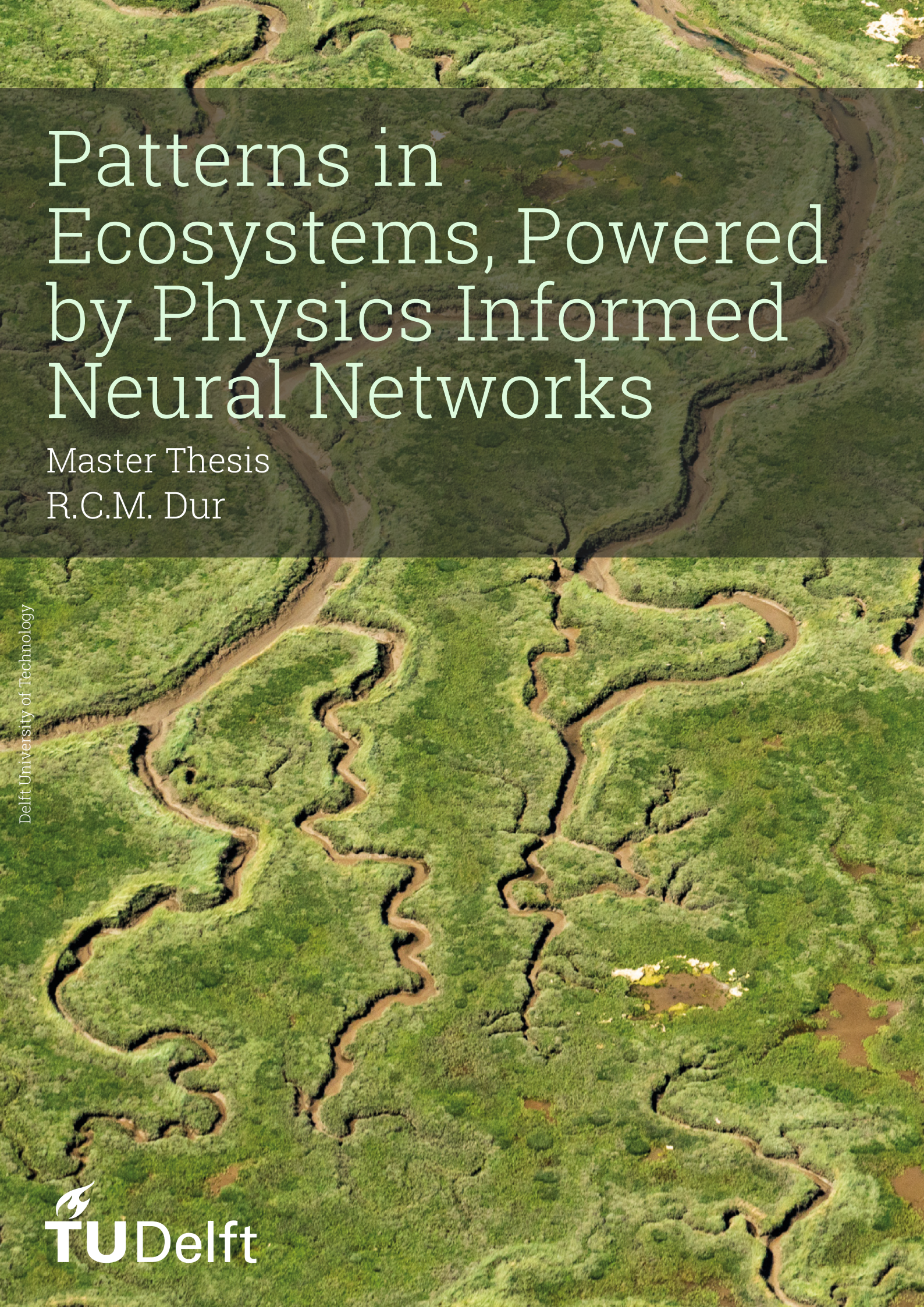


Patterns in Ecosystems, Powered by Physics Informed Neural Networks

Master Thesis

R.C.M. Dur

Delft University of Technology

TU Delft

Patterns in Ecosystems, Powered by Physics Informed Neural Networks

by

R.C.M. Dur

to obtain the degree of Master of Science

at the Delft University of Technology,

to be defended publicly on Monday September 21, 2026 at 14:00 PM.

Student number:	5086515	
Project duration:	November 13, 2025 – September 21, 2026	
Thesis committee:	Dr. M. Weinmann,	TU Delft, thesis advisor
	Dr. M. Skrodzki,	TU Delft, supervisor
	Prof. dr. J. van de Koppel,	NIOZ
	Dr. J. Baaijens,	TU Delft

Cover: Photograph by Edwin Paree (NIOZ)

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft

Preface

This thesis marks the end of my Master's, and thereby my time as a computer science student at the Delft University of Technology. By looking at the cover of this thesis, you may not have gotten the impression that this is a thesis in the field of computer science. During this thesis, I've come to learn a lot about tidal channels, salt marshes and physical interactions between water, sediment and vegetation and how the resulting landscape patterning can be crucial for the survival of many species, as well as for ourselves. My work has focussed on building a new and more powerful type of simulation to predict how tidal landscapes evolve, so that we can better understand them. The multidisciplinary character is what particularly pulled me towards this topic.

This topic is somewhat related to my Bachelor's end project, where I studied spontaneous and self-organised pattern formation in a more abstract context. The realisation that these techniques can also be used to explain the formation of patterns in long-stretching landscapes was fascinating to me. Especially since these patterns are important for biodiversity, health of (marine) ecosystems, storage of carbon, and protection of our infrastructure. The ability to make faster and more accurate predictions about these patterns is therefore highly relevant and therefore very interesting to work on.

For their enthusiasm and support throughout this project, I want to thank my supervisors Michael Weinmann and Martin Skrodzki from the Computer Graphics and Visualisation group. Also, I want to specially thank my external supervisor Johan van de Koppel for introducing me to the Dutch National Oceanographic Institute (NIOZ), and for helping me understand the processes behind tidal channel network formation. I enjoyed being invited to interesting events surrounding this topic, giving me the opportunity to meet several researchers in this field, for which I want to thank all my supervisors.

Finally, I would like to thank my family and friends, for supporting me throughout this journey. Especially my former DreamTeam colleagues, who were working on their own theses in parallel.

R.C.M. Dur
Delft, September 2026

Abstract

Multi-scale branching tidal channel networks carry several important functions for coastal wetland ecosystems, as well as for their surroundings. Many species living in and around the ecosystem are supported and protected by the channel network, as the channels carry important nutrients and resources. Furthermore, the channels increase the resilience of the landscape against sea-level rise by transporting sediment. Accurate predictions and understanding about the formation of these channel networks is therefore crucial. The speed and accuracy of numerical simulations is limited by, for example, temporal and spatial scale-differences between the processes in the landscape and working with realistic boundary shapes is tedious. Physics-Informed Neural Networks (PINNs) can produce mesh-less and differentiable solutions, while providing fast inference and flexibility towards boundary shapes. These properties can be highly beneficial to coastal wetland simulations, as the continuous nature of their solutions can aid in bridging the spatial and temporal gap between the different landscape-processes. In this work, we embed a complex multi-scale hydro-morphodynamical system to capture the formation of tidal channels into a physics-informed framework. We show that a PINN can be effectively trained, without data, to solve the 2D Shallow Water Equations (SWEs) in a closed water basin with flat bottom topography. The network solves these three coupled equations with steps of one second per iteration and is capable of working in complexly shaped domains with non-trivial obstacles. Effective training relies on oscillators introducing surface-disturbances. We propose that a PINN can further be trained, without data, to solve the 2D SWEs in a partially open landscape, by employing a numerical solver for the bottom topography and vegetation. A simple Convolutional Neural Network (CNN) is designed to translate from the discrete numerical solutions to the mesh-less representations of the PINN. We show that the PINN trains more effectively on landscapes with a high degree of spatial complexity. The accuracy of the hydrodynamic predictions made by the PINN influences the spatial complexity of the landscapes in its training set, and thereby the training effectiveness of the PINN itself. Important aspects of the hydrodynamic solutions are qualitatively accurate, and we show that they are detailed enough to trigger a channelisation feedback in the bed topography. Our physics-informed hydrodynamics are not sufficiently accurate to support the full ensemble of scale-dependent feedbacks behind the multi-scale tidal channels. We propose an ensemble of three networks to solve the morphological equations, in addition to the hydrodynamical ones, by means of a PINN. Learning effectiveness of one network in the ensemble heavily depends on the quality and accuracy of the partial solutions of the neighbouring networks. Our multi-network setup facilitates multi-stage or asynchronous training, which could potentially lead to more effective training. By showing that a PINN can be trained without data to find fast, reasonably accurate, consistent and flexible solutions, we set the stage for an improved hydrodynamical solver that can be coupled to both a numerical and a physics-informed solver for the corresponding morphodynamics to simulate the complete hydro-morphodynamical system behind tidal channel formation. This could potentially also be used in other hydrodynamical applications.

Contents

Preface iii

Abstract v

1 Introduction 1
- 1.1 Tidal Channels in Coastal Wetlands . . . 2
- 1.2 Surrogate Modelling . . . 3
- 1.3 Thesis Contributions . . . 3

2 Related Work 5
- 2.1 Self-organisation in Coastal Wetland Ecosystems . . . 5
- 2.2 Modelling of Spatial Patterning . . . 6
- 2.3 Physics-Informed Neural Networks (PINNs) . . . 6
- 2.4 Refined Research Gap . . . 7

3 Background 9
- 3.1 Model for Channel Formation in Vegetated Coastal Wetlands . . . 9
  - 3.1.1 Channel Networks in Coastal Wetlands . . . 9
  - 3.1.2 Scale-dependent Feedback Recursion (SFERE) . . . 10
- 3.2 Hermite Spline Physics-Informed CNN . . . 14
  - 3.2.1 Hermite Splines . . . 14
  - 3.2.2 Implicit Field Descriptor . . . 15
  - 3.2.3 Training Pipeline . . . 15
- 3.3 Multi Objective Learning with Gradient Surgery . . . 16

4 Methods 17
- 4.1 Physics Informed Closed Water Basins . . . 18
  - 4.1.1 Shallow Water Equations . . . 18
  - 4.1.2 Virtual Closed Water Basins . . . 19
  - 4.1.3 Physics Informed Loss . . . 20
  - 4.1.4 Network Architecture . . . 21
  - 4.1.5 Training Procedure . . . 22
- 4.2 Physics-Informed Hydrodynamics in a Coastal Wetland . . . 23
  - 4.2.1 Water Management: Modelling Tides . . . 23
  - 4.2.2 Wetting Drying . . . 24
  - 4.2.3 Modelling Open Boundaries . . . 25
  - 4.2.4 Adjusted Network Architecture for Partially Open Domains . . . 25
  - 4.2.5 Non-Dimensionalisation . . . 26
  - 4.2.6 Dataless Training Environments for Tidal Landscape . . . 26
  - 4.2.7 Image Fitting CNN . . . 27
  - 4.2.8 Impact of Landscape Structuring on Training Performance . . . 27
  - 4.2.9 Training in Hybrid Configuration . . . 31
- 4.3 Physics-Informed Hydro-Morphodynamics . . . 32
  - 4.3.1 Physics-Informed Sediment and Vegetation . . . 32
  - 4.3.2 Multi-scale PDEs . . . 33
  - 4.3.3 Learning at Multiple Timescales . . . 34
  - 4.3.4 Multi Network Training . . . 34
  - 4.3.5 Training Procedure . . . 35

5 Implementation Details **37**
5.1 Re-implementation of SFERE in PyTorch . . . 37
5.2 Towards a Modular and Scalable Hermite Spline-PINN . . . 40
5.2.1 Increased Scalability and Readability . . . 40
5.2.2 Code Organisation . . . 41
5.2.3 Multi-Network and Multi-Phase training with Gradient Surgery . . . 42
5.3 Image Fitting CNN, Embedded in Training Pipeline . . . 44

6 Evaluation **45**
6.1 Physics Informed Hydrodynamics in Closed Water Basins . . . 45
6.2 Image Fitting CNN . . . 50
6.3 Physics Informed Hydrodynamics in Coastal Wetlands . . . 53
6.3.1 Analysing the Impact of Landscape Structuring on Training Effectiveness . . . 53
6.3.2 Hydrodynamics in Static Landscapes . . . 65
6.4 Physics Informed Morphodynamics in Coastal Wetlands . . . 72
6.4.1 Hybrid Morphodynamics . . . 72
6.4.2 Saltmarsh-PINN Morphodynamics . . . 77

7 Discussion **81**
7.1 Implications for Coastal Wetland Modelling . . . 82
7.2 Implications for Physics-Informed Modelling . . . 83
7.3 Implications for Scientific Computing . . . 84
7.4 Implications for Real World Challenges . . . 85
7.5 Limitations and Recommendations . . . 86

8 Conclusion **89**

References **91**

Figure Credits **97**

AI Statement **99**

A Additional Figures and Alternative Approaches **101**
A.1 SWE Flux Formulation . . . 101
A.2 Preliminary Ablation Study Results . . . 102
A.3 Analysing the Impact of Non-Dimensionalisation . . . 105
A.4 Hydrodynamics in Static Landscapes . . . 108
A.5 Hybrid Morphodynamics . . . 119
A.6 Saltmarsh-PINN Morphodynamics . . . 119
A.7 Automatic Random Domain Rotation . . . 129
A.8 PINN Learns Principal Component First . . . 130

1

Introduction

Spatial patterning occurs widely in nature, with structures ranging from waves and ripples on the shore (e.g Figure 1.1) to much more complex shapes like branches, fractals and spirals (e.g. Figure 1.2). Many of these patterns also appear in the animal kingdom, with skins covered in spots and stripes of varying sizes and colours (e.g. in Figure 1.3). Whether it be for camouflage or to give warning signals to predators or to help with regulating body temperature, the patterns have a clear purpose and form an essential survival strategy for these animals. However, evolution alone does not suffice to explain how these patterns formed, as was argued by D'Arcy Wentworth Thompson in his book *'On Growth and Form'* [63] in 1917. The physical processes underlying the formation of these patterns and structures plays a much more significant role.



Figure 1.1: Wind-blown ripples in sand. Photo "Sand ripples 2015" by Philip Ball, CC BY 4.0.



Figure 1.2: Romanesco vegetable, with a spiralling fractal geometry. Photo "Romanescu" by Rlunaro, CC BY-SA 3.0.



Figure 1.3: South Australian Female ornate boxfish (Aracana ornata) with a mixed pattern of spots and stripes. Photo "Boxfish Aracana ornata patterns" by the Birch Aquarium at Scripps, CC BY 4.0.

Because these types of structures and patterns appear so often in nature, humans intrinsically tend to make a distinction between patterns created by living organisms and those created by physical or chemical processes (like crystals). However, if you look closely, you will find many similarities between the patterns and structures created by organisms versus those created by wind or water. Philip Ball, in his book *'The Self-Made Tapestry: Pattern Formation in Nature'* [2], explains why these lively and lifeless patterns can be so similar and that this is not a coincidence. Nature employs a self-organisation strategy, through which the large scale patterns on animal skins and desert sand are formed by simple, small-scale or local interactions between its components. In 1952, Alan Turing described how a simple reaction-diffusion mechanism can suffice to explain how these seemingly complex natural patterns form [64]. These physical interactions are therefore the reason why these patterns form and evolution can help to understand why some patterns are more favourable than others for reasons of survival.



Figure 1.4: Salt marsh river network. Photo by Edwin Paree (NIOZ).



Figure 1.5: Mangrove in the Sian Ka'an Biosphere Reserve. Photo "Mangrove" by Thomas Peschak, CC BY-SA 4.0.



Figure 1.6: Mussel beds as seen from above. Photo by Norbert Dankers (Wageningen Marine Research).

Spatial patterning in nature goes far beyond individual organisms and small-scale systems. Entire ecosystems and landscapes can be shaped by local interactions that drive self-organisation [58, 6, 54, 42, 22, 3], ranging from fractal-like river networks (e.g. Figure 1.4) to mangroves (e.g. Figure 1.5) and mussel beds (e.g. Figure 1.6). And like for animals, this patterning and structuring carries an important bio- or ecological function as well. Self-organisation in landscapes and ecosystems can positively impact their functioning and increase their resilience to disturbances [3, 56, 35, 76, 38, 30, 47, 55]. When an ecosystem is disrupted, its physical structure can be affected, as can the availability of resources within the ecosystem. Its self-sustaining dynamics can be permanently altered, rapidly changing the living conditions for its inhabitants and possibly even impacting neighbouring ecosystems over longer periods of time. Examples of disruptive events are flooding, extreme weather conditions, human development, the spreading of diseases or sudden introduction of invasive species. Climate change may increase the frequency and intensity of such disruptive events. For this reason, it is crucial to create detailed understanding of the mechanics that drive spatial patterning in these ecosystems and how their self-organised structure can offer resistance against these disturbances.

1.1. Tidal Channels in Coastal Wetlands

Tidal channel formation in coastal wetlands is a notable example of spatial landscape structuring that positively impacts the ability of the wetland to resist disruptions. Coastal wetlands are typically situated in the transitional area between land and ocean. Water travels through large networks of upland areas, rivers, streams, wetlands and estuaries before it eventually reaches the sea and these transitional zones are collectively referred to as coastal watersheds. By filtering water and creating buffers and nurseries, these zones are crucial for the health of entire marine systems, as well as populations on land. Specifically, tidal channel networks in (salt-)marshes (a specific type of coastal wetland) provide crucial resources for their highly populated hinterland by supporting water flow through runnels and complexly branching channel networks. In addition, the fractal-like structure of the networks acts as a natural mitigation of risks associated with flooding and strongly influences the resilience of the salt-marsh to sea level rise by transporting sediment. This way, the tidal channels protect both the species inhabiting the coastal wetland itself, as well as the surrounding area. It is therefore essential to understand the mechanics behind channel formation in salt-marshes.

The Scale-Dependent Feedback Recursion (SFERE, [66]) model has been developed to capture the spatial structuring of coastal wetlands by means of recursive scale-dependent feedbacks between the hydrodynamics in the landscape, as well as erosion and deposition of sediment and growth and decay of vegetation (commonly referred to as (geo-)morphodynamics). By modelling these effects, the development of channel networks can be captured on multiple branching scales. Simulation of these processes comes with an array of numerical challenges that require expensive computation. For example, it requires coupling of five highly non-linear equations that describe processes on different spatial and temporal scales. Furthermore, the spatial extent of these landscapes requires enormous grid sizes. These challenges lead to high computational cost and long simulation times.

1.2. Surrogate Modelling

In cases where such long simulation times cannot be afforded, a *surrogate* model can be employed that aims to solve (a version of) the problem at a faster pace. This is often at the cost of accuracy or realism of the solutions. For example, the model can be simplified to reduce the numerical complexity at the cost of realistic conditions or physical accuracy. In the modelling behind SFERE, for example, it was decided not to include advective transport of sediment. This reduced complexity, but also removed the ability of the simulation to capture meandering of river channels.

An important class of surrogate models, in general, is driven by *Machine Learning* (ML). By training on available data, they can make accurate and fast predictions about the future state of a system, without suffering from large numerical instabilities. Useful data-sources can for example be satellite images, ground measurements, photos and videos or historical data. Any model that makes predictions based on data from sources like these are known as *data-driven models*.

For many PDE systems, however, the amount and quality of available data are insufficient for most ML models to effectively learn the underlying dynamics. The accuracy of their predictions are mostly limited by the quality of the training data. Gathering more or higher quality data is often challenging, for example for practical or safety reasons. Especially for extensive landscapes like coastal wetlands, it is extremely challenging to obtain (large amounts of) high-quality data due to the physical extent of the landscape and the decadal time-span of the channel formation.

With their introduction in 2017, *physics-informed deep learning models* [51, 52, 53] have demonstrated that ML models (specifically deep learning models) can be much more effectively trained by directly incorporating the PDEs of the underlying problem in the learning process. More recently, it has been demonstrated that *physics-informed neural networks* (PINNs) can be effectively trained *without* the use of any training data from the field. Such models are commonly referred to as *data-less models*. Instead of merely *augmenting* the training data, the PDEs form the *only* source of information during training. Such data-less PINNs have for example been used to solve the incompressible Navier-Stokes equations [72, 73], as well as the damped wave equation [73], both of which are described by a system of two coupled PDEs. Besides accelerating simulations, PINNs can offer mesh-less and differentiable solutions making them especially suitable in cases where grid- and discretisation problems lead to instabilities.

1.3. Thesis Contributions

This thesis aims to extend the concept of data-less PINNs to the realm of coastal wetlands by embedding the processes behind tidal channel formation. Concretely, the objective is to train a PINN to solve the system of five highly coupled and non-linear PDEs as formalised in SFERE [66] by extending the training strategies employed by [72] and [73].

The first part of this thesis focusses on the hydrodynamics inside a coastal wetland, for which the following two main research questions are considered:

RQ1 To what extent can a PINN be trained to accurately solve the 2D shallow water equations in a closed basin with a flat bottom topography and without vegetation?

RQ2 To what extent can a PINN be trained to accurately solve the 2D shallow water equations in a partially open landscape, with uneven bed topography and spatially varying vegetation, by adapting or extending the model and training procedures established in **RQ1**?

We then continue by also considering the morphodynamics, for which we consider the following additional main research question:

RQ3 To what extent can a PINN be trained to accurately solve the system of hydro-morphodynamics inside a coastal wetland landscape for the formation of tidal channel networks, by adapting or extending the model and training procedures established in **RQ2**?

The contributions made to these research questions in this thesis can be summarised as follows. It is demonstrated that the 2D SWEs can be solved inside a closed flat-bottom basin, by training the PINN on a small set of virtual domains with randomly positioned obstacles and oscillators. Fast inference enables time-steps of $\Delta t = 1$s per iteration and the PINN is capable of generalising towards unseen

domains. Visual inspection indicates the presence of important hydrodynamical effects in our solutions, such as accurate reflections and interference of waves. Furthermore, the SWEs can be coupled to an uneven bed topography and vegetation in the landscape to train a PINN to solve the hydrodynamics inside a partially open coastal wetland landscape. We introduce an Image Fitting CNN to transform numerically generated pixel images to the mesh-less image representation used by the PINN. With this CNN, our PINN can be trained on starting conditions rendered by SFERE [66]. Furthermore, it facilitates a hybrid setup to solve the hydro-morphodynamical system with physics-informed hydrodynamics and numerically simulated morphodynamics. This hybrid setup can also be used during training. Landscapes with more complex spatial features facilitate more effective training. To solve the complete hydro-morphodynamical system with a PINN, we propose a multi-network setup to mitigate training issues related to the spatial scale-difference between the landscape processes. Furthermore, this setup allows for multi-phase or asynchronous training. Such training techniques have not been tested as part of this thesis, however the setup has been designed to facilitate them.

The remainder of this thesis is structured as follows. Chapter 2 presents an overview of works that are adjacent to the topic of this thesis, or works that are otherwise relevant, to refine the research gap outlined above. From these, three works are particularly important to understand before reading the thesis, and these are reviewed in more detail in Chapter 3. All physics-informed models developed in this thesis, as well as training strategies and experiments, are described in detail in Chapter 4 and some of the most important implementation details are highlighted in Chapter 5. Our results and findings are presented in Chapter 6, and discussed from the perspective of our research objectives. Chapter 7 continues the discussion by placing our findings in the context of the surrounding scientific and societal fields. Finally, we conclude the thesis in Chapter 8.

2

Related Work

Spatial patterning in nature in general has been a topic of interest for centuries. It was only in 1917 that D'Arcy Wentworth Thompson argued for a mathematical and physical explanation for the formation of these patterns, as oppose to a purely evolutionary one, in his book *'On Growth and Form'* [63]. His arguments were later reinforced by Alan Turing in 1952 who mathematically showed that two chemicals, reacting with each other and diffusing, could explain the spontaneous formation of spatial patterns, nowadays referred to as 'Turing Patterns' [64]. This reaction-diffusion model became a widely used framework for explaining and understanding spatial pattern formation in individual organism and small-scale systems [33, 34, 41]. Section 2.1 explains spatial patterning in the context of coastal wetlands and explains why self-organisation is important in those ecosystems. Next,

2.1. Self-organisation in Coastal Wetland Ecosystems

Self-organisation extends into the realm of entire ecosystems and landscapes as well [58, 6, 54, 42, 22, 3] and this structuring of landscapes can impact their functioning and increase the resilience of ecosystems to disturbances [3, 56, 35, 76, 38, 30, 47, 55]. Tidal channel networks in coastal wetlands are an important example of spatial patterning that is crucial for the functioning and resilience of the wetland [58, 76, 24, 62, 10, 57]. Their ecosystem services have benefits for species living both in and around the landscape, including humans.

Concerns are raised about the effects of climate change on coastal wetlands, as sea-level rise is predicted to result in significant loss of wetlands [8]. However, spatial patterning in ecosystems has also been argued to play a significant role in evading catastrophic tipping-points [55], which speaks to the ability of the coastal wetland to resist disruptions such as sea-level rise to the ecosystem. For this reason, it is important to understand exactly how the spatial structuring of coastal wetlands impacts the dynamic of the landscape under changing external conditions.

Accurate and fast simulations of coastal wetlands are important for reasons other than prediction of spatial patterning and landscape evolution over time as well. The buffers formed by coastal wetlands play an important role in protecting us from hazards like flooding or storms [78, 43, 81] and the impact associated with such events has grown with the increasing amount and value of coastal infrastructure and increasing population [61, 78]. On top of that, the effects of climate change can increase the significance and frequency of these events. Modelling the hydrological dynamics associated with flooding or tsunamis is challenging for reasons similar to the aforementioned biological self-organisation models, as both rely on complex interactions between the hydrodynamics and the soil.

2.2. Modelling of Spatial Patterning

The formation of tidal channels in coastal wetlands is driven by a scale-dependent feedback caused by biogeomorphic interactions [66] between water flow (hydrodynamics), sediment and sediment-stabilising organisms such as biofilms [76], algal mats [10] and plants [62, 60]. Modelling the biological self-organisation that results from these scale-dependent feedbacks results in single-scale Turing-like patterns [34, 54, 35, 76] and can therefore not explain the multi-scale structures of complexly branching channel networks [66]. Modelling more realistic hydrodynamical and biogeomorphic processes requires taking into account the effects of tidal dynamics and sediment transport [66] and calls for expensive simulations, such as Delft3D [11, 12].

In an effort to overcome these computational limitations, the Scale-dependent Feedback Recursion model (SFERE [66]) was developed, which explicitly accounts for the channelisation feedbacks to capture multi-scale structuring of the landscape. Vegetation (here referring to various sediment-stabilising organisms) controls a large part of this feedback mechanism. Therefore, SFERE [66] makes use of idealised sediment and tidal dynamics to accelerate the simulation and is capable of generating tidal channels in much shorter timespans with graphics accelerated hardware.

Various works are based on the simulations by SFERE, including but not limited to modelling transitions from salt-marshes to reed-land ecosystems [65], salt-intrusion in freshwater wetlands [59], the effects of sea-level-rise [7] and photorealistic 3D visualisations [23].

In areas of risk-assessment, decision making and coastal management it is not uncommon to reach for surrogate models that make an effective trade-off between performance and accuracy to allow significantly faster computation at the cost of lower accuracy [20, 37, 4, 17, 19]. Many of such surrogate models require large amount of data for extensive calibration or training of machine learning models [14, 75]. The required data is often not available or not of sufficient quality for wider application.

2.3. Physics-Informed Neural Networks (PINNs)

PINNs form a relatively new realm of deep-learning models [51, 52, 53] that can be effectively trained on relatively small datasets by augmenting them with physical laws that are acting within the respective target problem. By directly including a series of Partial Differential Equations (PDEs) that describe the dynamical system as a learning objective, PINNs can integrate domain-specific knowledge while fitting the available data [39, 18]. Besides forward solving, PINNs can also be used for inverse problems [32], solving calibration tasks or performing parameter estimation.

More recently, it has been demonstrated that PINNs may even be trained without the use of any training data, by only learning the imposed physical laws encoded as PDEs [29, 9, 49]. For example, the incompressible Navier Stokes equations could both be solved by a physics-constrained neural network [72] (i.e. using discrete solutions and finite differences (FD) for gradient estimation), as well as in a physics-informed *Spline-PINN* [73] (i.e. producing mesh-less solutions to support auto-differentiation). Spline-PINN makes use of Hermite Splines as a mesh-less implicit field descriptor. As will further be explained in Section 3.2.1, the locality of Hermite Splines allows them to be processed by a convolutional neural network (CNN) for increased generalisation capabilities over traditional *fully connected* networks.

2.4. Refined Research Gap

Spatial structuring through self-organisation in landscapes, and specifically the formation of tidal channel networks in coastal wetlands, fulfils an important ecological role for inhabitants in and around the ecosystem, as well as a protecting role for the people and infrastructure nearby. Predictive simulations for spatial patterning and ecosystem resilience against disruptive events are expensive. Modelling the underlying hydro-morphological dynamics with higher accuracy and at a faster pace is crucial to understand the impact of spatial landscape structuring on both the ecosystem itself, as well as its surroundings. Faster models typically rely on large amounts of high-quality data, which are challenging – or even impossible – to obtain. With the rise of PINNs, it has been demonstrated that models can be trained without the need for such extensive training data. They provide fast inference speeds by making efficient use of hardware acceleration.

The aim of this work is to embed the complex multi-scale hydro-morphodynamical system described by SFERE [66] in this data-less physics-informed framework. Inspired by the use of Hermite Splines and CNNs in Spline-PINN [73], the three main research questions listed in Chapter 1 can be further refined as follows.

To answer the first main question **RQ1**, the following sub-question is considered:

RQ1a How can a Hermite Spline-PINN be trained to solve the 2D shallow water equations inside a closed water basin with no bottom topography or vegetation?

To answer the second main question **RQ2**, the following sub-questions are considered:

RQ2a How can the setup from **RQ1** be adapted to allow coupling of the hydrodynamical equations to an underlying layer of sediment and vegetation? And how can the connection to the landscape to the main tidal channel be modelled?

RQ2b How can such hydrodynamical PINN be trained to solve the 2D shallow water equations, coupled to the underlying bed topography and vegetation, in a partially open coastal wetland domain?

RQ2c To what extent are the hydrodynamics predicted by this hydrodynamical PINN sufficiently accurate to facilitate an accurate simulation of morphological process inside the landscape?

Finally, to answer the third main question **RQ3**, the following sub-questions are considered:

RQ3a How can the setup from **RQ2** be adapted or extended to capture all hydro-morphological processes in the landscape?

RQ3b How can such hydro-morphological PINN be trained to solve both the 2D shallow water equations, as well as the morphological ones in a coastal landscape?

RQ3c To what extent does simultaneous training of the hydro- and morphological processes facilitate effective learning of the individual processes?

3

Background

Having introduced the topics of self-organising patterns in nature and physics informed machine learning, this chapter provides an in-depth review of several works that form the foundation of this thesis. This starts by giving a detailed explanation of the mathematical model behind channel formation in coastal wetlands in Section 3.1, followed by a technical review of the Hermite Spline-PINN [73] in Section 3.2. Finally, we provide a detailed explanation of gradient surgery to more accurately enforce constraints during multi-objective optimisation in Section 3.3.

3.1. Model for Channel Formation in Vegetated Coastal Wetlands

The formation of tidal channel networks in vegetated coastal wetlands is triggered by scale-dependent feedbacks that arise from interactions between water flow and sedimentary and vegetative processes. To study the formation of these channels, and particularly how vegetation affects such spatial patterning, the SFERE model [66] was developed and it is publicly available online [69, 70] and reproduced here for completeness. To that end, Section 3.1.1 provides an explanation for the processes behind the formation of channel networks on a conceptual level. Section 3.1.2 then explains how these processes are modelled by SFERE.

3.1.1. Channel Networks in Coastal Wetlands

When there is no vegetation, water flowing along a gentle slope in the terrain experiences a channelisation feedback that is triggered by slight topographic depressions. Water flow is attracted towards these slightly lower-situated areas, thereby locally increasing the incision rate near the topographic depression. Flow concentrates in the growing topographic depression and diverges away from the areas in between channels. This mechanism is illustrated in Figure 3.1. Erosion of sediment is reduced in areas directly surrounding the channels, causing the topography to be slightly sloped in the directions perpendicular to the channel. This topographic gradient, pointing away from the channels, facilitates the formation of secondary channels via the same channelisation mechanism. As branching of channels continues, this recursive process eventually shapes complexly branching networks.

In a coastal wetland, the combination of ebb and flood flow causes similar channelisation feedbacks. The complexity of the branching networks increases as the ratio between incision and diffusion rates grows and ratio is known as the landscape Péclet number [45, 46]. As this ratio increases, the spacing between parallel ridges decreases and their banks steepen, leading to the incision of more branching channels. This leads to the incision of even more branches. In a terrestrial setting, vegetation is typically considered to reduce the Péclet number, and therefore the network complexity, as vegetation cover decreases runoff and increases soil diffusion through bioturbation (the physical moving around of sediment by plants and organisms in general). In the context of coastal wetlands, and throughout this thesis, we will jointly refer to any sediment-stabilising organisms (biofilms [76], algal mats [10] or plants [62, 60]) as vegetation, in line with SFERE [66]. These organisms amplify the aforementioned abiotic channelisation feedback and can therefore increase the channel network complexity [76, 62, 31, 40, 68]. In areas with dense vegetation, roughness of the sedimentary bed is typically higher, as

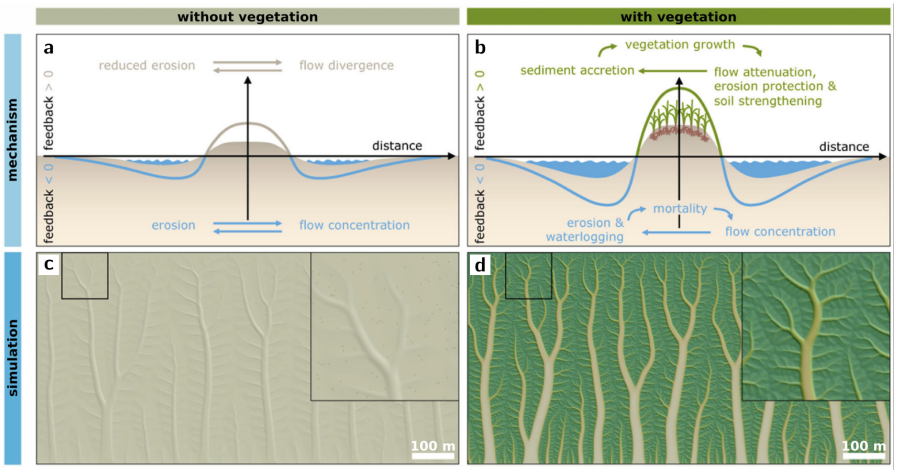


Figure 3.1: Vegetation influences tidal channel network formation by interacting with the abiotic channelisation feedback to increase the network complexity [66]. Figure adapted from van de Vijsel et al. (2023), CC BY 4.0. Panels a and b of the original figure (aerial imagery) have been omitted and panel lettering has been renumbered as [c-f] to [a-d] for clarity.

are erosion thresholds and sediment cohesion. This causes the water flow to decelerate, thereby promoting accretion of sediment in those areas. Water then diverges away from those areas, causing acceleration of flow around the patches of vegetation. Erosion rates will therefore increase in areas directly neighbouring patches of vegetation and this amplifies channel incision rates, as illustrated in Figure 3.1. Consequently, the vegetation patches will be situated on elevated terrain, where they experience better growth conditions. Altogether, these effects form a scale dependent feedback which, in combination with effects from tidal dynamics and sediment transport, can explain the multi-scale channelisation mechanism behind the fractal-like structures in salt marshes.

3.1.2. Scale-dependent Feedback Recursion (SFERE)

These water- and landscape processes are modelled by SFERE [66] in a complex multi-scale hydromorphodynamical system. The coastal wetland is modelled as a two-dimensional rectangular-shaped domain, bordered by topographic upland or dikes on three of its lateral boundaries. The fourth boundary represents the connection of the area to the main tidal channel and is modelled as an open-outflow boundary. In this model, ebb flow is assumed to be the dominant hydrodynamic driver. Consequently, this model assumes that ebb discharge is uniform in space and time, so that the entire tidal prism drains away steadily over the course of one tidal cycle. Concretely, this results in a spatially and temporally constant water source term H_{in} that effectively refills the tidal prism at precisely the rate at which it drains. H_{in} is estimated as the mean rate at which the tidal prism drains over a tidal period, and is hence related to the tidal range. The model neglects any effects caused by waves, representing a sheltered tidal marsh.

Water flow is prescribed by the depth-averaged Navier-Stokes equations, also referred to as the shallow water equations (SWE), and it is assumed that the temporal development of bottom topography S is relatively slow compared to variations in water layer thickness h, i.e. $\frac{\partial \eta}{\partial t} = \frac{\partial (h+S)}{\partial t} \approx \frac{\partial h}{\partial t}$, allowing us to write the continuity equation as:

$$\frac{\partial h}{\partial t} = -\frac{\partial (uh)}{\partial x} - \frac{\partial (vh)}{\partial y} + H_{in}. \tag{3.1}$$

The flow of water is based on pressure gradients, advection of momentum, bed friction and turbulent mixing (with eddy viscosity D_U), so that the flow velocity in horizontal (x) direction can be written as:

$$\frac{\partial u}{\partial t} = -g\frac{\partial (h+S)}{\partial x} - u\frac{\partial u}{\partial x} - v\frac{\partial u}{\partial y} - \frac{\tau_{bx}}{\rho h} + \nabla (D_U \nabla u) \tag{3.2}$$

and in the vertical (y) direction as:

$$\frac{\partial v}{\partial t} = -g\frac{\partial(h+S)}{\partial y} - u\frac{\partial v}{\partial x} - v\frac{\partial v}{\partial y} - \frac{\tau_{by}}{\rho h} + \nabla(D_U \nabla v) \tag{3.3}$$

where the bed shear stress components τ_{bx} and τ_{by} and their absolute value τ_b are given by

$$\frac{[\tau_{bx}, \tau_{by}, \tau_b]}{\rho} = \frac{g}{C_z^2}\sqrt{u^2+v^2}\left[u, v, \sqrt{u^2+v^2}\right] \tag{3.4}$$

and the Chézy coefficient C_z is given by Manning's formulation:

$$C_z = \frac{1}{n}h^{1/6} \tag{3.5}$$

$$n = n_b + (n_v - n_b)\frac{B}{k}. \tag{3.6}$$

Manning's bed roughness coefficient n is higher for vegetated areas than for bare sediment and is in this model assumed to linearly increase with vegetation stem density B. Roughness coefficients for bare and vegetated states are given by n_b and n_v, respectively, and k denotes the vegetation carrying capacity.

This model assumes concentrations of suspended sediment to be sufficiently high, so that availability of sediment is not limited by the lateral distance to channels. As the focus of SFERE is mainly on vegetative effects on tidal channel development, lateral advective sediment transport is neglected to speed up numerical computation. Consequently, the sedimentary bed elevation S is prescribed by a simplified balance between sedimentation, erosion and slope-driven sediment transport:

$$\frac{\partial S}{\partial t} = S_{in}\frac{h_e}{Q_s + h_e} - E_S\left(1 - p_E\frac{B}{k}\right)S\frac{\tau_b}{\rho} + \nabla(D_S \nabla S). \tag{3.7}$$

Over the course of simulation, some regions of the coastal wetland will accrete so much sediment that they prograde above the local water level. To ensure that the hydrodynamics can still be prescribed in those regions, SFERE employs a wetting drying algorithm to keep the water level above a critical threshold H_c:

$$h = \max(h, H_c). \tag{3.8}$$

The sedimentation rate is a measure of inundation time and asymptotically approaches a spatially uniform maximal value S_{in}. These sedimentation rates increase as a function of effective water layer thickness

$$h_e = h - H_c \tag{3.9}$$

and the rate at which the sedimentation rate increases with h_e is controlled by parameter Q_s. Erosion of sediment depends on the abiotic erodibility E_S. Trapping of suspended sediment by vegetation is expected to result in deposition patterns that are qualitatively similar to those arising from vegetation-induced bed roughness and erosion protection. Therefore, with higher sediment stabilising potential p_E of vegetation comes reduced erodibility. As erodibility generally decreases with depth due to consolidation, it is here assumed to be proportional to sediment elevation S. Finally, erodibility increases with higher bed-shear stress τ_b. Lastly, slope-driven sediment transport is modelled as a diffusive process, where the sediment-diffusivity

$$D_S = D_0\left(1 - p_D\frac{B}{k}\right) \tag{3.10}$$

is defined as the susceptibility to slope-driven transport and is inversely related to sediment cohesion. In line with previous studies, the abiotic sediment diffusivity D_0 is assumed to be reduced by below-ground vegetative biomass. There, p_D describes the potential of vegetation to reduce slope-driven transport.

Finally, vegetation stem density B changes as a function of growth, mortality and lateral dispersion:

$$\frac{\partial B}{\partial t} = rB\left(1 - \frac{B}{k}\right)\left(\frac{Q_q}{Q_q + h_e}\right) - E_B B\frac{\tau_b}{\rho} + \nabla(D_B\nabla B) \tag{3.11}$$

assuming logistic growth with rate r, that decreases asymptotically with the depth of the water. Mortality rates is dependent on the bed shear stress τ_b/ρ, where E_B is the mortality rate per unit shear stress. Lastly, vegetation is assumed to expand clonally and this is modelled as a diffusive process with diffusivity D_B.

The topography is initially flat and horizontal, meaning that the sedimentary elevation is zero everywhere ($S = 0$). As a consequence, flow velocities are zero throughout the domain ($u, v = 0$). There is a uniform initial layer of water with thickness H_0 and the domain is initially unvegetated, except for some randomly placed tussocks. These tussocks cover precisely one pixel that gets assigned the maximum carrying capacity (maximum stem density) and are placed with a probability of p_{est}. The result is a heterogeneous bed roughness that, together with the bed slope that readily arises from the sedimentation-erosion balance, induces drainage flow and triggers the scale-dependent feedback.

The right edge of this rectangular domain is modelled as an open-outflow boundary that represents the connection between the intertidal area and the main tidal channel. The water that accumulates as a result of the H_{in} term can leave the domain through this boundary. The gradient in flow-velocity is therefore assumed to be zero ($\frac{\partial^2 u}{\partial x^2}, \frac{\partial^2 v}{\partial x^2} = 0$) and there is assumed to be no gradient in water layer thickness and vegetation. Erosion and deposition of sediment are assumed to be in balance at the open boundary, such that sediment elevation always remains zero. All other boundaries are closed boundaries, where moving water is reflected. Furthermore, at the closed boundaries, there is assumed to be no gradient in water layer thickness, sedimentary elevation and vegetation density in the direction perpendicular to the boundary.

The SFERE model plays a significant role in the development of the various physics-informed models throughout this thesis. Whenever reference is made to a SFERE simulation, the parameter configuration from Table 3.1 is applied. These parameters largely match those used in the original model, with the exception of the domain and grid sizes, which have been altered to match the solutions produced by our PINNs.

Parameter	Value	Description
l_x	200 m	Domain length in x-direction
l_y	200 m	Domain length in y-direction
n_x	800	Number of grid cells in x-direction
n_y	800	Number of grid cells in y-direction
Δx	$l_x/n_x = 0.25$ m	Grid resolution in x-direction
Δy	$l_y/n_y = 0.25$ m	Grid resolution in y-direction
Δt	0.0125 s	Time step size
Φ	44712 s $[T_{M2}]^{-1}$	Morphological acceleration factor, calculated as the number of seconds in one M2-tidal period.
D_0	10^{-7} m^2 s^{-1}	Sediment diffusivity in the absence of vegetation
D_B	6.0×10^{-9} m^2 s^{-1}	Vegetation diffusivity (c̃lonal expansion rate)
D_U	0.5 m^2 s^{-1}	Turbulent eddy viscosity
E_S	2.5×10^{-4} s m^{-2}	Sediment erosion rate in absence of vegetation
E_B	1.0×10^{-5} s m^{-2}	Vegetation erosion rate
g	9.81 m s^{-2}	Gravitational acceleration constant
H_0	0.02 m	Initial water layer thickness
H_c	10^{-3} m	Critical water layer thickness
H_{in}	1.0×10^{-5} m s^{-1}	Water input (tidally averaged discharge velocity)
k	1500 m^{-2}	Vegetation carrying capacity (max. stem density)
n_b	0.016 s m$^{-1/3}$	Manning's roughness coefficient for bare sediment
n_v	0.2 s m$^{-1/3}$	Manning's roughness coefficient for vegetated sediment
p_D	0.99	Fraction by which sediment diffusivity is reduced when vegetation is at carrying capacity
p_E	0.9	Fraction by which sediment erosion is reduced when vegetation is at carrying capacity
p_{est}	0.002	Probability of vegetation seedling establishment
Q_q	0.02 m	Water layer thickness at which vegetation growth is halved
Q_s	6.0×10^{-4} m	Water layer thickness at which sediment input is halved
ρ	1000 kg m^{-3}	Reference water density
r	3.2×10^{-8} s^{-1}	Intrinsic plant growth rate (=1 year^{-1})
S_{in}	5.0×10^{-9} m s^{-1}	Maximal sediment input rate

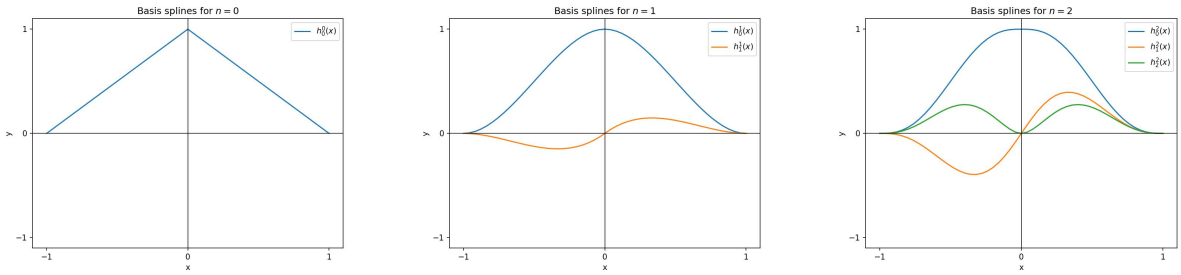
Table 3.1: Parameter configuration used for all simulations of SFERE [66] throughout this thesis. Values inspired by SFERE and altered to match the solution dimensions of our physics-informed models.

3.2. Hermite Spline Physics-Informed CNN

This section provides a detailed technical review of the Hermite Spline Physics-Informed CNN [73]. On that account, Section 3.2.1 gives an introduction to Hermite splines in general and Section 3.2.3 explains how they can be incorporated in a physics informed learning pipeline based on a CNN.

3.2.1. Hermite Splines

In its general form, a Hermite spline is a piecewise polynomial that is defined by its function value and first n derivatives at given support points. The support points are positioned at unit distance intervals (e.g. along a line or regular grid) and, in contrast to B-Spline kernels, their support ranges only over the directly neighbouring supports. In other words, the support of Hermite spline kernel functions is *localised*. Each support point, positioned at discrete $\hat{\mathbf{x}} \in \hat{\mathbf{X}}$, stores a series of spline coefficients $c^i_{\hat{\mathbf{x}}}$ that correspond to Hermite spline kernel functions $h^n_i(x)$. These are defined such that the values and the first n derivatives at the neighbouring support points $[x = -1, 0, 1]$ are set to zero, except the i^{th} derivative at $x = 0$ which takes a value such that $h^n_i(x) \in [-1, 1]$. Figure 3.2 shows an example of Hermite spline kernel functions in 1D for $n = 0, 1, 2$.



(a) First order basis spline (corresponds to linear interpolation).

(b) Second order basis splines.

(c) Third order basis splines.

Figure 3.2: Hermite spline basis kernel functions in 1D for $n = 0, 1, 2$. For each of these kernel functions, the $(n + 1)$-th derivatives are of bounded variation. Figure inspired by [73].

The total continuous piecewise polynomial is constructed by linearly combining Hermite spline kernel functions at the neighbouring supports and by choosing a spline order n, the $(n+1)^{\text{th}}$ derivatives are of bounded variation. It is important to choose an appropriate spline order, to accommodate the solution space of the given PDE. Kernel functions in multiple dimensions are constructed from a tensor product of multiple 1D kernel functions. For example, a Hermite spline kernel function with one temporal and two spatial dimensions is obtained as follows:

$$h^{l,m,n}_{i,j,k}(x, y, t) = h^l_i(x)h^m_j(y)h^n_k(t).$$

In this 3D setting, the support points are positioned on a grid $\hat{x}, \hat{y}, \hat{t} \in \hat{X}, \hat{Y}, \hat{T}$ and are represented by discrete spline coefficients $c^{i,j,k}_{\hat{x},\hat{y},\hat{t}}$ for each corresponding Hermite spline kernel function $h^{l,m,n}_{i,j,k}(x, y, t)$. The continuous Hermite spline function is then constructed as follows:

$$g(x, y, t) = \sum_{\substack{i,j,k \in [0:l]\times[0:m]\times[0:n] \\ \hat{x},\hat{y},\hat{t} \in \hat{X}\times\hat{Y}\times\hat{T}}} c^{i,j,k}_{\hat{x},\hat{y},\hat{t}} h^{l,m,n}_{i,j,k}(x - \hat{x}, y - \hat{y}, t - \hat{t})$$

where g can be used to approximate the solution to a (system of) PDEs $\mathbf{F}(\mathbf{u}, \theta) = \mathbf{0}$, with dependent variables $\mathbf{u}$ and parameters θ, with a degree of accuracy limited by spline order n. Partial derivatives of g with respect to x, y and t can be directly obtained by taking the corresponding derivatives of the spline kernel functions. This eliminates the need for Finite Differences (FD) or similar approximation methods to estimate gradients. The ability to directly compute gradients inside the domain is commonly referred to as *auto-differentiation* and forms one of the promising traits of this physics-informed setup.

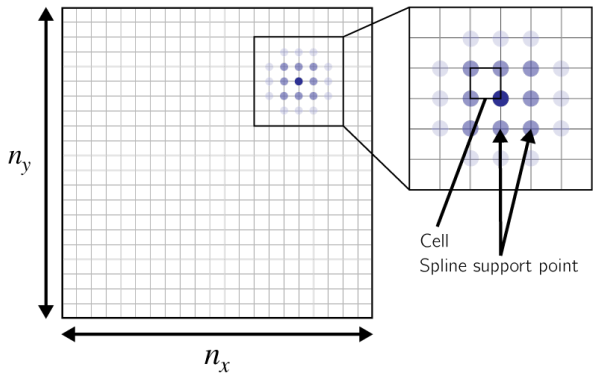


Figure 3.3: Hermite spline support points are positioned in a regular grid across the domain, for a total of $n_x \times n_y$ support points. The support points divide the area into a number of cells, where each cell is surrounded by precisely four support points: one on each corner.

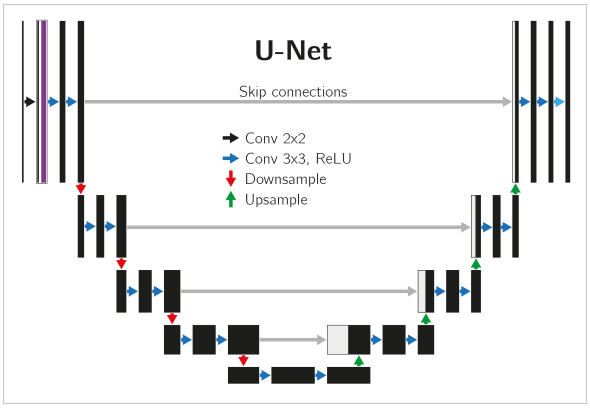


Figure 3.4: U-Net Convolutional Neural Network architecture that directly processes Hermite spline support points at time point t and produces the spline support points for the same environment, a time step Δt further in time.

3.2.2. Implicit Field Descriptor

By arranging the support points on a regular grid of size $n_x \times n_y$, as in Figure 3.3, they form a 2D field descriptor for the continuous solution at time t. The spline coefficients corresponding to the support points can be provided as input to a physics-informed neural network, to obtain the state of the solution at time $t + \Delta t$. Combining the support points at the network in- and output allows for continuous sampling in time as well. Because the spline coefficients are positioned on a regular grid, and because the support of the corresponding kernels only ranges over the directly neighbouring points, the Hermite Spline fields can be efficiently processed by a Convolutional Neural Network (CNN).

This way, the spline coefficients form an *implicit* field description: they represent a continuous field, as well as its derivatives, implicitly. In contrast, an *explicit* field description usually takes the form of a regular grid topology and relies on Finite Differences (FD) to approximate its gradients. Such explicit descriptor is for example used by [72] and is often used in physics-*constrained* approaches [80]. This Hermite Spline-PINN is of the physics-*informed* type [51], for it is based on an implicit field descriptor.

Implicit neural networks can be trained efficiently and they produce a continuous solution that can be evaluated anywhere in space and time. However, they often struggle to generalise towards domains unseen during training. To that end, this Hermite Spline-PINN combines the efficient training capabilities of a physics-*informed* neural network with the generalisation capabilities of a CNN. Specifically, we make use of a U-Net architecture, as is illustrated in Figure 3.4.

3.2.3. Training Pipeline

Training of this Spline-PINN CNN is done without the use of any training data from the field. Instead, the network operates on a series of virtual domains that are designed to match the underlying physical task. For example, the original Spline-PINN was trained to solve the incompressible Navier-Stokes equations in a section of a closed tube, and later also to solve the Damped Wave equations inside a closed environment with obstacles. The physics-informed training loss is based on the PDEs corresponding to the system, as well as its boundary conditions, and is computed across the volume (i.e. space and time) spanned by the spline coefficients of the training batch. After processing of each batch, the training pool is updated with the newly predicted states to fill the pool with more realistic data over the course of training.

This concept can be generalised to describe a Spline-PINN strategy for a system of PDEs of general form. While such generalisation is not explicitly described in Spline-PINN [73], they implicitly make use of this generalised idea by demonstrating that the same fundamental approach can be used for two distinct problems. Consider a system of PDEs of general form $\mathbf{F}(\mathbf{u}, \theta) = \mathbf{0}$ on a domain Ω, with dependent variables $\mathbf{u}$ and parameters θ, and an associated set of boundary conditions $\mathbf{B}(\mathbf{u}, \theta) = g(x, y, t)$ on $\partial\Omega$. The training pool consists of a 2D Hermite Spline field description for each of the

dependent variables $\mathbf{u}$ to implicitly capture the continuous state of the spatial domain (i.e. across the x- and y-dimensions) at time t. Furthermore, one or more boundary masks are stored alongside each entry of the training pool to allow for varying boundary placement. After processing of a training batch, the network in- and output form a continuous volume of the solution (i.e. in x- y- and t-dimensions) between times t and $t+\Delta t$ and this volume is uniformly sampled to obtain a set of training samples. For each sample, the physics informed loss is calculated, both for the PDEs and the boundary conditions of the system, after which gradient descend is used under an Adam optimiser (learning rate $= 0.0001$).

The physics informed loss is composed of one loss term for each of the PDEs in the system, as well as one term for each of the boundary conditions. In general, this yields two losses:

$$\mathcal{L}_{\text{PDE}} = \int_{\Omega} \int_{t}^{t+\mathrm{d}t} \|\mathbf{F}(\mathbf{u}, \theta)\|^2 \qquad \text{on } \Omega$$

$$\mathcal{L}_{\text{bound}} = \int_{\partial\Omega} \int_{t}^{t+\mathrm{d}t} \|\mathbf{B}(\mathbf{u}, \theta) - g(x, y, t)\|^2 \qquad \text{on } \partial\Omega$$

that are expressed as an integration over the volume spanned by the spline coefficients. All spline coefficients can initially be set to zero, eliminating the need for data from the field, and coefficients can also be reset to zero from time to time to learn the warm-up phase.

Throughout this thesis, we will make use of this generalised Spline-PINN form to construct the physics-informed models to solve the increasingly complex physical system as we move from closed water basins to hydrology in a coastal landscape to simulating the complete five-equation system.

3.3. Multi Objective Learning with Gradient Surgery

As will later become clear, one of the challenges in this work is to optimise for multiple learning objectives simultaneously. When there are multiple objectives (tasks) to solve simultaneously, a neural network may tend to accept solutions that are optimal for one task, but sub-optimal for others. The result is that one task can dominate over others, making the learning process unbalanced. Specifically, three conditions have been identified by [77] that can significantly degrade learning performance:

1. **When gradients from multiple tasks are in conflict with one another.**
   This happens when the angle ϕ_{ij} between the gradients of two tasks i and j exceeds 90 degrees. In this case, the gradients essentially point in opposing directions, making it difficult to find an optimisation step that improves the performance of both tasks.
2. **When the difference in gradient magnitudes is large, leading to some task gradients dominating others.**
   During an optimisation step, the mean gradient is used to steer the model towards the right direction. Consequently, if the magnitude of one task's gradients is much larger than that of other tasks, the mean gradient will be dominated by that task.
3. **When there is high curvature in the multi-task optimization landscape.**
   If the curvature of the optimization landscape in the direction of the multi-objective gradient is high, this can negatively impact the effectiveness of the optimisation process.

To mitigate these effects, a 'gradient surgery' strategy is proposed by [77], whereby the gradients of one task are projected onto the normal plane of the gradient of another task. By doing so, the conflicting component of the task's gradient is removed and this reduces the amount of gradient interference between the tasks. This gradient surgery approach is also referred to as *PCGrad*. A complete analysis of this technique is provided in the original work [77]. PCGrad can be used in combination with any gradient-based optimiser. During an optimisation step, the optimiser would simply make a step into the direction of the updated PCGrad gradients instead of the original (possibly conflicting) gradients. In practice, PCGrad is built as a wrapper around the original optimiser and instead of performing back-propagation on the total loss L_{tot}, we provide the PCGrad instance with an array of loss terms, after which it will apply back-propagation on each of the individual terms and perform surgery on the resulting gradients.

4

Methods

Inspired by the modelling of channel formation in salt-marshes proposed by SFERE [66], this section presents our approach for the embedding of a complex multi-scale hydro-morphodynamical system of five coupled PDEs into a PINN. Within this system, the first three PDEs capture the hydrodynamics and the remaining two PDEs describe erosion and deposition of sediment and the growth and decay of vegetation. Because these subsystems are coupled by terms in the PDEs, a solution to one equation directly affects that of the others. As the water dynamics form the main driving mechanism for the scale-dependent feedbacks that drive self-organisation in the landscape, we first describe the underlying conceptual approach for a simplified scenario in which only the hydrodynamical equations are learned in Section 4.1. Building on the insights obtained in this simpler setting, we provide the necessary details to couple the physics-informed SWEs to an underlying bed topography and vegetation in the landscape in Section 4.2. Finally, Section 4.3 covers the modelling of the complete salt-marsh system by means of an ensemble of PINNs. Figure 4.1 illustrates this breakdown and highlights some of the most important challenges and insights associated with each approach.

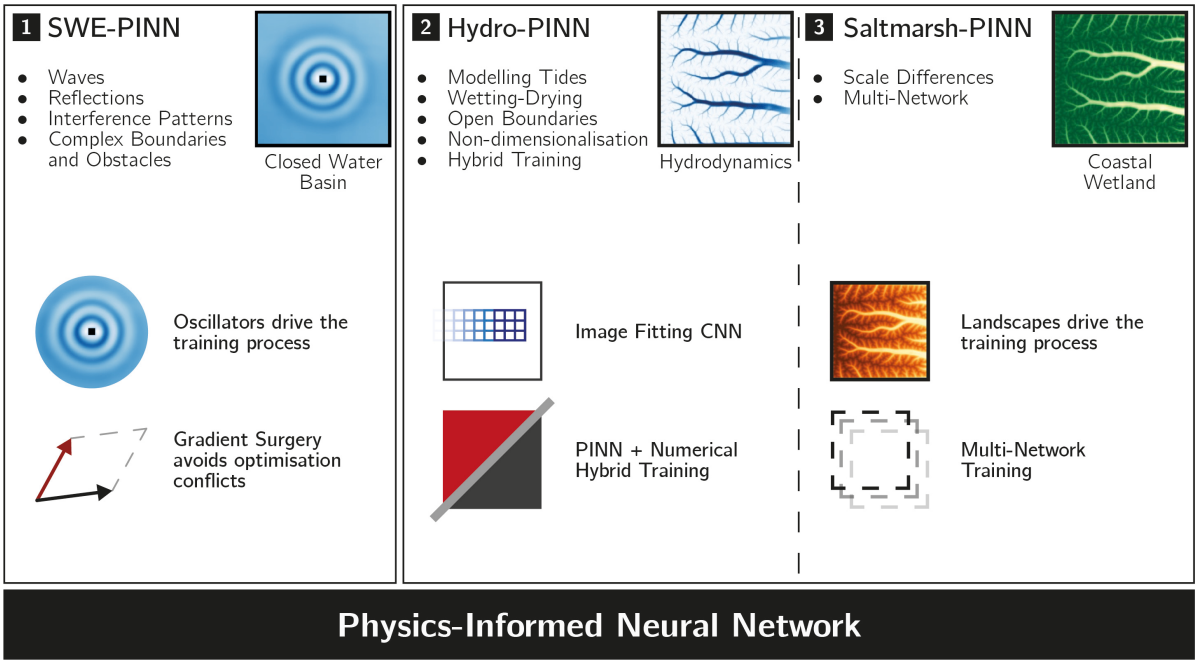


Figure 4.1: Breakdown of our design approach into three physics-informed models. We start by conceptualising the more simplified SWEs inside a closed basin before coupling them to a sedimentary bed and vegetation to finally arrive at the full salt-marsh model. Notable challenges for each concept are printed in italics.

4.1. Physics Informed Closed Water Basins

The hydrodynamics inside a widespread body of water are commonly formalised with the Shallow Water Equations (SWEs). These equations focus on the horizontal movement of water and SFERE [66] uses a variant that is coupled to a sedimentary bed topography and vegetation in the landscape. In this section, a Spline-PINN is designed and trained to solve the SWEs in a simpler form, without such coupling, inside a closed basin with no sediment or vegetation. Section 4.1.1 describes these simplified equations in detail. Directly afterwards, Section 4.1.2 describes the synthetic water basin environments on which the PINN is trained, including their initial- and boundary conditions. Section 4.1.3 then combines these PDEs and conditions to formalise the physics-informed loss terms that are used during training. Towards the end, Section 4.1.4 describes the network architecture for this setup, before Section 4.1.5 explains how this PINN can be trained. For the remainder of the thesis, we refer to this shallow water PINN model as the SWE-PINN.

4.1.1. Shallow Water Equations

The SWEs are derived from the depth averaged Navier-Stokes equations and describe the horizontal flow of water in a basin or reservoir where the average water depth is relatively much smaller than the size of the domain. With no sediment or vegetation in the domain, this system consists of a continuity equation:

$$\frac{\partial h}{\partial t} = -\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right)(h + H_0) - \frac{\partial h}{\partial x}u - \frac{\partial h}{\partial y}v - \epsilon h \tag{4.1}$$

and two momentum equations for the horizontal and vertical flow velocities u and v, respectively:

$$\frac{\partial u}{\partial t} = -g\frac{\partial h}{\partial x} - ku - u\frac{\partial u}{\partial x} - v\frac{\partial u}{\partial y} + \nu\nabla^2 u \tag{4.2}$$

$$\frac{\partial v}{\partial t} = -g\frac{\partial h}{\partial y} - kv - u\frac{\partial v}{\partial x} - v\frac{\partial v}{\partial y} + \nu\nabla^2 v \tag{4.3}$$

where H_0 is the mean water level and h the local deviation thereof, so that the water surface level at any given point is their sum $\eta(x, y, t) = h(x, y, t) + H_0$. The other parameters describe the gravitational acceleration g, the viscous drag coefficient k, the viscosity of the fluid ν and an artificial damping coefficient ϵ. This specific formulation is inspired by [1]. In principle, one could also consider an alternative formulation of the SWEs that is based on fluid fluxes, instead of modelling the flow velocities directly. For this, we would like to refer to Appendix A.1.

These equations differ to SFERE [66] in a few important places. Firstly, the continuity equation is rewritten to allow expressing the water layer thickness by the local deviation h of the mean water level H_0. Based on the PINN implementations by [73], it may be beneficial to learn h as a symmetric quantity around zero and it is therefore decided to follow their design in this regard. Section 4.1.4 sheds more light on the possible benefits of a zero-mean output. The momentum equations have been uncoupled from any bed topography and vegetation. Specifically, the bed friction terms $\tau_{bx}/\rho h$ and $\tau_{by}/\rho h$ are replaced by the viscous drag terms ku and kv, and the turbulent eddy viscosity D_U with a more general kinematic fluid viscosity ν.

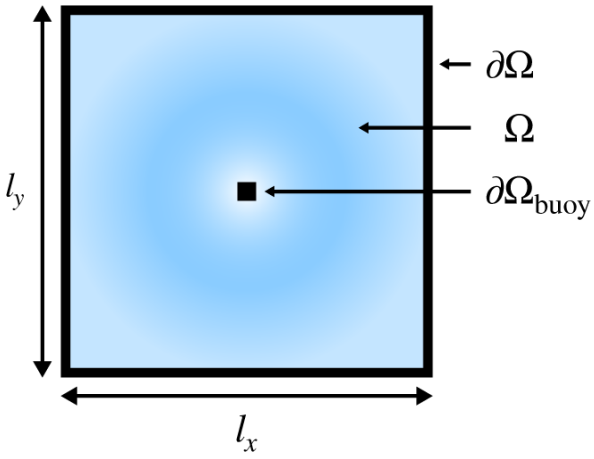
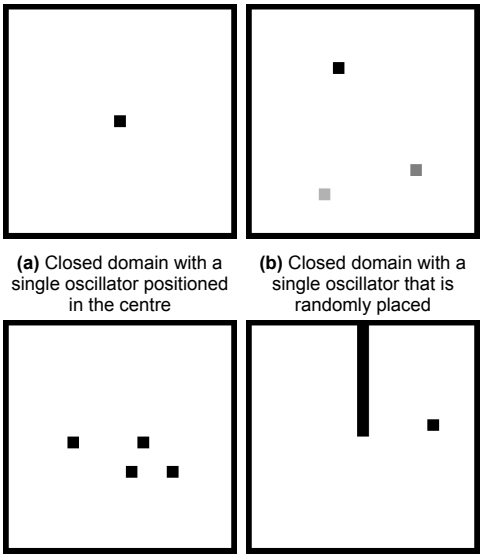


Figure 4.2: The water basin is l_x metres long and l_y metres wide. The domain Ω is surrounded by a closed boundary $\partial\Omega$ and one or more buoys may be randomly positioned within the domain and are represented by a buoy boundary $\partial\Omega_{\text{buoy}}$.



(a) Closed domain with a single oscillator positioned in the centre

(b) Closed domain with a single oscillator that is randomly placed

(c) Closed domain with multiple randomly placed oscillators

(d) Closed domain with a vertical barrier and a single oscillator positioned to the right of the barrier

Figure 4.3: Four types of water basin layouts, used during training of the SWE-PINN.

4.1.2. Virtual Closed Water Basins

The SWE-PINN developed in this section is trained without data from the field. Instead, the network learns by applying the SWEs from Section 4.1.1 inside a virtual water basin. The basin is square and spans an area of $l_x \times l_y$ metres, as illustrated in Figure 4.2. Initially, the basin is filled with a constant water layer of depth H_0 and the flow velocities u and v are zero. Waves are introduced by one or more randomly placed oscillating buoys and they are modelled with a *Dirichlet boundary condition*, to modulate the function value of h:

$$h(t) = A \sin\left(\frac{\pi}{10} t\right) \quad \text{Dirichlet boundary condition on } \partial\Omega_{\text{buoy}},$$

where A denotes the amplitude of the oscillation and t the time in seconds since the start of the simulation.

Waves encountering the edges of the domain should be reflected in the direction perpendicular to the boundary. As the domain is square, ideally this would be implemented by imposing two separate boundary conditions along the horizontal and vertical boundaries, respectively, where the vertical boundaries (at $x = 0$ and $x = l_x$) only reflect the horizontal component of the wave and the horizontal boundaries (at $y = 0$ and $y = l_y$) only the vertical component. However, as will later become clear, each boundary condition requires its own loss term and hence poses a separate learning task for the PINN. Having a large number of tasks can complicate the learning process, as the balancing of individual loss terms can significantly impact training productivity [77]. It is therefore decided to represent all four edges of the domain with a joint boundary condition on $\partial\Omega$, by assuming that the flow velocity at all boundaries should be zero in both x- and y-directions. This is commonly referred to as a *no-slip condition*, and is modelled with a Dirichlet condition on u and v:

$$u, v = 0 \quad \text{Dirichlet boundary condition on } \partial\Omega.$$

Parameter	Value	Description
H_0	1 m	Mean water layer thickness
A	0.5 m	Wave amplitude
g	9.81 m s^{-2}	Gravitational acceleration
ϵ	0.0001 s^{-1}	Damping coefficient
k	0.001 s^{-1}	Viscous drag coefficient
ν	0.5 m^2 s^{-1}	Kinematic viscosity of the fluid

Table 4.1: Overview of parameters used to train and evaluate a physics informed neural network on the system of shallow water equations.

The water level is assumed to be constant at the boundary, and this is modelled with a *Neumann condition* to prescribe that the gradient of h should be zero:

$$\nabla h = \begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{bmatrix} h = \mathbf{0} \quad \text{Neumann boundary condition on } \partial\Omega$$

where, again, the horizontal and vertical directions are combined into a single constraint.

Obstacles may be placed inside the domain and waves should reflect on their boundaries. These obstacles can take any shape, and they may blend into the outside boundary to form (long-stretched) barriers. Internally, these obstacles are modelled as part of the domain boundary $\partial\Omega$, for which they inherit the no-slip condition and their effect can be learned by means of the same loss terms. Furthermore, modelling them under the same boundary mask allows for creating more complex boundary shapes (e.g. circles, organic shapes, etc.). Four basin layouts are used during training and they are illustrated in Figure 4.3. The parameters used during training are included in Table 4.1.

4.1.3. Physics Informed Loss

The PDEs and boundary conditions described in the previous Section 4.1.2 form the physics-informed learning objectives analogous to those in the original Spline-PINN described in Section 3.2.3. Specifically, the continuity equation (Equation 4.1) translates to the continuity loss term L_h defined as:

$$L_h = \int_\Omega \int_t^{t+\mathrm{d}t} \left\| \frac{\partial h}{\partial t} + \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right)(h + H_0) + \frac{\partial h}{\partial x} u + \frac{\partial h}{\partial y} v + \epsilon h \right\|^2$$

and the momentum equations (Equation 4.2 and Equation 4.3), to terms L_u and L_v, respectively:

$$L_u = \int_\Omega \int_t^{t+\mathrm{d}t} \left\| \frac{\partial u}{\partial t} + g\frac{\partial h}{\partial x} + ku + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} - \nu\nabla^2 u \right\|^2$$

$$L_v = \int_\Omega \int_t^{t+\mathrm{d}t} \left\| \frac{\partial v}{\partial t} + g\frac{\partial h}{\partial y} + kv + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} - \nu\nabla^2 v \right\|^2.$$

Since there are two distinct boundary masks $\partial\Omega$ and $\partial\Omega_{\text{buoy}}$, the boundary loss is defined by two separate terms. The boundary loss on the edges of the domain $\partial\Omega$ is defined as:

$$L_{b,\text{closed}} = \int_{\partial\Omega} \int_t^{t+\mathrm{d}t} \left[\|u\|^2 + \|v\|^2 + \|\partial_x h\|^2 + \|\partial_y h\|^2 \right]$$

and for the boundary conditions within the buoy area $\partial\Omega_{\text{buoy}}$ as:

$$L_{b,\text{buoy}}(t) = \int_{\partial\Omega_{\text{buoy}}} \int_{\hat{t}}^{\hat{t}+\mathrm{d}t} \left\| h - h_{\partial\Omega_{\text{buoy}}}(t) \right\|^2.$$

All terms are combined into a total loss:

$$L_{\text{tot}}(t) = \alpha L_h + \beta(L_u + L_v) + \gamma(L_{b,\text{closed}} + L_{b,\text{buoy}}(t)). \tag{4.4}$$

where parameters $\alpha = 1$, $\beta = 1$ and $\gamma = 20$ control the weight of each term. This weighing of terms can significantly impact the training performance, as unbalanced losses can lead to poor performance on one or more terms, resulting in longer training times or degraded network performance. In addition to these weights, the gradient surgery multi-objective learning strategy from Section 3.3 is applied to find an even better balance among the different terms and prevent optimisation conflicts [77]. For this gradient surgery step, the terms are grouped into a continuity term, a momentum term and a boundary term:

$$\begin{aligned} \mathcal{L}_{\text{continuity}} &= \alpha L_h && \text{Continuity loss,} \\ \mathcal{L}_{\text{momentum}} &= \beta(L_u + L_v) && \text{Momentum loss, and} \\ \mathcal{L}_{\text{bound}} &= \gamma(L_{b,\text{closed}} + L_{b,\text{buoy}}) && \text{Boundary loss.} \end{aligned}$$

4.1.4. Network Architecture

This SWE-PINN uses a U-Net architecture identical to that of the original Spline-PINN, detailed in Section 3.2. The water layer thickness h is represented by a Hermite Spline field of order 1, and both flow velocity components with order 2, giving a total spline latent size of $(1+1)^2 + 2(2+1)^2 = 22$. The hidden size of the U-Net is chosen slightly larger at 25.

In the design of the output layer of the network, we can use the conservation properties inside the closed water basins to simplify the learning process. Recall from Section 4.1.1 that h is defined as the local deviation from the mean water column height H_0. As the total amount of water in the basin is conserved, the average value of h across the domain must equal zero:

$$\int_\Omega h(x, y) = 0 \quad \text{By definition of mass conservation}$$

and h is therefore a *zero-meaned* quantity. Instead of having to learn this conservation law, it can be enforced by subtracting the mean value of h in the output layer of the network. This directly prevents the PINN from adding or removing water, and this is commonly known as a *hard constraint*. A similar constraint can be imposed to enforce the conservation of momentum across the domain. By assuming that the amplitude of the waves in the domain A is much smaller than the average depth of the basin H_0, the fluxes $(H_0 + h)u$ and $(H_0 + h)v$, in x- and y-direction respectively, can be approximated as

$$\begin{aligned} (H_0 + h)u &\approx H_0 u \\ (H_0 + h)v &\approx H_0 v. \end{aligned}$$

As a consequence, the flow velocity components u and v are approximately linearly proportional to their respective momentum fluxes, e.g.

$$\begin{aligned} u \propto H_0 u \rightarrow u \sim (H_0 + h)u \\ v \propto H_0 v \rightarrow v \sim (H_0 + h)v. \end{aligned}$$

Under this assumption, the zero-mean property that fluxes $(H_0 + h)u$ and $(H_0 + h)v$ have inside a closed water basin propagate through to the flow velocity components u and v and the conservation of momentum can be enforced by subtracting the mean of both quantities after each iteration as well.

4.1.5. Training Procedure

The SWE-PINN is trained on a collection of 1000 virtual water basins, sampled from the four types illustrated in Figure 4.3. The first and most simple setup in Figure 4.2a contains just one oscillator that is placed in the centre of the domain and no obstacles. To train the network on different oscillator placements, the second type of basin in Figure 4.2b moves the oscillator to a random position. In the third type (Figure 4.2c), multiple oscillators are randomly placed and the last type contains a long stretched boundary and an oscillator next to it. All basins are randomly rotated upon initialisation of the dataset. After processing each batch, the output of the network is recycled back into the dataset to allow the network to learn over longer time spans and this is in line with Spline-PINN [73]. A basin may be reset if it has completed more than 5000 passes through the network. All buoys oscillate with an amplitude of $A = 0.5$ and a frequency of $f = 1/20$ s^{-1}, where one iteration through the network corresponds to a time step of $\Delta t = 1$s. Gradient surgery [77] is applied throughout the entire duration of training.

4.2. Physics-Informed Hydrodynamics in a Coastal Wetland

In this section, the focus shifts from closed water basins to hydrology in a partially open tidal landscape. Water behaves vastly differently in an open landscape compared to a closed reservoir, and the solution should be focussed on water flowing along ridges and runnels instead of surface waves interfering and reflecting. The network must learn how the water interact with the surrounding soil and biomass under repeated inundation and drainage of the tidal landscape. Following this continuous cycle of flooding and draining, the amount of water in the domain is no longer constant and the network must learn to deal with this (global) conservation-breaking property. Specifically, the network must learn how water can flow in and out of the domain through its connection with the main tidal channel.

To address these challenges, several adjustments are made to the training strategy as well as the network architecture and this updated model is further referred to as the Hydro-PINN. To model the effect of tides, Section 4.2.1 covers a water management strategy that is suitable for this PINN. To properly handle areas that fall dry, Section 4.2.2 explains in detail the wetting-drying approach that is used. Then, the connection of the wetland to the main tidal channel is modelled in Section 4.2.3. To accommodate these modifications, Section 4.2.4 explains how the network architecture has to be adapted to these adjustments and Section 4.2.5 explains why non-dimensionalisation is needed for a proper scaling of variables. Section 4.2.6 explains how this PINN can be trained on a collection of SFERE [66] images by making use of our Image Fitting CNN, explained in Section 4.2.7, to transform pixel images to spline latent space. An experiment is setup in Section 4.2.8 to investigate how different landscape images impact training. Finally, Section 4.2.9 explains how this PINN can be trained in conjunction with a numerical solver for the sedimentary bed topography and vegetation in the landscape.

4.2.1. Water Management: Modelling Tides

Coastal wetlands are usually connected to a section of the tidal channel that is sheltered from open sea. Large waves cannot reach this area, but the water level rises and falls with the tides. As the water level rises, the landscape floods until the majority of the wetland is covered. Sediment is carried in with the flood-flow and settles as the flow calms down. During a transition back from high to low tide, the water is transported by the channel network and drains from the landscape until it is almost completely dry. This is referred to as *ebb-flow*.

This constant wetting and drying is challenging to model numerically, as the SWEs tend to become unstable at near-zero water levels. In SFERE [66], it is therefore assumed that ebb flow is the dominating factor contributing to the self-organising processes, removing the need to model the complete rising and falling of the tides. Instead, the connection of the landscape with the main tidal channel is modelled as an everlasting outflow boundary where water can freely drain from the domain. Water is then uniformly and continuously injected across the landscape, and interacts with the surrounding soil and biomass in much the same way as tidal ebb-flow, before draining from domain.

Under this assumption, dry fall in the domain can be kept to a minimum by constantly adding a uniform layer of water to compensate for the drainage. This constant uniform injection of water can be modelled with the addition of a water source term H_{in} to the continuity equation

$$\frac{\partial h}{\partial t} = -\frac{\partial (hu)}{\partial x} - \frac{\partial (hv)}{\partial y} + H_{in} \tag{4.5}$$

and its value can be estimated as the mean rate at which the tidal prism drains [66]. The uninterrupted flow that results from this continuous injection and drainage makes for an elegant mathematical definition of the problem, as the solutions should eventually reach an equilibrium. For these reasons, this approach is considered a suitable water management strategy for this Hydro-PINN.

By inserting H_{in} in the continuity equation, however, any dryness can only be effectively compensated for once the PINN has learned its effects on the solution. To ensure that enough water is injected in the domain, the H_{in} term is implemented manually in this PINN, by adding a layer of thickness $H_{in} \cdot \Delta t$ between iterations. As this is a uniform layer of water, this can easily be implemented via addition to the first spline order of h. Excluding the water source term from the learning process via this manual approach has another advantage, namely that H_{in} can immediately be adjusted for landscapes with a longer or shorter tidal range without re-training.

4.2.2. Wetting Drying

Despite opting for a water strategy that keeps the extent of dry-fall to a minimum, dry areas will still appear in the wetland and it is crucial to implement a suitable coping mechanism for this dryness. SFERE [66] implements a thin-film wetting-drying algorithm that fixes the water level at a certain critical threshold H_c, and thereby prevents the domain from drying out altogether. There, it is a necessary burden to ensure that the hydrodynamics can continue to be prescribed in areas that have fallen dry. In this physics-informed setting, such wetting-drying strategy is not always required as the network learns from a series of randomly sampled collocation points throughout the domain [5]. After sampling, there are simply collocation points that are dry and others that are wet, and there is no need to provide explicit instructions for the boundary between them. By defining an optimisation objective in both the wet and dry areas, the network can automatically infer what should happen at the boundary.

Moreover, as a result of using Hermite Splines in this physics-informed approach, it is not possible to fully restrict the water levels to purely positive values. Even if we restrict the first order spline weights to positive values, the remaining orders may still carry the net water level into the negative number space in some areas. Nevertheless, certain terms in the physics-informed loss cannot be computed for water levels at or below zero. In this Hydro-PINN, we must therefore apply a thin-film algorithm ***after*** sampling of the Hermite spline fields. To prevent discontinuities in the gradients of the surface level, we make use of a $\mathrm{softplus}$ function:

$$\hat{h} = \mathrm{softplus}(h - H_c) + H_c.$$

Note how this does not affect the solution space of the neural network, but merely the loss function by which its performance is evaluated. Because this thin-film algorithm alters the solution ***after*** it was generated by the network, some portions of the output space of the network suddenly map to the same solution. In these regions, the loss function is no longer able to distinguish between solutions and may falsely inform the network of its performance with respect to the physical laws. Consequently, dry areas (i.e. where $h \leq H_c$) form a ***blind spot*** to the loss function and can lead to counterproductive learning. To keep these blind spots to a minimum, the first spline order of h is restricted to positive values and the $\mathrm{softplus}$ algorithm is only applied to terms in the PDE that cannot be computed directly with h. All other terms are computed with the original water surface level h as generated by the network. The flow velocity equations are therefore formulated as:

$$\frac{\partial u}{\partial t} = -g\frac{\partial (h+S)}{\partial x} - u\frac{\partial u}{\partial x} - v\frac{\partial u}{\partial y} - \frac{\tau_{bx}}{\rho \hat{h}} + \nabla(D_U \nabla u)$$

$$\frac{\partial v}{\partial t} = -g\frac{\partial (h+S)}{\partial y} - u\frac{\partial v}{\partial x} - v\frac{\partial v}{\partial y} - \frac{\tau_{by}}{\rho \hat{h}} + \nabla(D_U \nabla v)$$

where the Chézy coefficient, needed to compute the bed shear stress components $(\tau_{bx}, \tau_{by}, \tau_b)/\rho$ as in Equation 3.4, is updated to:

$$C_z = \frac{1}{n}\hat{h}^{\frac{1}{6}}.$$

The SWEs are known to become unstable at near-zero water levels. In areas that are nearly dry, the velocity terms u and v can reach enormous values and this can quickly make the simulation unstable. With the wetting-drying strategy described in this section, we allow the PINN to produce water levels that are near-zero, precisely zero or even slightly below zero. To prevent the velocity terms from exploding, we apply a moving boundary constraint that prescribes zero-flow in regions where the water level has fallen below a certain threshold:

$$u, v = 0 \quad \text{On } \Omega, \text{ where } h < H_c$$

where H_c refers to the same critical threshold that was used by SFERE to construct their thin-film.

4.2.3. Modelling Open Boundaries

As was mentioned in Section 4.2.1, the connection of the domain to the main tidal channel is modelled with an open outflow boundary. For that, the closed boundary is severed on one side and the open boundary is inserted. During training, the PINN must learn to model the free outflow of water from the wetland through this side of the domain. In SFERE [66], this boundary is implemented by imposing a new boundary condition:

$$\frac{\partial^2 u}{\partial x^2} = 0, \quad \frac{\partial^2 v}{\partial x^2} = 0 \qquad \text{on vertical boundaries, and}$$
$$\frac{\partial^2 u}{\partial y^2} = 0, \quad \frac{\partial^2 v}{\partial y^2} = 0 \qquad \text{on horizontal boundaries}$$

to assume that there is a constant gradient in flow speed through the open boundary. Including this constraint in the physics-informed model would require two separate boundary masks, for the horizontal and vertical boundaries, respectively. The ability to vary the placement of the open boundary during training is crucial, as CNNs can be sensitive to the absolute positioning of the open boundary in the domain [26]. However, such separation of boundaries can complicate the learning process by introducing additional learning tasks that need to be simultaneously optimised. In the design of the SWE-PINN, we mitigated a similar problem in Section 4.1.2 by merging the two directional constraints. Here, however, such approach would result in an overly restrictive condition that assumes zero gradient in flow speed in both directions, regardless of the boundary placement. As a result, the network may not be able to learn an ideal open outflow boundary at either side of the domain, resulting in less efficient drainage and eventually a build up of water in the landscape. As is explained in Section 3.1.2, the open outflow boundary is a crucial component for the formation of tidal channel networks in the landscape. Without it, water would never start flowing towards the main tidal channel and the scale-dependent feedback mechanism would not be triggered.

In a physics-informed learning setting, like the one considered here, the boundary conditions are implemented as *restrictions*: soft constraints represented by a term in the loss function that steer the network towards solutions that adhere to the imposed conditions. To say that a certain part of the boundary is open essentially means to have a *restriction-less* boundary through which the water can freely flow in and out of the domain. The implementation of such restriction-less boundary may thus be realised by completely excluding that part of the boundary from any imposed constraints. The network will experience no restrictions at the open boundary and learn to find an optimal solution that adheres to the physical laws inside the domain. For those laws to fully hold, automatically water must start flowing in and out of the domain. In the training of this network, we therefore exclude the open boundary from any flow velocity constraints. Zero-gradient in the water level h is still applied at the open boundary, in line with SFERE [66].

4.2.4. Adjusted Network Architecture for Partially Open Domains

The network architecture for this Hydro-PINN is almost identical to that of the SWE-PINN, described in Section 4.1.4, with the exception of a few changes to the in- and output layers. Firstly, since the hydrodynamical equations are now coupled to the underlying sedimentary bed topography and vegetation in the landscape, the quantities S and B are provided as additional inputs to the network. The second important change is a direct result of the partial opening of the landscape. As mentioned, in a partially open domain, mass is no longer conserved on the scale of the global landscape. The in- and outflow of water causes the average amount of water and the average direction of flow to be non-zero. All variables h, u and v are therefore no longer zero-meaned quantities, meaning that the hard constraints described in Section 4.1.4 can no longer be applied. The removal of these constraints opens a new *degree of freedom* for the network, as it becomes possible for it to mistakenly add or remove water to and from the domain and this may have implications on the learning process and the stability of the network during training and evaluation.

Furthermore, the continuity equation can no longer model h as the local deviation around the mean water level H_0, as in Equation 4.1, but has to be re-written as in Equation 4.5, where h now refers to the water column height, as measured from the underlying bed topography.

Quantity	Symbol	Approximate value range		Physical Unit
		min	*max*	
Water layer thickness	h	0	0.1	m
Flow velocity (x)	u	-1	1	m s^{-1}
Flow velocity (y)	v	-1	1	m s^{-1}
Sedimentary bed elevation	S	0	0.2	m
Vegetation stem density	B	0	1500	m^{-2}

Table 4.2: Smallest and largest pixel values per variable, as they appear in SFERE [66], by approximation.

Quantity	Symbol	Scaling Factor	
		First spline order	*Other spline orders*
Water layer thickness	h	2	2
Flow velocity (x)	u	5	2
Flow velocity (y)	v	5	2
Sedimentary bed elevation	S	2	2
Vegetation stem density	B	1500	500

Table 4.3: Scaling factor for first spline order and all remaining spline orders for each quantity in the system.

4.2.5. Non-Dimensionalisation

In traditional machine and deep learning, data normalisation is a common and important pre-processing step where the features of a dataset are scaled to similar magnitudes and ranges [21, 36]. However, for physics-informed methods in general and specifically for our data-less PINN, such normalisation is not always possible as the target solution to the physical problem is not available. In the case of our PINN, it is therefore important to ensure that the in- and output variables vary within a reasonable range [74].

With the introduction of morphological quantities S and B, a proper scaling of variables has become more important. Where h, u, v and S operate on approximately the same scale, the vegetation stem density B can attain values that are orders of magnitude larger. Table 4.2 illustrates this scale-difference by indicating the approximate value range for each variable. These values are based on outputs of SFERE [66]. Scaling the quantities according to these approximate ranges requires scaling of the spline coefficients $c^{i,j,k}_{\hat{x},\hat{y},\hat{t}}$ for each spline order separately. In this Hydro-PINN, each quantity is assigned two scaling factors: one for the first spline order and another for all remaining orders. These are defined in Table 4.3. The values in this table are deliberately chosen larger than the value-ranges indicated in Table 4.2, as we should be careful not to restrict the output range too much. Making the scaling factors too small can drive the network into regions of the activation function in the output layer of the network where gradients almost completely vanish and effective learning is compromised.

4.2.6. Dataless Training Environments for Tidal Landscape

The effectiveness of training the SWE-PINN, described in Section 4.1, on a small set of training environments can be largely attributed to the randomly placed buoys that generate waves in the domain. They are essentially the driver of the training process by introducing disturbances on the water surface. In an open landscape, the water dynamics are much more focussed on the flow of water and how it reacts to sudden changes in bed topography or pockets of vegetation. For that reason, the oscillating buoys are removed and the necessary disturbances to the water flow must instead be created by the underlying sedimentary topographies and vegetation profiles. The sediment and vegetation in the domain are thus the new *drivers* of the training process. As the intention is for this PINN to find hydrological solutions in a coastal wetland domain, a natural starting point is to train the network on a collection of landscape profiles that are generated by SFERE [66].

Before the discrete images produced by SFERE can be used as an input to train the PINN, however, they must be transformed to a corresponding Hermite spline representation. For this, a custom CNN is designed to fit the spline weights to a discrete image and this is explained in Section 4.2.7. Directly afterwards, in Section 4.2.8, an ablation study is setup using this image-fitting CNN to discover the impact of landscape structuring in a series of SFERE images on the training performance of the PINN.

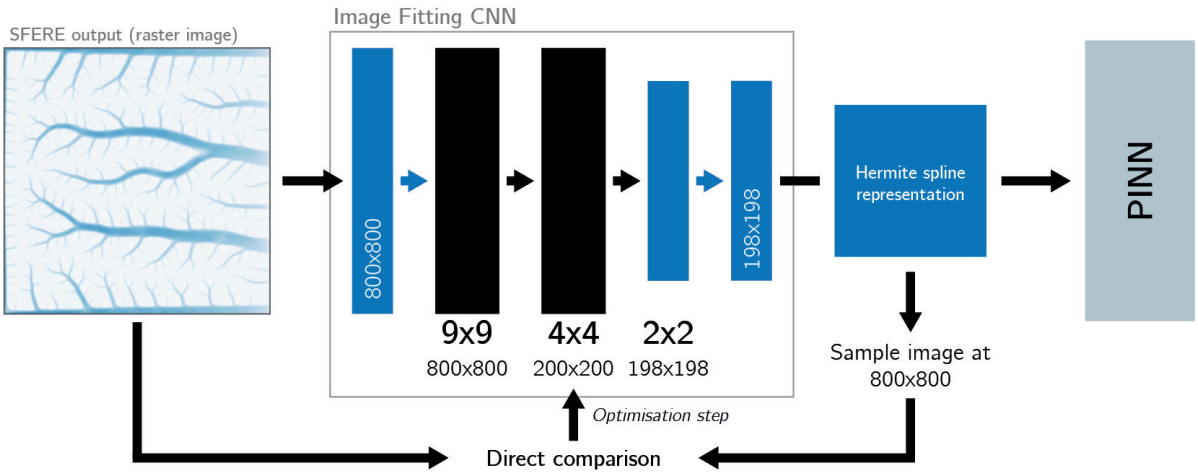


Figure 4.4: Raster images produced by the SFERE [66] are transformed to Hermite spline images with a Convolutional Neural Network. After training, these spline images can directly be used as (training) input for the PINN.

4.2.7. Image Fitting CNN

By design, this PINN only accepts Hermite spline images as input. Any discrete starting condition image must therefore first be transformed and this can be done with a CNN. The output of this Image Fitting CNN (ImFit-CNN) should match the input dimensions of the PINN, for which the CNN should produce spline fields of 198×198 support points, each consisting of the required number of spline weights to meet the chosen spline order for each variable. To ensure that enough detail is present in the images for the higher spline orders to properly fit to them, the network is configured to receive images with a resolution of 800×800 pixels as input. Each support point in the output spline image is therefore influenced by the surrounding pixels in the input image with a radius of 4 pixels. The CNN must therefore have a convolutional layer with a kernel size of at least 9×9 to capture sufficient information from the pixels surrounding each support point. A second convolutional layer downsamples this intermediate 800×800 output to a 200×200 image by applying a 4×4 kernel with a stride of 4 and no outside padding. The final spline weights are then obtained through one last convolutional layer with a kernel size of 2×2. This pipeline is illustrated in Figure 4.4.

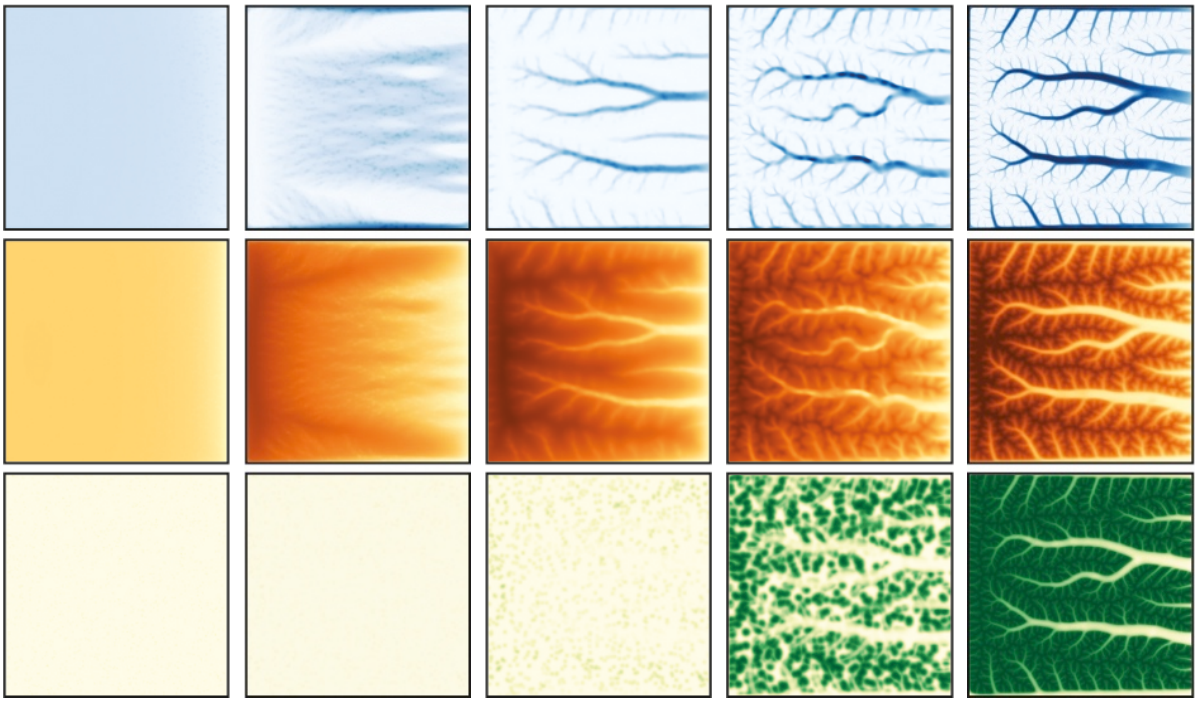
A separate ImFit-CNN instance is trained for each variable h, u, v, S and B in the system, by using a collection of outputs from the SFERE [66] simulation as training data. This training set is created by taking 100 snapshots of SFERE with an interval of 10.000 iterations. The training loss is calculated by sampling the spline output at a resolution identical to the input image and directly comparing it to the corresponding input. The training loss can therefore be formalised as:

$$L_{\text{imfit}} = \sum_{x,y} \|T(x,y) - P(x,y)\|^2$$

where T is the target image produced by SFERE and P the predicted output after sampling of the Hermite Spline field.

4.2.8. Impact of Landscape Structuring on Training Performance

Throughout a typical simulation of SFERE [66], five categories of landscape structuring can be observed and they are summarised in Figure 4.5. The images in that figure are generated with the parameters in Table 3.1. Starting from the initial condition, with flat topography and randomly placed vegetation tussocks, water starts to flow towards the open boundary and a slope gradually appears in the soil (Figure 4.5a). The sedimentary channelisation feedback process starts first, creating incisions at the open boundary that eventually grow out to wide channels that transport large amounts of water towards the outflow boundary (Figure 4.5b, Figure 4.5c). As the simulation continues, the vegetation tussocks start to grow and play a more significant role in the channelisation process (Figure 4.5d). Eventually, the system reaches an equilibrium and the multi-scale network channels have fully formed (Figure 4.5e).



(a) Stage 1: (0-30K iterations) Formation of slope in sediment topography

(b) Stage 2: (30K-100K iterations) Incisions form in sediment, flow concentrates

(c) Stage 3: (100K-250K iterations) Channelisation in sediment, further flow concentration

(d) Stage 4: (250K-500K iterations) Vegetation grows and impacts channelisation

(e) Stage 5: (500K-1M iterations) Equilibrium

Figure 4.5: Throughout the simulation, five categories of landscape structuring can be observed. Images are taken from a SFERE simulation using the parameters in Table 3.1. For each stage, the water layer thickness h (top row), sedimentary bed elevation S (middle row) and vegetation stem density B (bottom row) are illustrated. Landscape structuring experiences a logarithmic slowdown, causing later stages to last for a larger number of iterations.

What can also be seen in Figure 4.5, is that the development of channels and structures in the landscape is not a linear process. It takes a growing number of iterations to move from one category to the next, as the patterns in the landscape coarsen logarithmically. This logarithmic slowdown can be explained somewhat analogous to the Lifshitz-Slyozov-Wagner (LSW) theory for Ostwald ripening [13], where the growing rate of droplets slows dramatically as their radius increases. As the channel spacing in these coastal wetlands increases with the size of the channel, it takes longer for soil to redistribute and, as a result, further coarsening of the channels takes longer.

The simulation stages summarised in Figure 4.5 show different characteristics and to investigate how they impact the training performance of the PINN, an ablation study is setup where the network is separately trained on samples from each category. Comparing the performance of the five resulting networks may give valuable insights into the training process and can help to understand what aspects of the training environments can successfully guide the network towards suitable solutions. Throughout this section, these are referred to as **Hydro-PINN-Stage-1** through **Hydro-PINN-Stage-5** and they are trained on samples from the simulation stages 1 through 5 in Figure 4.5.

Early experiments suggest that large regions of dryness in the domain can negatively impact the performance of the PINN. The extent of dry regions naturally grows with the drainage rate of the landscape, especially when it drains faster than the water source term H_{in} can compensate for. To illustrate, training a PINN on the landscapes of stage 5 shows reasonable hydrodynamical solutions in Figure 4.8, where it can be seen that water gathers in the channel network and flows towards the open boundary. When trained on landscapes from stage 4, as in Figure 4.7, the flow velocity components u and v turn slightly chaotic as soon as all the water has drained. The worst performance can be observed in Figure 4.6 for a PINN that was trained on stage 2 landscapes. Already after 20 forward passes, the landscape structure has vanished from the solution and the tidal prism seems to have almost completely drained after 200 iterations. In response, the network predicts highly inaccurate flow velocities.

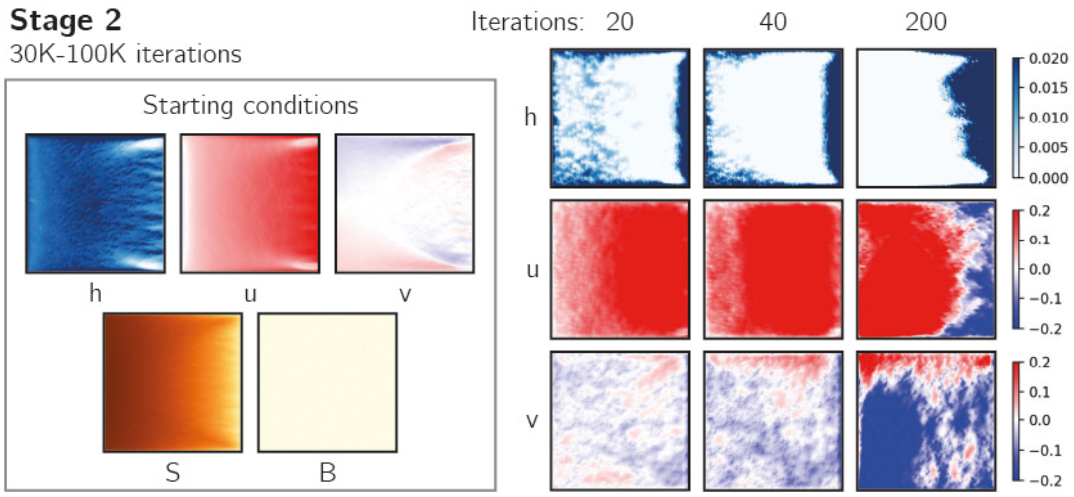


Figure 4.6: Preliminary hydrodynamical predictions of **Hydro-PINN-Stage-2** after 20, 40 and 200 iterations.

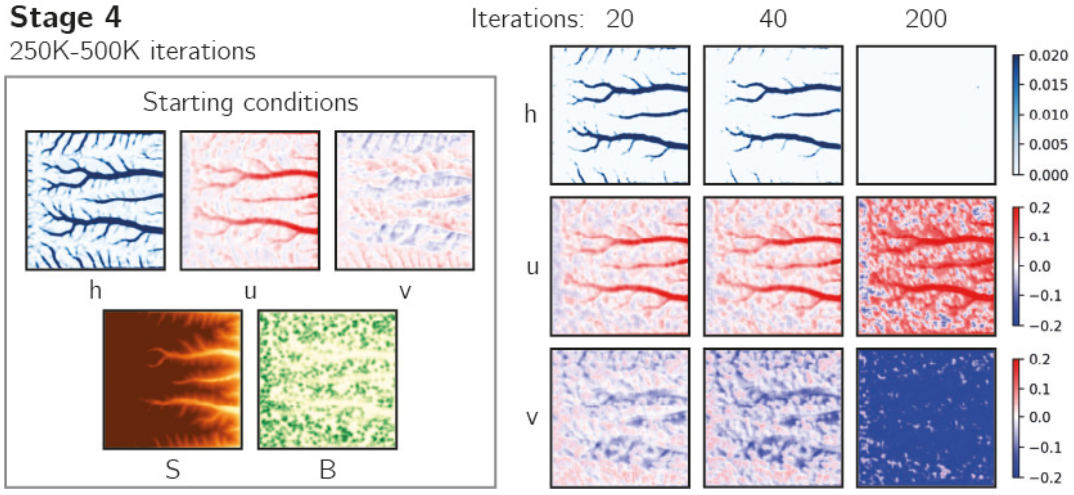


Figure 4.7: Preliminary hydrodynamical predictions of **Hydro-PINN-Stage-4** after 20, 40 and 200 iterations.

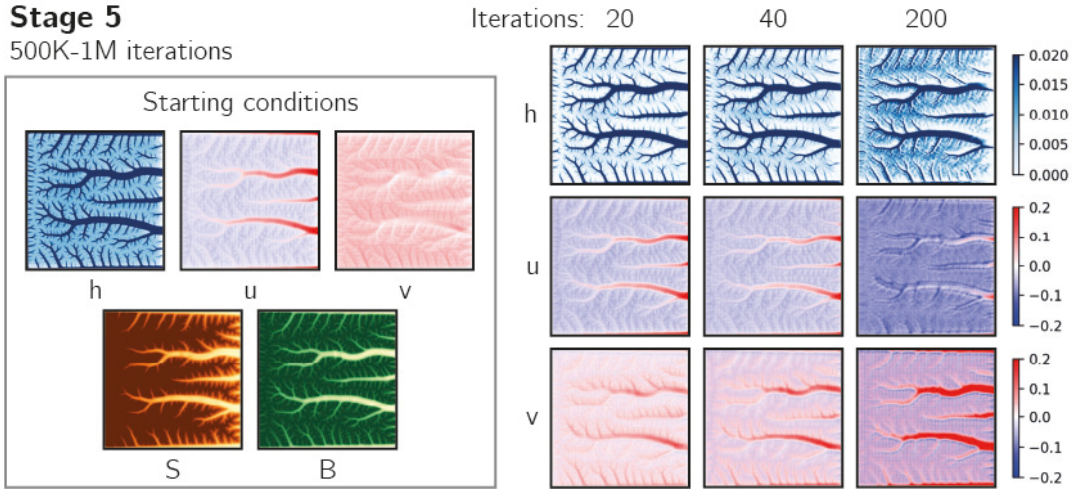


Figure 4.8: Preliminary hydrodynamical predictions of **Hydro-PINN-Stage-5** after 20, 40 and 200 iterations.

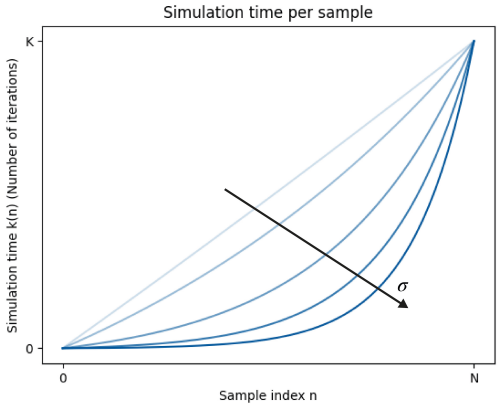


Figure 4.9: Simulation time per sample, modelled as an exponential function. The steepness of the curve increases with σ.

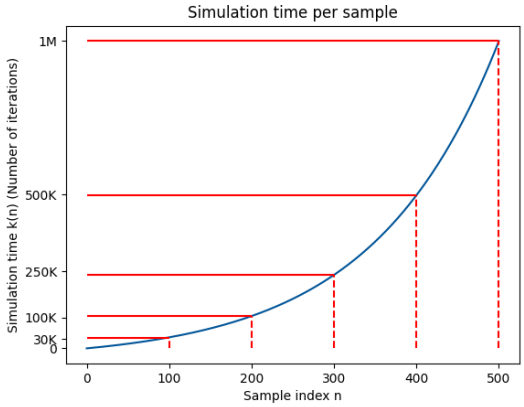


Figure 4.10: By choosing $\sigma = 3.3$, the exponential curve matches almost perfectly with the categorisation in Section 4.2.6: The samples are equally spread among the simulation ranges separated by 30K, 100K, 250K, 500K and 1M iterations of SFERE [66]

In these preliminary experiments, the water source term H_{in} was part of the continuity equation and therefore had to be learned by the PINN. Furthermore, any negative water surface levels were not compensated for, leading to deep 'wells' below ground that required a significant amount of water to fill, which likely amplified the negative effects of drought on the training productivity. In addition, the open boundary was poorly modelled, further details about which can be found in Appendix A.2. Lastly, training of the individual networks was slightly unfair as the training sets only contained snapshots of SFERE [66] with an interval of 10.000 iterations. Due to the logarithmic slowdown in the landscape structuring, described earlier in this section, later stages of landscape coarsening contained significantly more snapshots than earlier ones.

To substantiate these preliminary results, a confirmation study is setup with better-grounded choices for water management, wetting-drying, modelling of the open boundary, proper scaling of variables and a less biased distribution of training samples. Instead of learning the water source term H_{in}, the manual approach detailed in Section 4.2.1 is applied. In line with the wetting-drying strategy explained in Section 4.2.2, the network is free to produce negative water surface levels, however any negative water heights are always compensated for by clamping the first spline order of h to a minimum of zero. The open boundary is modelled as a restriction-less boundary that is not explicitly represented by a term in the loss function. This is in line with Section 4.2.3. In- and output variables are scaled according to the non-dimensionalisation of Section 4.2.5. Like before, the Image Fitting CNN from Section 4.2.7 is used to transform outputs of the SFERE simulation into spline-based training environments.

A less biased distribution of training samples is obtained by following an exponential timescale to compensate for the logarithmic slowdown in the landscape coarsening. For that, consider that the SFERE [66] simulation is run for a total of K iterations and that a total of N samples are recorded. The function $k(n)$ maps a sample index $n \in [0, N-1]$ to the number of simulation iterations $k \in [0, K-1]$ at which that sample should be taken. By choosing an exponential function for k, the landscape develops approximately equally quickly between consecutive snapshots, as the sampling frequency lowers drastically over time to match the logarithmic slowdown. We define $k(n)$ as follows:

$$k(n) = \left\lfloor \frac{\exp(\sigma n) - 1}{\exp(\sigma \cdot (N-1))} \cdot K \right\rfloor \tag{4.6}$$

where σ determines the *steepness* of the curve, as is illustrated in Figure 4.9. It is easy to see that for any choice of $\sigma > 0$, $k(0) = 0$ and $k(N-1) = K-1$ so that the function covers the full simulation timespan. By choosing a steepness value of $\sigma = 3.3$, the curve matches almost perfectly with the categorisation in Section 4.2.6. This means that the total number of samples K can be split into five equally sized groups that correspond to each of the output categories, as is illustrated in Figure 4.10.

In this confirmation study, a total of $K = 500$ snapshots are taken, meaning that each PINN in this study is trained on precisely 100 samples. During training, the domains are randomly rotated to remove as much asymmetry from the training set as possible. The PINNs only focus on the hydrodynamical part of the solution, meaning that the sedimentary and vegetative profiles remain fixed over the course of training. Following the preliminary observations in Figures 4.6, 4.7 and 4.8, we expect to see a difference in performance among the PINNs, depending on the category of landscape structuring that they are trained on. Landscapes with higher drainage-rates are expected to yield lower training performance, as a bigger portion of the landscape may fall dry. The water source term H_{in} is kept constant across all five PINNs and across the entire duration of training.

4.2.9. Training in Hybrid Configuration

In the previous section (Section 4.2.8), a training strategy is described whereby the network learns to solve the hydrodynamics inside landscapes with a sedimentary bed topography and vegetation density that remain fixed between training iterations. Because these landscapes are fixed, and because the tidal dynamics have been simplified to a constant outflow of water, as described in Section 4.2.1, the hydrodynamic solutions must eventually reach an equilibrium. And because the output solutions of the PINN are recycled back into the training set after each iteration, eventually the training set becomes saturated with converged landscape states.

Repeatedly training the network on the same converged set of landscapes will simply cause the network to memorise the solutions for those landscapes. As a result, the network will poorly perform on starting conditions other than those encountered during training. In the training procedure of the original Spline-PINN, as well as that of our SWE-PINN, the recycling of training environments is what allows the network to be effectively trained on such a small set of starting conditions. The output of the PINN fills the training set with large amounts of varying starting conditions by re-using the solutions that it had produced in earlier iterations.

With the Image Fitting CNN from Section 4.2.7, a training pipeline can be designed for this Hydro-PINN that enables updating of the training landscapes (i.e. sedimentary and vegetative layers) using a numerical solver for both morphological quantities. In such a hybrid setup, the landscapes on which the network is trained change in response to the the hydrodynamics that it had produced in earlier iterations and, as a result, the network is presented with a significantly larger pool of differing landscapes.

Because the sedimentary bed elevation and the vegetation density are represented by a Hermite Spline field, the hybrid numerical solver does not rely on Finite Differences to approximate the gradients in the landscape. Instead, the auto-differentiated gradients can be used to compute the corresponding PDEs and to perform an update step using the desired morphological acceleration factor. When the updated landscape has been computed, it can be transformed back into a Hermite Spline field via the Image Fitting CNN after which it can be immediately placed back into the training set. The pipeline for this hybrid training procedure is illustrated in Figure 4.11.

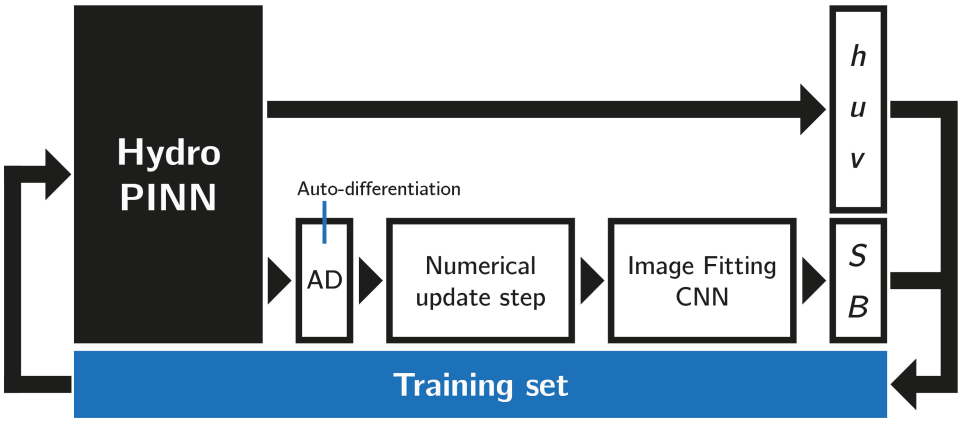


Figure 4.11: Training pipeline for hybrid training of the Hydro-PINN in conjunction with a numerical solver for the morphological quantities.

4.3. Physics-Informed Hydro-Morphodynamics

This section builds on the Hydro-PINN from Section 4.2 and extends it by including the sedimentary and vegetative dynamics that have been described in detail in Section 3.1.2. These morphological dynamics are described by two additional PDEs, and Section 4.3.1 explains how these can be incorporated in the learning process. Section 4.3.2 explains how the addition of these PDEs results in a multi-scale system of equations, both in spatial and temporal dimensions, and how this may complicate the training process. Section 4.3.4 continues by explaining how a multi-network setup can be used to alleviate these scale-differences. The resulting network is further referred to as the Saltmarsh-PINN.

4.3.1. Physics-Informed Sediment and Vegetation

As is covered in detail in Section 3.1.2, SFERE couples two morphological equations to the SWEs to describe the evolution of the sedimentary bed elevation and the vegetation density throughout the landscape. These PDEs describe a (simplified) balance between deposition and erosion of sediment, as well as slope-driven sediment transport, and the growth and decay of vegetation which is influenced by water depth and flow, as well as clonal growth.

These morphological equations form two learning objectives that this Saltmarsh-PINN should optimise for in addition to the existing hydrodynamical tasks. Like before, these equations directly translate to their corresponding loss terms:

$$L_S = \int_\Omega \int_t^{t+\mathrm{d}t} \left\| \frac{\partial S}{\partial t} - S_{in} \frac{h}{Q_s + h} + E_S \left(1 - p_E \frac{B}{k}\right) S \frac{\tau_b}{\rho} - \nabla(D_S \nabla S) \right\|^2$$

$$L_B = \int_\Omega \int_t^{t+\mathrm{d}t} \left\| \frac{\partial B}{\partial t} - rB\left(1 - \frac{B}{k}\right)\left(\frac{Q_q}{Q_q + h}\right) + E_B B \frac{\tau_b}{\rho} - \nabla(D_B \nabla B) \right\|^2$$

and they are considered two separate learning tasks for the network (i.e. gradient surgery [77] is applied to each term individually). In total, the network therefore has a total of six learning tasks: three hydrodynamical equations, two morphological equations and one boundary task.

At both the open and closed boundaries of the domain, there is assumed to be no gradient in sedimentary bed elevation or vegetation stem density. Like some boundary conditions in the previous sections, these zero-gradient conditions are directional constraints that require a separate loss term for horizontal and vertical boundaries. In Section 4.1.2 and later also in Section 4.2.3, it was decided to either merge the two directional constraints or to disregard them in the case of an open boundary, to minimise the number of learning objectives.

However, in the case of these two morphological quantities, combining these conditions can lead to unrealistic scenarios and opposing constraints, which can negatively impact the self organising triggers in the landscape. As is explained in Section 3.1.1, the scale dependent feedback is triggered by incisions in the bed topography at the open boundary. Water concentrates in these incisions and is evacuated from elevated areas between channels. By posing a zero-gradient condition in both directions at the open boundary, we directly restrict the network from creating these incisions, which can break the scale dependent feedback trigger.

It is therefore decided to work with separate boundaries on the horizontal and vertical boundaries of the domain and to accept that optimisation can become more complex after the increased number of learning objectives. This design choice is applied retroactively, meaning that all simplifications of directional boundary constraints in earlier sections are reversed. The resulting domain boundary is divided into four sections, where $\partial\Omega_x$ and $\partial\Omega_y$ represent the closed domain boundaries in vertical and horizontal direction, respectively, and $\partial\Omega_{\mathrm{open},x}$ and $\partial\Omega_{\mathrm{open},y}$ the open boundary.

At the closed boundaries, a zero-gradient condition is imposed on the water level height h, as well as on the sedimentary bed elevation S and the vegetation stem density B:

$$\begin{aligned} \frac{\partial h}{\partial x} &= 0 \quad \text{on } \partial\Omega_x, \text{ and} \\ \frac{\partial h}{\partial y} &= 0 \quad \text{on } \partial\Omega_y \end{aligned}$$

and idem for S and B. Additionally, at the closed boundary, the flow velocity is assumed to be zero in the direction perpendicular to the boundary:

$$\begin{aligned} u &= 0 \quad \text{on } \partial\Omega_x, \text{ and} \\ v &= 0 \quad \text{on } \partial\Omega_y. \end{aligned}$$

At the open boundary, similarly we impose a zero-gradient condition on the water level height h and vegetation stem density B. The sedimentary bed elevation is assumed to be zero, as deposition and erosion of sediment are assumed to be in balance at the open boundary:

$$S = 0 \quad \text{on } \partial\Omega_{\text{open},x} \text{ and } \partial\Omega_{\text{open},y}.$$

Lastly, water is free to flow in and out of the domain for which no boundary condition is posed on the flow velocity components u and v at the open boundary, in line with Section 4.2.3. Each of these boundary conditions forms a separate loss term, but to the network they are presented as one grouped learning objective $\mathcal{L}_{\text{bound}}$. This means that gradient surgery is applied on the boundary loss as a whole.

4.3.2. Multi-scale PDEs

With the addition of these two morphological equations, the system finally comprises all the landscape processes that are necessary to capture the multi-scale tidal channel formation like was done in SFERE [66]. Aside from a few minor changes to the equations in the system, all of which have been motivated throughout this chapter, this Saltmarsh-PINN works with the same system of equations that was used in their numerical simulation. In contrast to the Hydro-PINNs in earlier sections, the processes that are described by these five PDEs operate on vastly different temporal scales, as well as varying spatial scales. In this section, the scale-differences are briefly highlighted and we discuss how they can impact training of this Saltmarsh-PINN.

The hydrodynamics are most sensitive, in the sense that water reacts quickly to changes in the bed topology and is largely influenced by movement of the surrounding water, as well as by vegetation. These water dynamics therefore typically work on time-scales of seconds to minutes. Bio-morphological processes, on the other hand, operate on much larger timescales. Deposition and erosion of sediment are slow processes that typically require seasons to years to inflict any significant change to the landscape. Likewise, it can take years to centuries for vegetation to grow and cover a large enough portion of the landscape to start impacting the self-organising mechanisms. Section 4.3.3 explains how this Saltmarsh-PINN incorporates multiple *temporal* scales into the learning process.

To see how these processes operate on multiple spatial scales, firstly recall that the self-organisation of coastal landscapes is driven by a scale-dependent feedback that occurs between the three processes. In Section 3.1.1, it is described how vegetation imposes a stronger scale-dependent feedback compared to sediment alone, resulting in a more complex landscape structuring (i.e. more branching levels). Our PINN therefore has to recognise the impact of vegetation density in a certain region on the spatial scale-difference between the positive and negative feedbacks in the surrounding area. And it must do so for both the sediment and the vegetation. On the macro scale, the PINN furthermore has to recognise that the channelisation mechanism is a recursive process that generates branches on smaller and smaller levels as the channels develop.

Capturing these differences in scale poses an important challenge to the network and a single U-Net may have difficulty capturing all of these effects at once. A well known limitation of PINNs is that training can be extremely costly [27, 28] and it is likely that the cost of training increases significantly as we task the network with more non-linear and tightly coupled dynamical components. Some physics-informed methods partition the problem, either in space or in time, to lower the training cost [27, 28]. In the design of this Saltmarsh-PINN, we partition the problem by splitting the PDEs across multiple networks. This is somewhat similar to *operator-splitting* in numerical methods, where equations are solved individually and the final solution is created by combining the sub-solutions. SFERE [66] follows a similar approach, where the flow velocity equations are solved first before the remaining three equations. Section 4.3.4 explains in detail how the problem is partitioned across multiple networks and how this impacts learning.

4.3.3. Learning at Multiple Timescales

Most numerical hydro-morphological simulations rely on a *morphological acceleration factor* to bridge the timescale gap between the hydrodynamics and the bio-morphodynamics. The simulation time-step can often not be much larger than, for example, $\Delta t = 0.01$ seconds, as the hydrodynamics would otherwise become unstable. Without such acceleration factor, it is numerically not feasible to reach simulation times at the scale of the sedimentary and vegetative processes. SFERE [66] uses a morphological acceleration factor of $\Phi = 44712$ s $[\mathrm{T}_{\mathrm{M2}}]^{-1}$ and represents the number of seconds in one M2-tidal period T_{M2}. When applying an update step for either S or B, this factor is used as a multiplier to the time step Δt:

$$S(t + \Delta t) = S(t) + \frac{\partial S}{\partial t} \Delta t \Phi$$

and idem for B, essentially repeating the update step Φ times.

Mesh-less physics-informed networks do not work with such an update step. As such, it is not possible to accelerate the morphological processes by repeating the update step Φ times. Instead, if we wish for the PINN to accelerate the sedimentary and vegetative processes, this should be reflected in the respective loss terms by indicating that the rate of change should correspond to the amount of change across a longer time-span. This can be done by dividing the auto-differentiated $\partial S/\partial t$ by the morphological acceleration factor Φ before using it to compute the sedimentary loss term L_S, and similar for vegetation stem density B. Automatically, the network learns to produce accelerated solutions for those variables.

4.3.4. Multi Network Training

As was explained in Section 4.3.2, the processes described in this system of equations operate on vastly different temporal and spatial scales and a single U-Net may not be able to efficiently capture these multi-scale dynamics. For this reason, underlying this Saltmarsh-PINN are three separate U-Nets for the hydrodynamics, sedimentary and vegetative dynamics, respectively. All networks receive the same input, with the complete landscape state captured by the spline weights for all variables h, u, v, S and B. The hydrodynamical network is identical to the Hydro-PINN described in Section 4.2, with the exception of the updated boundary conditions that are described in Section 4.3.1. The other two networks output the spline coefficients corresponding to the sedimentary bed elevation S and vegetation stem density B, respectively. Variable scaling is applied across all networks, in line with Section 4.2.5, and all networks make use of the same U-Net architecture. The complete network architecture is illustrated in Figure 4.12.

As a result of using three separate neural networks, the different learning objectives are suddenly impacted by the performance of each of them separately. Specifically, the hydrodynamical network is responsible for the PDE objectives for the continuity and velocity equations L_h, L_u and L_v, and the sedimentation and vegetation networks for L_S and L_B, respectively. For the boundary loss term L_{bound}, the networks carry a shared responsibility, however it would theoretically be possible to split this term into separate objectives that can be learned by the individual networks.

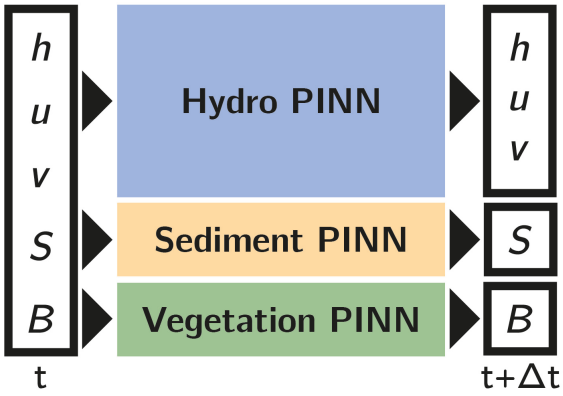


Figure 4.12: Three separate PINNs are used to solve the hydrodynamical, sedimentary and vegetative components of the multi-scale hydro-morphodynamical system.

Important to consider is that this Saltmarsh-PINN is still solving one system of equations. Simply because the learning has been spread over three networks does not mean that it suddenly solves three separate problems. It remains a highly coupled system, meaning that the solution to one equation can significantly impact the outcome of another. Strictly speaking, this also means that the performance of one network in this PINN can directly impact the performance of another. Balancing the learning momentum for each of the objectives is therefore just as important as before and, for this reason, it is decided not to split the different learning objectives among the different responsible networks, but instead to treat the three U-Nets as a single network during training. To achieve this, the parameters of all three networks are grouped and placed under a single `Adam` optimiser that can be wrapped in a `PCGrad` object for gradient surgery [77]. This allows for a direct balancing of all six learning objectives through gradient surgery and is compatible with objectives that the networks are jointly responsible for, like the boundary loss.

4.3.5. Training Procedure

Besides facilitating learning at multiple spatial and temporal scales, having a separate network for each process enables a much more flexible training procedure. For example, it allows for multi-stage learning where different networks are trained in different stages to give one of the U-Nets a head-start. It also allows for asymmetric training so that one of the networks completes one training cycle in the time that another needs to complete four or five. The added flexibility also makes the setup much more modular, so that each network could be swapped out and replaced with one that has a completely different architecture, for example. The hybrid setup discussed in Section 4.2.9 forms an example where this flexibility can be useful.

The training procedure for this Saltmarsh-PINN is mostly identical to that of the Hydro-PINN, where the numerical morphological solver is essentially replaced with two PINNs to solve each of the morphodynamical equations. The three networks are trained simultaneously and at the same rate, i.e. we do not make use of asymmetric or multi-phase training. Like for the Hydro-PINN, 500 landscape snapshots from SFERE are used as starting conditions and the landscapes are recycled into the training pool after being processed by the network.

5

Implementation Details

The design and implementation of the setups and experiments described in Chapter 4 are largely inspired by the models behind SFERE [66] and Hermite Spline-PINNs [73]. This chapter explains how both models were incorporated into our model implementations, and what adaptations were necessary to run our experiments. First, Section 5.1 outlines a custom re-implementation of SFERE using PyTorch [48] for GPU acceleration. This was considered a useful or even necessary step, as almost all code for our work would be written using PyTorch. Next, Section 5.2 explains how Spline-PINN [73] forms the basis for the implementation of our models, and what adaptations were needed for increased scalability and flexibility. Finally, Section 5.3 explains how embedding of the Image Fitting CNN from Section 4.2.7 in the training pipeline can facilitate numerical solving of a part of the system while still making use of the auto-differentiation capabilities of Hermite Splines.

5.1. Re-implementation of SFERE in PyTorch

Throughout this work, SFERE [66] is used as a reference solution to the hydro-morphodynamical simulations, as well as to generate training landscapes for our PINNs. The mechanics behind SFERE have been detailed in Section 3.1. Originally, this numerical solver was implemented in Python with OpenCL for GPU acceleration [69]. The programme is publicly available as a Jupyter Notebook on GitHub [67]. Similar to SFERE, all machine learning models developed in this thesis are implemented in Python, but with PyTorch instead of OpenCL. PyTorch is a widely used platform for machine learning, and Spline-PINN [73] makes extensive use of its frameworks and libraries to train and evaluate their physics-informed models. Using multiple different frameworks for GPU acceleration is generally not recommended and is also not practical for reasons of data formatting and management. As such, we re-implemented SFERE using PyTorch as part of this thesis. Our re-implementation is publicly available online [16], alongside any necessary documentation. The aim of this section is not to provide a complete overview of our re-implementation, but to highlight some of the most important mechanisms behind our simulation.

PyTorch makes use of highly optimised pre-compiled GPU operations that can be directly imported into Python, eliminating the need for custom GPU shaders and providing efficient memory management and compatibility with numerous GPU architectures without changes to the implementation. All operations are available as classes and functions in Python, making them directly compatible with the workflow of the programme. This can be largely beneficial for readability and maintainability, as well as for reproducibility.

The power of our PyTorch implementation comes from the use of convolutions to compute first- and second-order gradients of any quantity using fast, optimised GPU kernels. By specifying the operators $\partial/\partial x$, $\partial/\partial y$, $\partial^2/\partial x^2$ and $\partial^2/\partial y^2$ as convolutional filters, the operators can be implemented directly as a Python function, as in Listing 5.1.

Listing 5.1: Gradient operators are implemented as Python functions by using 3x1 convolutional filters in x- and y-orientations.

```
import torch
import torch.nn.functional as F

dx = 1
dy = 1

# Convolution filters
dx_kernel = torch.tensor([-0.5,0,0.5], device=torch_device).view(1, 1, 1, 3)
dy_kernel = torch.tensor([-0.5,0,0.5], device=torch_device).view(1, 1, 3, 1)
dx2_kernel = torch.tensor([1.0, -2.0, 1.0], device=torch_device).view(1, 1, 1, 3)
dy2_kernel = torch.tensor([1.0, -2.0, 1.0], device=torch_device).view(1, 1, 3, 1)

# Gradient operators as a Python function
def d_dx(quantity):
    return F.conv2d(quantity, dx_kernel, padding=(0,1)) / dx

def d_dy(quantity):
    return F.conv2d(quantity, dy_kernel, padding=(1,0)) / dy

def d2_dx2(quantity):
    return F.conv2d(quantity, dx2_kernel, padding=(0,1)) / dx**2

def d2_dy2(quantity):
    return F.conv2d(quantity, dy2_kernel, padding=(1,0)) / dy**2
```

In PyTorch, data is usually stored as *tensors*. For example, an image could be stored as a 2D tensor, where pixel values can be indexed by the row and column where it resides. Tensors can have any dimensionality, but it is conventional to work with 4D tensors where the first dimension corresponds to the batch size, the second to the number of (colour) channels in the image, and the two remaining channels for the x- and y-dimensions. This pattern can be upscaled to 3D data, by adding a z-dimension. For our SFERE implementation, this convention is followed and we create tensors for the water layer thickness h, flow components u and v, bed topography S and vegetation density B, as in Listing 5.2. In that listing, it can also be seen how we select an acceleration device by specifying the correct GPU architecture, or alternatively by performing all operations on the CPU.

Listing 5.2: Selecting an acceleration device and allocating memory for the salt-marsh simulation.

```
import torch
class SaltmarshDomain:
    def __init__(self):

        # Domain size
        self.width = 800
        self.height = 800

        # Select acceleration device
        self.device = torch.device("cuda")
        # self.device = torch.device("mps") # For Metal (macOS)
        # self.device = torch.device("cpu") # CPU

        # Allocate memory for hydro-morphodynamical quantities and move to GPU
        # Tensor dimensions: Batch-size (1); Image channels (1, scalar values); 2D spatial
            domain (WxH).
        # Initialise all values to zero
        self.h = torch.zeros(1, 1, self.width, self.height).to(self.device)
        self.u = torch.zeros(1, 1, self.width, self.height).to(self.device)
        self.v = torch.zeros(1, 1, self.width, self.height).to(self.device)
        self.s = torch.zeros(1, 1, self.width, self.height).to(self.device)
        self.b = torch.zeros(1, 1, self.width, self.height).to(self.device)
```

Any operations that are applied to these tensors will be performed on the device at which they reside. PyTorch offers support for most tensor operations, out of the box. Basic operations like addition, subtraction, pointwise multiplication and division of a tensor with a scalar, or with another tensor, follow the standard Python syntax. For other operations, the library provides functions, such as `torch.pow` to raise all values in the tensor to a certain power.

In preparation to the simulation, the instructions by [66] are followed and the domain is initialised with a uniform water layer of thickness H_0. In addition, flow velocities and sedimentary bed elevation are set to zero. Then, a domain-sized tensor is created with random values between $[0, 1]$. In those cells where the random value falls below p_{est}, vegetation stem density is set to carrying capacity k and zero everywhere else. This process is illustrated in Listing 5.3.

Listing 5.3: Initialisation of the simulation

```
class SaltmarshDomain:
    def __init__(self):
        [...]

        # Simulation parameters
        self.H0 = 0.02
        self.pest = 0.002
        self.k = 1500
        [... other parameters]

        # Initial conditions
        self.h[:, :, :, :] = self.H0 # Broadcast H0 across tensor h
        self.b[torch.where(torch.rand_like(self.b) <= self.pest)] = self.k
```

We will illustrate the simulation pipeline by outlining the order of operations in Listing 5.4. Each iteration starts by creating copies of the h, u, v, S and B tensors. Next, we apply the thin-film wetting drying algorithm as described by [66] and do any pre-calculations that are necessary for the bed friction terms, for example. The main body of the simulation consists of computing the PDEs (in Listing 5.4 illustrated for u and v) by making use of the differentiation operators described earlier. Finally, we perform the update step and apply the boundary conditions by making use of smart indexing in PyTorch.

Listing 5.4: General simulation workflow

```
class SaltmarshDomain:
    def simulate(self):

        # Copy the current state
        h, u, v, s, b = self.h, self.u, self.v, self.s, self.b

        # 1. Apply wetting-drying by clamping h to a minimum of self.Hc
        h = torch.clamp(h, min=self.Hc)

        # 2. Update flow velocities
        [... pre-calculations]

        # 2.1 Compute PDEs
        du_dt = -self.grav * self.d_dx(h + s) - u * self.d_dx(u) - v * self.d_dy(u) -
            tau_bx_per_rho/h + self.DU * (self.d2_dx2(u) + self.d2_dy2(u))
        dv_dt = -self.grav * self.d_dy(h + s) - u * self.d_dx(v) - v * self.d_dy(v) -
            tau_by_per_rho/h + self.DU * (self.d2_dx2(v) + self.d2_dy2(v))

        # 2.2 Update step
        u = u + du_dt * dt
        v = v + dv_dt * dt

        # 2.3 Boundary conditions
        # Closed boundary at the top
        u[:,:,:,0] = -u[:,:,:,1]
        v[:,:,:,0] =  v[:,:,:,1]

        [... other closed boundaries]

        # Open boundary at the right
        u[:,:,-1,:] = 2*u[:,:,-2,:] - u[:,:,-3,:]
        v[:,:,-1,:] = v[:,:,-2,:]

        [... similar for h, S and B]

        # Copy back the state
        self.h, self.u, self.v, self.s, self.b = h, u, v, s, b
```

5.2. Towards a Modular and Scalable Hermite Spline-PINN

The original code base of Spline-PINN [73] features an efficient implementation for the use of Hermite Splines in a physics-informed U-Net [71] for up to two physical quantities. Several counters and size records are required to keep track of the size and location of the spline latent space for both variables. This is doable for two quantities, but can lead to unreadable and unreliable code when scaled to higher dimensionalities. For the more complex PDE systems considered in this thesis, a more scalable and flexible implementation is therefore needed and Section 5.2.1 explains how this was achieved as part of this thesis. As the complexity and size of the code scaled with the increased complexity of this new problem, important parts of the code were re-organised and this is explained in Section 5.2.2. Finally, Section 5.2.3 explains how multiple Spline-PINNs could be simultaneously trained, while making use of gradient surgery for multiple objectives.

5.2.1. Increased Scalability and Readability

As is explained in detail in Section 3.2.1, the physical quantities are represented by Hermite Spline fields in our physics-informed models. The function value and first n derivatives are stored at support points, that are laid out on a regular grid. Depending on the chosen spline order n, this results in a $W \times H \times D$ tensor for each quantity, where W and H are the width and height of the grid and $D = (n+1)^2$ the number of spline weights stored at each support point. With only two quantities in the original implementation, Spline-PINN could easily keep track of the dimensionality of this data without the need for classes or containers. With more quantities, however, and with our more complex setting, separating these data management responsibilities using classes in Python makes the code significantly more flexible and readable, and therefore more reliable.

In memory, the spline weights are stored collectively in one tensor of size $W \times H \times K$, where

$$K = \sum_i D_i = \sum_i (n_i + 1)^2$$

is the sum of the dimensionalities of all variables (physical quantities) i in the system. They are stored collectively for efficient storage and processing by the PINN. This tensor is also referred to as the *hidden state* or the *spline latent space*. Pointers are needed to remember what parts of the latent space belong to which physical quantity and it is the responsibility of the `Dataset` class to keep track of these. Spline-PINN [73] uses manually tracked pointers and manipulates the spline latent space by indexing the hidden state directly. In our implementation, we opt for a more scalable approach that does not require such manual memory management.

To do this, two classes are implemented to manage the data inside the hidden state tensor. Firstly, the `SplineVariable` class represents a physical quantity and provides functionality for the interpretation and manipulation of the Hermite Spline data that is stored inside the hidden state. Secondly, the `SplineArray` class gathers all `SplineVariable` instances in the programme and manages the order in which their data is stored and retrieved inside the hidden state tensor. Furthermore, it allows spline variables to be identified by their name or mathematical symbol.

Listing 5.5 illustrates how these classes can be used to create an array of five Hermite Spline quantities, each associated with a name, and how this array can be used to create a collective hidden state tensor of the appropriate size. Furthermore, it illustrates how spline variables can be indexed by name and how its corresponding portion of the spline latent space can be retrieved. Initialisations, manipulations and sampling of the solution are more flexible and readable operations by making use of these functionalities and this allows reliable up-scaling towards more complex PDE systems. Adding and removing quantities to and from the dataset can be done without the need to rewrite other parts of the code, because we use names for indexing.

Our spline variable class makes use of the same highly optimised Hermite Spline sampling routines as in Spline-PINN [73, 71], featuring efficient PyTorch operations for differentiation, GPU based sampling and caching of convolutional filters.

Listing 5.5: Creating an array of spline variables and using object-oriented constructs to interpret and manipulate the hidden state tensor.

```
# Initialisation of spline variables
# Each SplineVariable is associated with a name, its spline order, and
# whether derivatives and laplacians should be included during sampling.
self.variables = SplineArray(
    SplineVariable("h", 1, requires_derivative=True),
    SplineVariable("u", 1, requires_derivative=True, requires_laplacian=True),
    SplineVariable("v", 1, requires_derivative=True, requires_laplacian=True),
    SplineVariable("s", 1, requires_derivative=True, requires_laplacian=True),
    SplineVariable("b", 1, requires_derivative=True, requires_laplacian=True),
    device=self.device
)

# Allocate memory for the hidden state
dataset_size = 1000
W, H = 200
hidden_state = torch.zeros(dataset_size, self.variables.hidden_size(), W, H)

# Retrieve the SplineVariable instance belonging to the physical quantity "h"
variable_h = self.variables["h"]

# Extract the support point weights belonging to "h" from the given hidden state
weights_h = self.variables.extract_from(hidden_state, "h")

# Use SplineVariable functionality for continuous sampling of the h-field
offset = [offets are different for training and evaluation]
h, grad_h, laplacian_h = variable_h.interpolate_at(weights_h, offset)

# Direct manipulation of the spline latent space
# 'get_singular_slice_for' returns the slice of the first order spline weights belonging to
    variable 'h'
hidden_state[:, self.variables.get_singular_slice_for("h")] = self.H0
```

5.2.2. Code Organisation

This section aims to provide a brief outline of the adaptations that we made to the code organisation inspired by Spline-PINN [73, 71] to facilitate the more complex and lengthy routines needed for our problem setting. For that, we explain the design philosophy behind the code base organisation without going into detail regarding specific files or routines. Our code is publicly available online [15], alongside any necessary documentation.

The use of Python classes and containers, e.g. for clear and efficient memory management, has been more extensively applied in other parts of our code base as well. For example, every PyTorch tensor that appears in our implementation is *owned* by an instance of a class. And for each class that stores tensors, upon initialisation we specify on what device (i.e. GPU or CPU) its data should reside. This ensures that it is always clear what device is used for certain operations and where the corresponding memory is stored. By wrapping tensors inside a class, we can immediately specify how and when its data should be moved to the GPU. This keeps the necessary tensor-specific routines close to the data, so that any data management operations can be removed from the main logic related to training or evaluation of the models. This increases the readability of the code, and reduces the risk of mistakes.

To name an example where this is relevant: In Spline-PINN [73, 71], the code related to the training routine is responsible for moving the necessary data between the GPU and the CPU, whereas in our code this is handled by the dataset. After all, the dataset is the owner of the data. Upon retrieving a batch from the dataset, the corresponding data is automatically moved to the GPU and once a training or evaluation iteration has been completed, the resulting data is moved back to the CPU upon passing them back to the dataset.

Compared to Spline-PINN, our code base contains more and longer routines as the complexity of the underlying physical problems is larger. These adaptations have helped to keep the code understandable at a glance and to reduce the risk of mistakes. Every routine is implemented as a method inside a class, and every class contains a `self.device` attribute. Among other things, this makes it significantly easier to trace issues related to memory management, such as leaks or inefficient operations.

5.2.3. Multi-Network and Multi-Phase training with Gradient Surgery

When setting up the network architecture for the Saltmarsh-PINN in Section 4.3, it is explained how an ensemble of three PINNs is collectively responsible for the optimisation of multiple objectives. In Section 4.3.4, it is explained how we would still like to apply gradient surgery [77] to each of the PDE and boundary loss terms and that this poses a shared responsibility for the three networks to find an optimal solution with respect some of the terms. This section aims to briefly explain how this shared learning responsibility is realised in the training routine of our PINN, while also making use of gradient surgery. We also explain how our implementation facilitates multi-phase or asynchronous training of the three networks.

In this section, we stick to the example of the Saltmarsh-PINN from Section 4.3, where we make use of three separate PINNs. In code, these return as the `water_net`, the `sediment_net` and the `vegetation_net`. When the networks are trained simultaneously, their parameters can be gathered into a single `Adam` optimiser as in Listing 5.6. This optimiser can then be wrapped inside a `PCGrad` object for gradient surgery [77].

Listing 5.6: Multi-network learning for the optimisation of shared objectives

```
# Gather the parameters of all three networks into a single Adam optimiser
# and optionally provide separate learning-rates.
self.optimizer = Adam([
    {"params": self.water_net.parameters(), "lr": self.params.lr},
    {"params": self.sediment_net.parameters(), "lr": self.params.lr},
    {"params": self.vegetation_net.parameters(), "lr": self.params.lr},
])

# Wrap the optimiser into a PCGrad object for gradient surgery
self.optimizer = PCGrad(self.optimizer)
```

In the training routine, we can control which networks in the ensemble are trained by targeting them with the loss terms that we provide to the gradient surgery operator (PCGrad). For example, in Listing 5.7 we initially only provide the terms `loss_h`, `loss_u`, `loss_v` and `loss_bound` to PCGrad, to target only the `water_net` PINN. When we would also like to train the `sediment_net` or `vegetation_net`, we can make use of boolean switches `self.training_sediment` and `self.training_vegetation` to append the respective loss terms.

Listing 5.7: In- or excluding loss terms to target the learning of specific networks in the ensemble

```
# For backprop using PCGrad, construct each loss term
pcgrad_losses = [
    loss_h,
    loss_u,
    loss_v,
    loss_bound
]

# In the sediment training stage, add sediment to PCGrad as well
if self.training_sediment:
    pcgrad_losses.append(loss_s)

if self.training_vegetation:
    pcgrad_losses.append(loss_b)

# Reset old gradients to 0 and compute new gradients with backpropagation
self.water_net.zero_grad()
self.sediment_net.zero_grad()
self.vegetation_net.zero_grad()

# PCGrad backprop pass
self.optimizer.pc_backward(pcgrad_losses)
```

Since the boundary loss term `loss_bound` is composed of all boundary objectives, the same switches should be applied whenever `loss_bound` is computed. In Listing 5.8, an example is included to demonstrate how we can target the open boundary loss term to each of the three networks.

Listing 5.8: Using switches to in- or exclude boundary objectives

```
# Open boundary loss
loss_bound_open = torch.mean(sample_opened_mask[:,:,1:-1,1:-1] * self.loss_function(
    grad_h
), dim)

if self.training_sediment:
    loss_bound_open = loss_bound_open + torch.mean(sample_opened_mask[:,:,1:-1,1:-1] * self.
        loss_function(
        s.repeat(1, 2, 1, 1) # S = 0
    ), dim)

if self.training_vegetation:
    loss_bound_open = loss_bound_open + torch.mean(sample_opened_mask[:,:,1:-1,1:-1] * self.
        loss_function(
        grad_b / self.params.k  # Normalize any loss related to b by max. carrying capacity
            to match scale-difference
    ), dim)
```

By controlling when the switches are turned on or off, we can create a multi-phase or asynchronous training progression. It is important to realise that this way of implementing conditional training of a group of networks can create shocks during training by introducing sudden and unpredictable changes in the loss landscape. This can negatively affect the training effectiveness of the network ensemble. The loss landscapes for each network are still somewhat coupled via the gradient surgery object, so that a sudden introduction of a new loss term targeted to one network can still shock the learning curve of another. This could be prevented by using separate optimisers for the three networks, however in our implementation of the Saltmarsh-PINN this separation was deliberately avoided, as is explained in Section 4.3.4. Shocks in the learning curve are also inherent in multi-phase or asynchronous learning, since the solutions produced by the three networks are coupled by terms in the PDEs. Completely avoiding them is therefore impossible and this should be accounted for whenever an asynchronous training strategy is applied.

5.3. Image Fitting CNN, Embedded in Training Pipeline

With the Image Fitting CNN, explained in detail in Section 4.2.7, it becomes possible to perform a numerical update step on the Hermite Spline fields that are stored inside the hidden states of the PINN. By sampling the desired physical quantities on a regular grid of some chosen resolution, and by employing auto differentiation to obtain gradients in the field, the corresponding PDEs can be directly computed and time-integrated with a numerical time-step. The Image Fitting CNN can then be used to transform the resulting regular image back to the Hermite Spline hidden state. This process is illustrated in Listing 5.9.

Listing 5.9: Image Fitting CNN allows partial numerical solving

```
# Sample the continuous spline coefficients to obtain regular images for each of the fields
# and apply automatic differentiation to obtain the corresponding gradients
#
# Use resolution factor 4 to obtain an 800x800 image
h, grad_h, laplacian_h, u, grad_u, laplacian_u, v, grad_v, laplacian_v, s, grad_s,
    laplacian_s, b, grad_b, laplacian_b = self.dataset.interpolate_superres(old_hidden_state,
     4)

# Because these quantities were obtained from Hermite Splines, we can apply wetting-drying
    similar to the PINN
he = F.softplus(h)

# [Compute any intermediate quantities]

# Compute a numerical update-step
ds_dt = self.params.Sin * (he / (self.params.Qs + he)) - self.params.Es * (1.0 - self.params.
    pE * (b / self.params.k)) * s * tau_b_per_rho + div_Ds_grad_S
db_dt = self.params.r * b * (1.0 - (b/self.params.k)) * (self.params.Qq / (self.params.Qq +
    he)) - self.params.EB * b * tau_b_per_rho + self.params.DB * laplacian_b

s = s + ds_dt * (self.params.dt * self.params.morphological_acc_factor)
b = b + db_dt * (self.params.dt * self.params.morphological_acc_factor)

# [Apply boundary conditions]

# Use the Image Fitting CNN to find the Hermite Spline latent space corresponding to this
    numerical solution
new_hidden_state = imfit_net(torch.cat([h, u, v, s, b], dim=1))
```

6

Evaluation

The overarching objective of this thesis is to embed a complex multi-scale hydro-morphodynamical system for the formation of tidal channels into a physics-informed learning framework. Throughout this work, three physics-informed models were setup for variations of the problem with increasing levels of complexity. This chapter presents the predictions made by each model in different configurations and discusses them from an inwards-looking perspective, i.e. with respect to the research objectives specified in Chapter 1 and Chapter 2. We start by assessing the performance and accuracy of the physics-informed hydrodynamics in a closed water basin in Section 6.1. Next, the accuracy of the Image Fitting CNN is discussed in Section 6.2. Then, we shift to the more complex setting of a partially open coastal wetland and start by focussing on the hydrodynamical solutions in Section 6.3. Finally, we consider the full hydro-geomorphological system in Section 6.4. In a separate discussion chapter, Chapter 7, all these findings will be placed into the broader perspective of the surrounding research fields by discussing them from an outwards-looking angle.

6.1. Physics Informed Hydrodynamics in Closed Water Basins

This section focuses on the performance of the SWE-PINN for hydrodynamics in closed water basins. The precise setup for this model is described in Section 4.1. The goal of the design and evaluation of this model is to verify that the proposed physics informed method produces sensible solutions to this simpler hydrological problem and to gain understanding about the behaviour of physics informed models when tasked with variations of the problem. The objective is not to outperform any existing solutions, but rather to demonstrate that a PINN is capable of finding solutions by solely training on the underlying coupled PDEs. To that end, we look for visual characteristics of the solutions to assess their accuracy. Additionally, to test its generalisation capabilities, the model is presented with more complex domain shapes, varying placement of obstacles, as well as oscillators with varying amplitudes and frequencies. Furthermore, the effectiveness of gradient surgery for balanced multi-objective learning is tested by training both with and without PCGrad.

The training loss is indicated for each loss term L_h, L_u, L_v and L_{bound} separately in Figures 6.1 and 6.2 for training *with* and *without* PCGrad, respectively. From these figures, it is clear that training with gradient surgery leads to faster convergence. Where Figure 6.1 already shows convergence for all terms after roughly one third of the training time, for Figure 6.2 one might argue that terms L_u and L_v have not yet reached convergence after the full training duration. However, disabling gradient surgery seems to allow further optimisation of L_u and L_v, as they reach lower loss values in Figure 6.2 compared to Figure 6.1. The boundary loss L_{bound} on the other hand seems to slightly worsen over time without gradient surgery.

The reason for reaching better loss values for L_u and L_v when PCGrad is disabled is likely because of optimisation conflicts between these terms and the boundary loss L_{bound}. After some time, this would mean that further optimisation of L_u and L_v is only possible at the cost of higher boundary losses and this is precisely what is indicated by Figure 6.2. The network prioritises the learning of L_u and L_v over

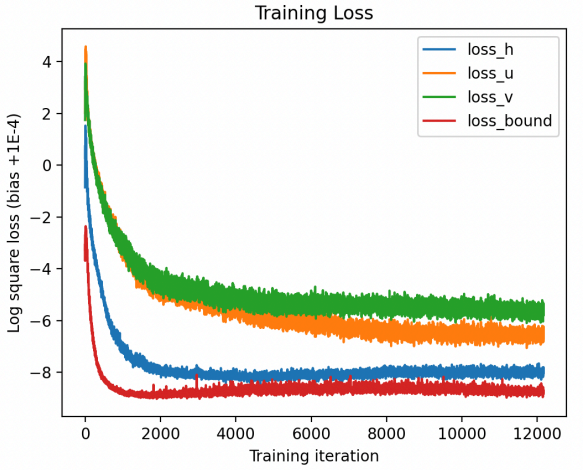


Figure 6.1: Training loss over the course of training the SWE-PINN *with* PCGrad gradient surgery for three days.

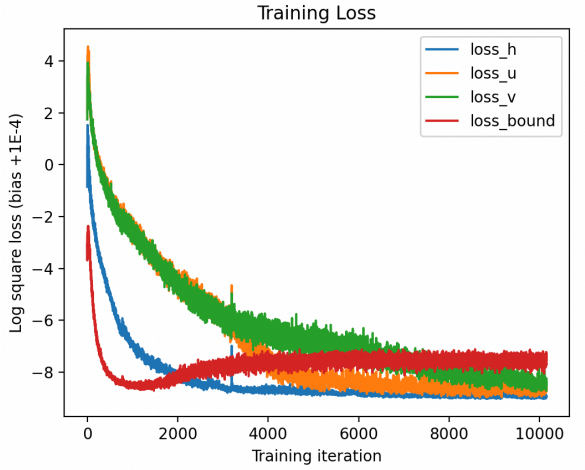


Figure 6.2: Training loss over the course of training the SWE-PINN *without* PCGrad gradient surgery for three days.

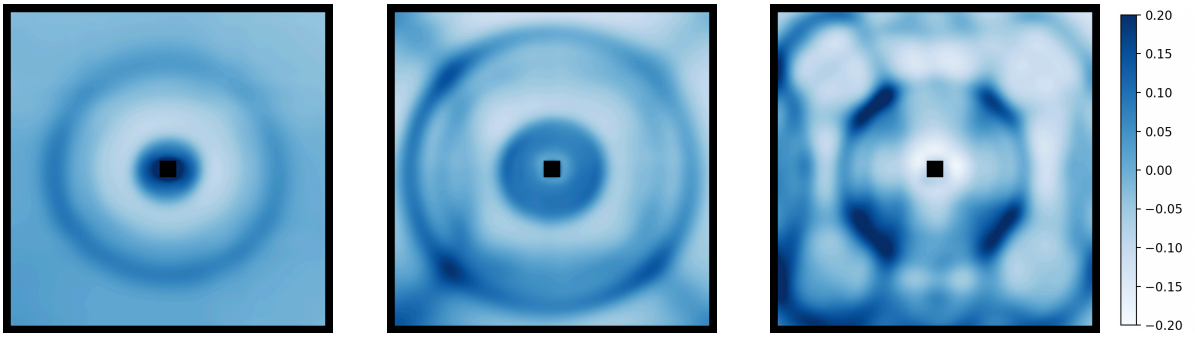
that of L_{bound}, after which the former two terms continue to decrease and the latter starts to increase. PCGrad is designed to circumvent these conflicts by removing opposing gradients between the terms and shows to be effective in Figure 6.1.

Both PCGrad and non-PCGrad versions of this SWE-PINN can simulate waves starting from the centre and travelling outwards until they reach one of the edges of the domain, as can be seen in Figure 6.3. With gradient surgery, the PINN is noticeably better at predicting reflections on the outside boundaries in Figure 6.3b compared to Figure 6.3e. Consequently, the PCGrad SWE-PINN accurately predicts more complex interference patterns between parts of the waves that are travelling towards the boundary and parts that have already reflected in Figure 6.3c. Without gradient surgery, the solution shows almost no signs of interference in the domain, as can be seen in Figure 6.3f, revealing the absence of any reflected waves.

Waves can be seen reflecting upon encountering a vertical barrier inside the domain in Figure 6.4a. Moreover, a part of the wave can be seen bending around the tip of the barrier. However, when gradient surgery is disabled, the network seems to disregard the barrier completely and the waves simply permeate the obstacle as in Figure 6.4b. Multiple oscillators can be randomly placed to produce waves that immediately begin to interfere with each other, as in Figure 6.5. Both with and without PCGrad, the SWE-PINN shows its ability to model multiple oscillators effectively and the interference patterns look similar between the two versions.

Across all aforementioned figures, the buoy oscillates with an amplitude of $A = 0.5$. The resulting waves, however, have a much smaller amplitude with the PCGrad SWE-PINN producing waves of approximately half the size and the non-PCGrad PINN with amplitudes no larger than 10% of the original. What further stands out from Figure 6.3, is that the PCGrad SWE-PINN produces waves with a notably sharper and more circular wave-front than the non-PCGrad variant.

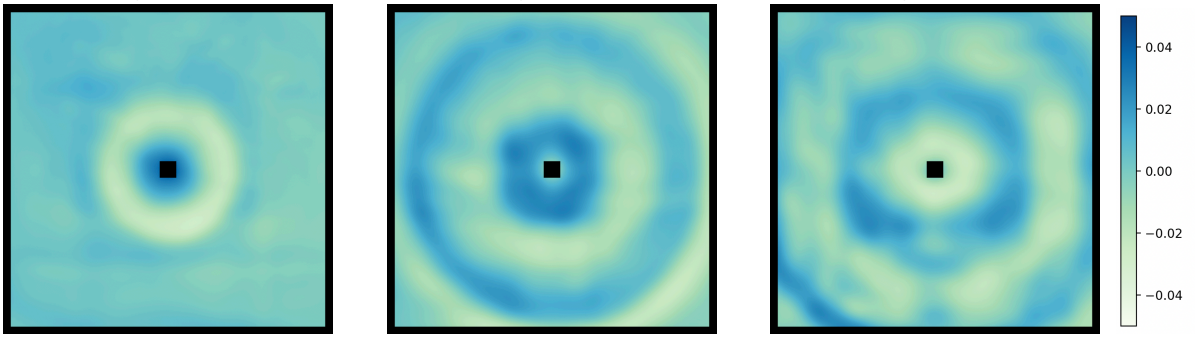
The first generalisation cases can be created by altering the oscillation amplitude and frequency of the buoys. After lowering the amplitude to $A = 0.1$ in Figure 6.6, the PCGrad SWE-PINN accurately predicts waves that are notably less tall and that interact with the outside boundaries and other waves in much the same way as before, in Figure 6.6a. In general, however, the solution seems to be less sharply defined and some water can be seen to gather in the lower left corner of the domain. As can be seen in Figure 6.6b, the non-PCGrad variant seems unable to work with this smaller amplitude, as the patterns are clearly different from those in Figure 6.6a and do not resemble travelling waves or an interference pattern that can result from this basin configuration. Shorter and longer wave-lengths can be created by changing the frequency of the oscillations, as is illustrated in Figure 6.8. Without being trained on these different frequencies, the PCGrad SWE-PINN is still capable of solving these hydrodynamics and producing waves with an appropriate wave-length, given the updated frequency.



(a) Water basin state at $t = 25$s after training with PCGrad

(b) Water basin state at $t = 50$s after training with PCGrad

(c) Water basin state at $t = 100$s after training with PCGrad

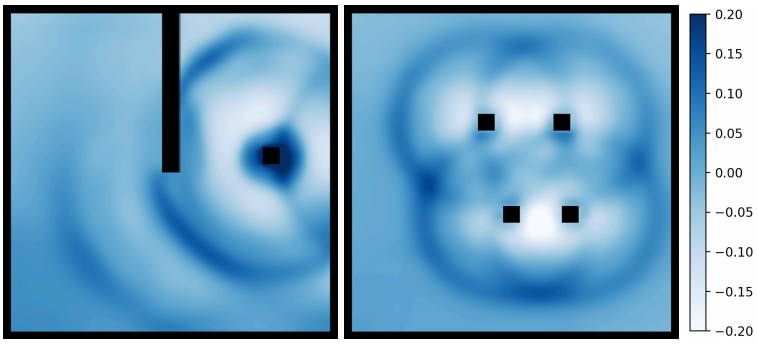


(d) Water basin state at $t = 25$s after training without PCGrad

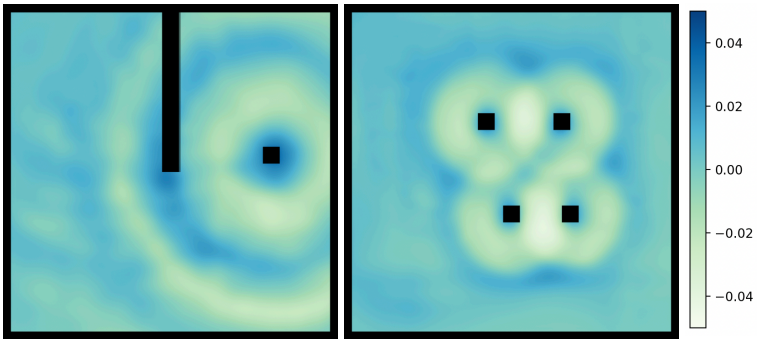
(e) Water basin state at $t = 50$s after training without PCGrad

(f) Water basin state at $t = 100$s after training without PCGrad

Figure 6.3: State of a water basin with an oscillator in the centre, after 25, 50 and 100 seconds into the simulation. *Note that the non-PCGrad network produces waves with a much smaller amplitude.*



(a) PCGrad enabled



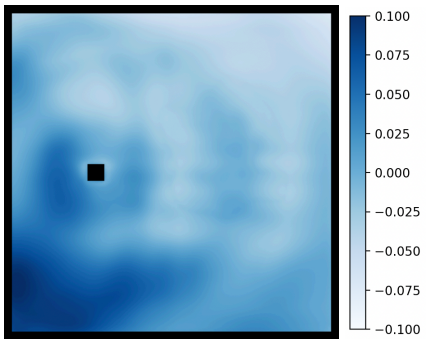
(b) PCGrad disabled

Figure 6.4: Waves reflecting on a vertical barrier, positioned in the top centre of the domain ($t = 45$s)

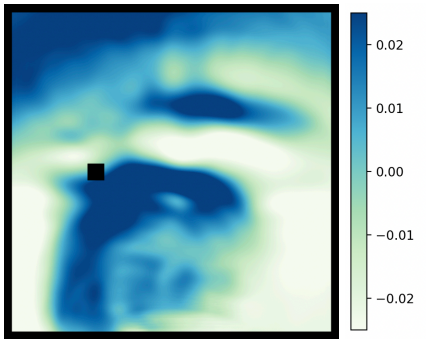
(a) PCGrad enabled

(b) PCGrad disabled

Figure 6.5: Multiple randomly placed oscillators simultaneously create waves in the domain ($t = 20$s)

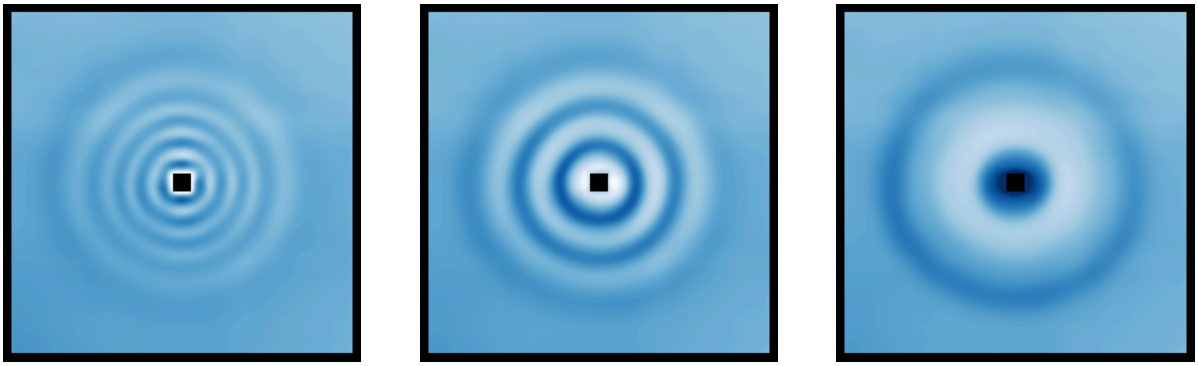


(a) PCGrad enabled



(b) PCGrad disabled

Figure 6.6: Buoy oscillating with an amplitude of $A = 0.1$ ($t = 50s$)



(a) Frequency $f = 1/5$ s^{-1} **(b)** Frequency $f = 1/8$ s^{-1} **(c)** Frequency $f = 1/20$ s^{-1}

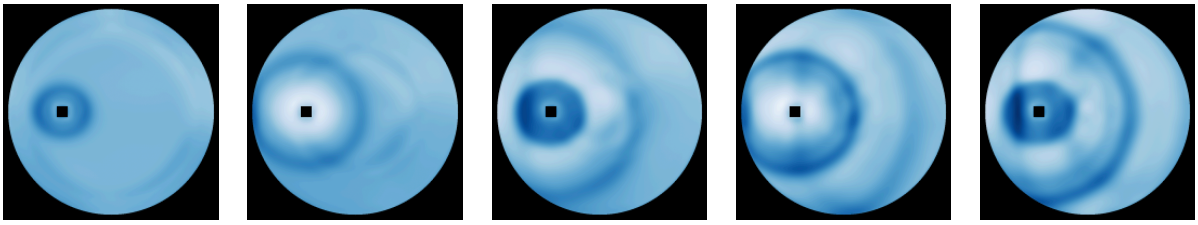
Figure 6.8: Buoys oscillating at different frequencies. PCGrad was enabled during training and figures are taken at $t = 25$s

The network can also generalise towards different and more complex domain shapes. For example, the PCGrad SWE-PINN can find adequate solutions for water inside a circular domain in Figure 6.9, despite not having seen this boundary shape during training. After bouncing off the circular boundary on the left of the domain, the reflected wave morphs into a wave-front that is almost perfectly vertical. Obstacles inside the domain can be of virtually any shape and the SWE-PINN can find an adequate solution without re-training. Two examples are presented in Figures 6.10 and 6.11. In the first figure, we can see that the waves form an adequate interference pattern around the oscillator, as they reflect on the domain edges, as well as on both the obstacles. As the waves travel around or between the obstacles, they fan out as they bend around the corners. The second figure shows a double-slit experiment, where the domain is divided into two compartments, separated by a vertical wall with two narrow slits. As the waves created by the oscillator in the left compartment reach these slits, they start acting like oscillators of their own and create wave patterns in the right compartment. The amplitude also drastically lowers as the wave passes through the slits.

From these results, it is clear that the use of gradient surgery has a significant and positive impact on the training effectiveness of this SWE-PINN. Not only does it accelerate convergence during training, but it effectively avoids conflicts among the different learning tasks, leading to even optimisation of all terms in Figure 6.1. Furthermore, training with gradient surgery leads to qualitatively significantly better solutions. The PCGrad SWE-PINN shows accurate simulation of waves, reflections and interference patterns and generalises towards complex domain shapes, including obstacles, and varying oscillation parameters. In contrast, the non-PCGrad variant produces waves with a much smaller amplitude, completely ignoring both the outside boundaries as well as any obstacles inside the domain, so that reflections and interference patterns can hardly be observed. In addition, it has more difficulty generalising towards unseen cases, as became clear in Figure 6.6b.

The difference in performance can be explained as follows. Without gradient surgery, the learning objectives L_u and L_v were at some point in conflict with the boundary objective L_{bound}. From Figure 6.2, we derived that the network prioritised further optimisation of the former two terms over that of the latter. The observations that the non-PCGrad SWE-PINN ignores the outside boundaries, as well as obstacles inside the domain, and that the waves produced by the buoys (which are modelled as a sub-objective in L_{bound}) are of significantly smaller amplitude, is a direct result of the network under-prioritising the optimisation of the boundary loss term. The boundary objective is simply not well optimised for, leading to badly modelled interactions between the water and the buoys, obstacles, and outside boundary.

Besides this, the badly modelled boundary conditions have a negative impact on the training effectiveness of the remaining objectives as well. Namely, the oscillators are the drivers of this training process: By introducing disturbances on the water surface, the network is forced to start optimising for the hydrodynamical objectives L_h, L_u and L_v in the areas surrounding the oscillator. By under-prioritising the boundary loss term, the network has essentially suffocated the very thing that allows it to optimise for the other terms. The observation that terms L_u and L_v are able to reach lower loss values in Figure 6.2 than in Figure 6.1 is then likely a result of the network over-fitting towards *under-disturbed* water surfaces and this may explain why the non-PCGrad SWE-PINN shows worse generalisation performance.



(a) $t = 10$s **(b)** $t = 20$s **(c)** $t = 30$s **(d)** $t = 40$s **(e)** $t = 50$s

Figure 6.9: Waves reflecting on the boundary of a circular water basin after 10, 20, 30, 40 and 50 seconds. PCGrad was enabled during training, and this domain shape is not included in the training set.



Figure 6.10: Waves created by a buoy positioned towards the right of the domain wrap around a vertical obstacle in the centre of the space. Interference patterns appear in the area behind the obstacle.



Figure 6.11: SWE-PINN simulation of a double-slit experiment with an oscillation frequency of $f = 1/8$ s^{-1}.

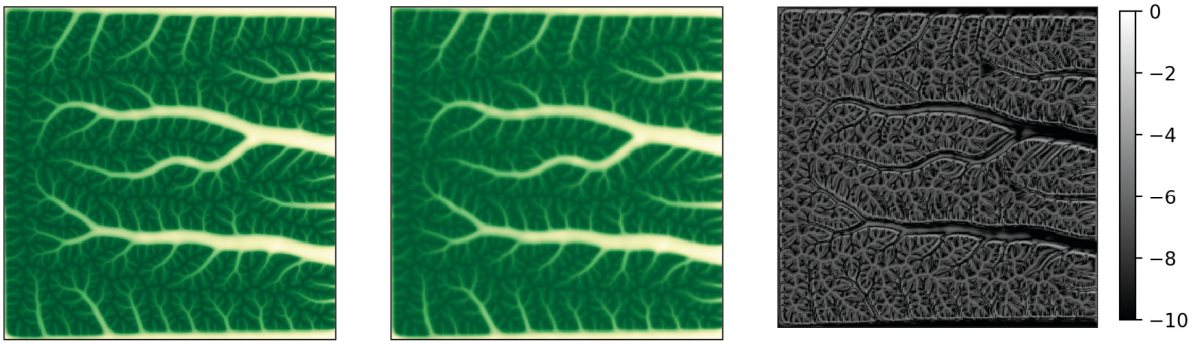
All combined, this is to say that a SWE-PINN can effectively learn to solve the 2D shallow water equations by training on a series of virtual water basins, with different configurations for obstacles and oscillators, and by applying gradient surgery to avoid conflicts between the various objectives and stimulate balanced learning. Therein, the oscillators are the drivers of the training process and the network would not learn without them. And this forms the answer to sub-question **RQ1a**.

By visually inspecting the hydrodynamical solutions produced by the PCGrad SWE-PINN in this section, we have observed four key characteristics of the solution that indicate that this setup is able to produce accurate hydrodynamical predictions. Firstly, waves travel evenly throughout the domain and experience visible damping as they travel further. Upon reaching one of the outside boundaries, or an obstacle inside the domain, the waves are reflected and start travelling in the opposite direction. Thirdly, waves can be seen bending around the edges of an obstacle. And lastly, the PINN is able to form complex interference patterns.

The first main question **RQ1** in this thesis can therefore be answered as follows. The data-less SWE-PINN described in Section 4.1 can be trained to solve the 2D shallow water equations in a closed water basin with flat bed topography and without vegetation, by using a small set of virtual water basins with various obstacles and oscillators to introduce the necessary disturbances in the water surface. Gradient surgery allows the network to find a proper balance between the different optimisation objectives. The produced hydrodynamical solutions are accurate from a qualitative standpoint, as all relevant hydrodynamic characteristics (waves, reflections, bending and interference) appear clearly in the simulations. The network furthermore generalises towards unseen cases.

6.2. Image Fitting CNN

The purpose of the Image Fitting CNN, detailed in Section 4.2.7, is to transform discrete images to spline latent representations. This is a necessary step before images produced by SFERE can be used as starting conditions during training. To assess the accuracy of the ImFit-CNN, it is presented with images from both the training set, as well as a separate validation set. All input images have a resolution of 800×800 pixels and, after being processed by the network, the spline images are sampled at an identical resolution to allow direct comparison. Logically, it is expected that the sampled outputs form an exact match to the input, however the precision of the match is limited by the maximum polynomial degree that can be represented with the chosen spline order. Thorough evaluation of the ImFit-CNN is crucial, as it is part of the training pipeline for the Hydro-PINN. Training of the ImFit-CNN took approximately four hours on an NVIDIA GeForce RTX 2080Ti.

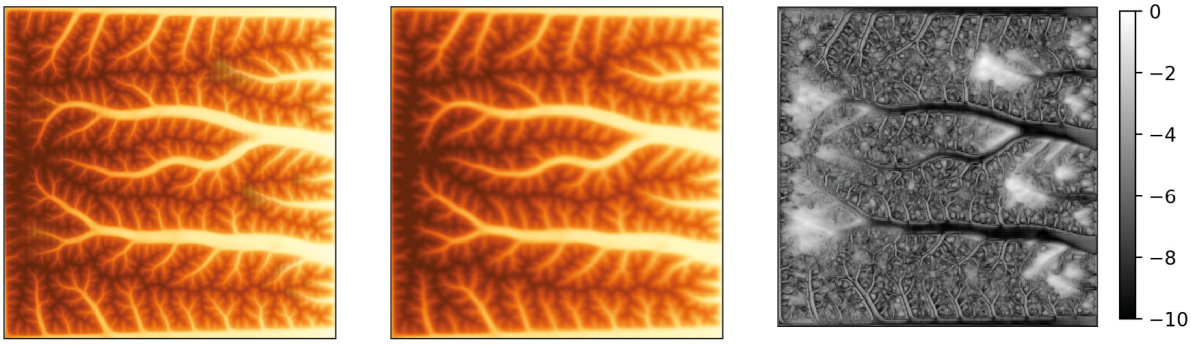


(a) SFERE output of the vegetation stem density after reaching equilibrium.

(b) Spline image output of the Image Fitting CNN, sampled at an 800×800 resolution.

(c) Spatial log loss between the SFERE output and its corresponding spline fit.

Figure 6.12: Vegetation stem density after reaching equilibrium in a SFERE simulation (left), alongside the corresponding spline image fitted by the ImFit-CNN (centre) and a spatial log loss for comparison of the two (right).



(a) SFERE output of the sedimentary bed elevation after reaching equilibrium.

(b) Spline image output of the Image Fitting CNN, sampled at an 800×800 resolution.

(c) Spatial log loss between the SFERE output and its corresponding spline fit.

Figure 6.13: Sedimentary bed elevation after reaching equilibrium in a SFERE simulation (left), alongside the corresponding spline image fitted by the ImFit-CNN (centre) and a spatial log loss for comparison of the two (right).

When presented with a discrete image of the vegetation density in a fully developed salt-marsh, as in Figure 6.12a, the ImFit-CNN produces a spline image that forms an almost perfect match in Figure 6.12b. Some areas of the spline image are slightly less sharp, but the overall structure of the landscape, including the channelisation at multiple spatial scales, is fully preserved. As a result of the lowered sharpness, the edges of the channels are slightly less defined and smaller channels are somewhat washed out. Spatial logarithmic comparison of both images, as in Figure 6.12c, confirms that most of the loss appears in areas with sharp features and high frequency signals in the image. Figure 6.13 shows the sedimentary bed elevation corresponding to this vegetation profile. The SFERE output (Figure 6.13a) suffers from slight numerical artifacts that appear as vertical lines in the landscape. After processing this image with the ImFit-CNN in Figure 6.13b, these artifacts are no longer

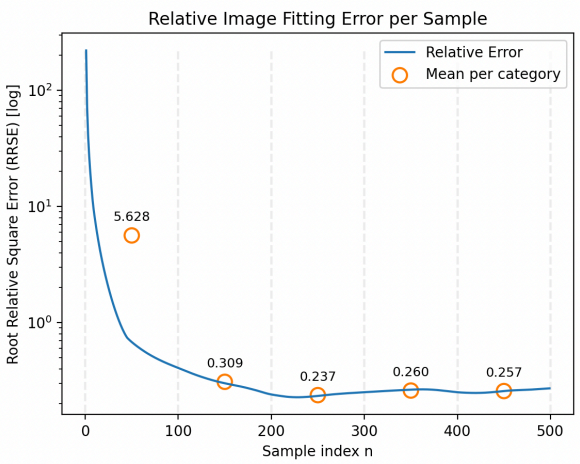


Figure 6.14: Root Relative Square Error for each sample in the validation set, consisting of 500 snapshots of a SFERE simulation that were taken according to an exponential time-scale (as detailed in Section 4.2.8) so that samples are evenly spread across the five output categories defined in Section 4.2.6. For each of these categories, the mean RRSE is indicated with an orange circle.

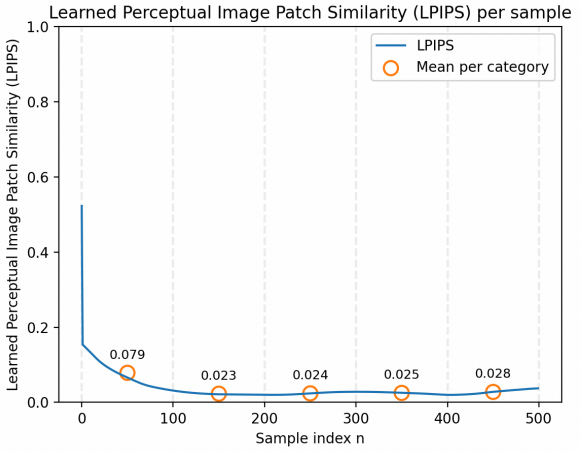


Figure 6.15: Learned Perceptual Image Patch Similarity (LPIPS, [79]) for each sample in the validation set, consisting of 500 snapshots of a SFERE simulation that were taken according to an exponential time-scale (as detailed in Section 4.2.8) so that samples are evenly spread across the five output categories defined in Section 4.2.6. The mean of each category is indicated with an orange circle.

visible. Figure 6.13c reveals the exact location of these numerical artifacts in the form of bright clouds that mostly appear towards the end of wider channels.

Both images in Figures 6.13a and 6.12a are part of the training set. Every sample in this set contains one image for each of the variables h, u, v, S and B. Because each variable works on a different value scale (e.g. $0 \leq h \lesssim 0.1$ and $0 \leq B \leq 1500$), the Root Relative Square Error (RRSE) is a suitable metric to quantify the performance of the network with respect to a set of samples and is calculated as:

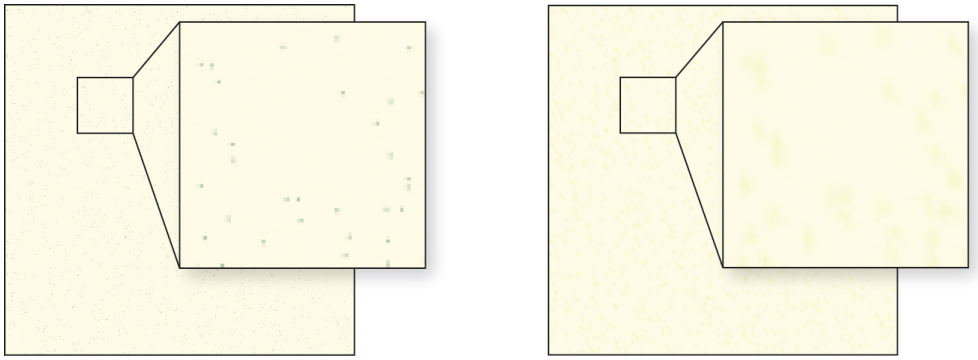
$$\mathrm{RRSE}(T, P) = \sqrt{\frac{\Sigma_{x,y}[P(x,y) - T(x,y)]^2}{\Sigma_{x,y}[T(x,y) - \bar{T}]^2}}$$

where T is the target image (i.e. the output produced by SFERE) with mean pixel value $\bar{T}$, P is the predicted image produced by the ImFit-CNN. As this metric is relative to the pixel-value scale of each image, it allows for a combined loss calculation by taking the average error of all five images in a sample:

$$E(i) = \frac{1}{5}\Big[\mathrm{RRSE}(T_{h,i}, P_{h,i}) + \mathrm{RRSE}(T_{u,i}, P_{u,i}) + \mathrm{RRSE}(T_{v,i}, P_{v,i}) + \mathrm{RRSE}(T_{S,i}, P_{S,i}) + \mathrm{RRSE}(T_{B,i}, P_{B,i})\Big]$$

where $T_{h,i}$ corresponds to the target water level height h of the i-th sample and $P_{h,i}$ to its prediction by the network and idem for the other variables. Any value of $E(i) < 1$ indicates that our model has better fitting capabilities than a random guess, and $E(i) = 0$ indicates a perfect match.

In addition to this direct and traditional metric, a perceptual similarity metric is used to compute the structural similarity between two images. In contrast to traditional comparison metrics, perceptual measures do not assume pixel-wise independence and rather aim to quantify *perceptual distance* [79]. As the network transforms images to a completely different image representation with Hermite splines, the exact positioning of features in the image might deviate ever so slightly, and perceptual metrics are less sensitive to these local disturbances. Learned Perceptual Image Patch Similarity (LPIPS, [79]) uses internal activations of deep neural network layers, for example trained on a high-level image classification task, and is found to be surprisingly useful for similarity comparisons.



(a) Numerical output of the vegetative stem density after one iteration of SFERE.

(b) Hermite spline image corresponding to the vegetation output after one iteration of SFERE.

Figure 6.16: Vegetation stem density after one iteration of SFERE (left) and its corresponding spline image that was fitted by the ImFit-CNN (right). After zooming, it becomes apparent that the image fitting network cannot find a proper fit to the vegetative initial condition used by SFERE. Vegetation tussocks are blurred and less well visible.

The ImFit-CNN performs with an average RRSE of 0.272 across the training set. An independent validation set is constructed by sampling the SFERE simulation according to the exponential timescale defined in Section 4.2.8. On this validation set, the ImFit-CNN performs with an average RRSE of 1.338, indicating notably worse performance compared to the training set. Figure 6.14 shows how the relative error is distributed among the validation samples, with worst performance across samples from the first category (samples 0-100). In fact, excluding samples from that category, the network performs with an average RRSE of 0.266 across samples 100-500, so that the validation error is approximately equal to the training error for that part of the validation set. The average scores for LPIPS are 0.029 across the training set and 0.036 on the validation set. Figure 6.15 indicates the perceptual similarity per sample, as well as the average score per output category.

Figure 6.16 zooms in on the vegetation in the starting condition of SFERE, with randomly placed vegetation tussocks, and its corresponding ImFit-CNN output and it is clearly visible that the ImFit-CNN struggles to find a good fit. These tussocks span precisely one pixel and thus form a very sharp signal in the image. With the chosen spline order, Hermite splines are not able to represent such sharp features very well and the spline image therefore shows blurry spots in those locations. In Figure 6.14, it can be seen that the error gradually decreases as we move away from the first samples towards the 100^{th} sample. In these very early iterations of SFERE, the small vegetation tussocks slowly start to diffuse into their neighbouring pixels drastically lowering their sharpness. As a result, the spline kernels can find an increasingly better fit to these vegetation patches and the error gradually lowers.

From these results, it is clear that the Image Fitting CNN is capable of accurately transforming a pixel image produced by SFERE to a corresponding Hermite spline representation, so that it can be used for the training of a PINN. The network generalises to images unseen during training, with the exception of images that contain sharp features. For the purpose of training our PINN, losing sharpness does not have to be a problem. Firstly, the PINN makes use of precisely the same spline order that is used in the Image Fitting CNN. Therefore, the PINN itself is also not able to capture and process these high frequency signals for which training cannot be positively or negatively affected by the presence or absence of sharp features in the training set. More importantly, one might consider the removal of very sharp features beneficial to the learning process, as it has a de-noising effect which has been illustrated in Figure 6.13.

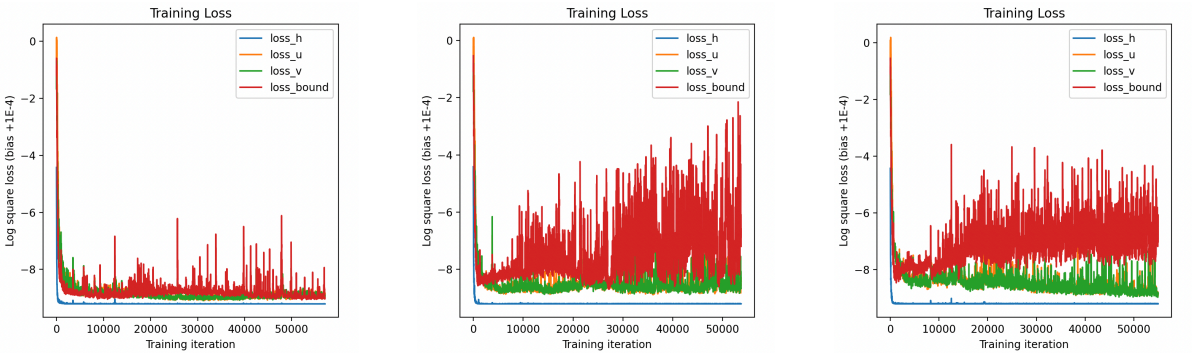
6.3. Physics Informed Hydrodynamics in Coastal Wetlands

This section focusses on solving the hydrodynamics in the more complex setting of a partially open coastal wetland, for which the Hydro-PINN model was developed in Section 4.2. As described in that section, this model is trained using images produced by SFERE [66], sampled from the five categories of spatial landscape structuring, identified in Figure 4.5. In this section, we evaluate the training effectiveness and hydrodynamical performance of this Hydro-PINN. We start by analysing the impact of the geological structuring inside the landscapes on the training effectiveness of the PINN in Section 6.3.1. Afterwards, the Hydro-PINN is trained in conjunction with a morphological solver to allow the sedimentary bed topography and vegetation densities to evolve during training and the resulting hydrodynamical performance is evaluated in Section 6.3.2. In that section, both a numerical and a physics-informed morphological solver are used and their impact on the training effectiveness of the Hydro-PINN is evaluated.

6.3.1. Analysing the Impact of Landscape Structuring on Training Effectiveness

With the Image Fitting CNN embedded in the training pipeline, the Hydro-PINN can be trained using landscapes generated by SFERE [66]. Five categories of landscape coarsening were defined in Figure 4.5 and an ablation study was setup in Section 4.2.8 to investigate how the training effectiveness of the PINN is affected by the structural characteristics of each category. Five Hydro-PINNs are separately trained on landscapes from each of the categories, and the results are presented and discussed in this section.

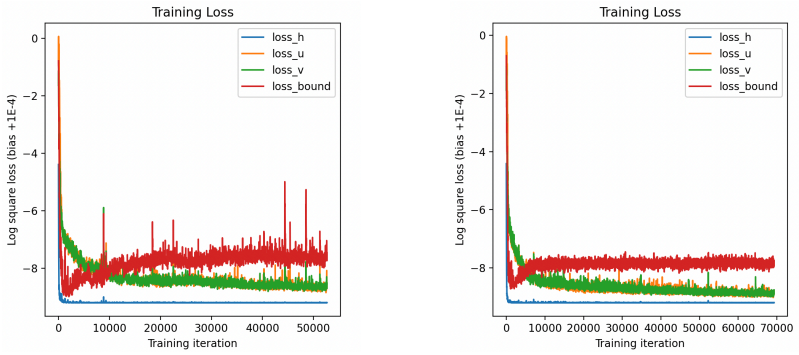
Training of each network took 36 hours on an NVIDIA A40 and the training losses are presented in Figure 6.17, where it can be seen that all networks have converged. However, these figures also indicate that **Hydro-PINN-Stage-2** and **Hydro-PINN-Stage-3** traversed the optimisation landscape with considerably more noise in Figures 6.17b and 6.17c, and that this is mostly caused by the boundary loss term. What can also be seen is that the boundary loss seems to increase after a relatively short training period in the case of **Hydro-PINN-Stage-4** and **Hydro-PINN-Stage-5** in Figures 6.17d and 6.17e, before reaching convergence.



(a) Training loss for **Hydro-PINN-Stage-1**

(b) Training loss for **Hydro-PINN-Stage-2**

(c) Training loss for **Hydro-PINN-Stage-3**



(d) Training loss for **Hydro-PINN-Stage-4**

(e) Training loss for **Hydro-PINN-Stage-5**

Figure 6.17: Training loss for five identical Hydro-PINNs, trained on sets of landscapes corresponding to the five categories defined in Figure 4.5.

The training effectiveness of these networks is evaluated by means of a qualitative assessment of the resulting hydrodynamical solutions. Specifically, we evaluate each network on the set of landscapes that it was trained on and look for visual clues that speak to the accuracy of the solution. The purpose of this study is to investigate what types of landscapes lead to more effective training. By training *effectiveness*, we thereby mean the degree to which the structural properties of the landscapes in the training set have allowed the network to find solutions that adhere to the physical laws inside *those* landscapes. Therefore, we do not test for generalisation in this evaluation.

This qualitative assessment is supported by a quantitative assessment of the network performance with respect to the PDE and boundary loss terms. Specifically, each network is evaluated on all samples of its training set, in all four landscape orientations, for a duration of 1000 iterations and the physics-informed losses are recorded. These losses cannot directly be used as a comparison metric across the networks and across the landscapes, as the loss magnitudes are influenced by the gradients and flow velocities inside each specific landscape. In other words, every landscape has a *characteristic scale*, which is determined by the magnitude of its gradients and fluid flows. To allow comparison of loss terms, they must first be *non-dimensionalised* with respect to the characteristic scale of each landscape.

During evaluation, the loss terms L_h, L_u, L_v, $L_{\text{bound,closed}}$ and $L_{\text{bound,open}}$ are normalised with respect to the corresponding characteristic scales S_h, S_u, S_v, $S_{\text{bound,closed}}$ and $S_{\text{bound,open}}$ for each landscape in the training set. We refer to the resulting normalised quantities as *residuals*. In total, we consider four residuals

$$\mathcal{R}_h = \frac{L_h}{S_h}, \quad \mathcal{R}_{uv} = \frac{L_u + L_v}{S_u + S_v}, \quad \mathcal{R}_{\text{bound}} = \frac{L_{\text{bound,closed}}}{S_{\text{bound,closed}}} \quad \text{and} \quad \mathcal{R}_{\text{bound,open}} = \frac{L_{\text{bound,open}}}{S_{\text{bound,open}}}$$

where the flow velocity loss terms have been combined into a single residual $\mathcal{R}_{uv}$, to remove asymmetries from the evaluation. While treated as one loss term during training, we separately compute residuals for the closed and open boundaries.

The characteristic scales are defined as the 90$^{\text{th}}$ percentile of the sum of magnitudes of each term in the corresponding PDE, computed across the landscape, based on the training landscapes produced by SFERE [66]. Concretely, S_h is defined as follows:

$$S_h = q_{90}\left(\left|\frac{\partial(hu)}{\partial x}\right| + \left|\frac{\partial(hv)}{\partial y}\right|\right)$$

where q_{90} indicates that we take the 90$^{\text{th}}$ percentile of the given quantity across the landscape. Similarly for the momentum loss terms L_u and L_v, we can define characteristic scales S_u and S_v as follows:

$$S_u = q_{90}\left(\left|\frac{\partial(h+S)}{\partial x}\right| + \left|u\frac{\partial u}{\partial x}\right| + \left|v\frac{\partial u}{\partial y}\right| + \left|\frac{\tau_{bx}}{\rho h}\right| + \left|\nabla^2 u\right|\right)$$

$$S_v = q_{90}\left(\left|\frac{\partial(h+S)}{\partial y}\right| + \left|u\frac{\partial v}{\partial x}\right| + \left|v\frac{\partial v}{\partial y}\right| + \left|\frac{\tau_{by}}{\rho h}\right| + \left|\nabla^2 v\right|\right).$$

As mentioned, two separate residuals are considered for the closed and open boundaries and the corresponding characteristic scales are computed as the 90$^{\text{th}}$ percentile of the gradient- and flow-magnitudes inside a 15% margin around the domain boundary:

$$S_{\text{bound,closed}} = q_{90}\left(\left|\frac{\partial h}{\partial x}\right| + \left|\frac{\partial h}{\partial y}\right| + |u| + |v|\right) \quad \text{and} \quad S_{\text{bound,open}} = q_{90}\left(\left|\frac{\partial h}{\partial x}\right| + \left|\frac{\partial h}{\partial y}\right|\right).$$

These characteristic scales are computed for each landscape inside the training set. In Figure 6.18, the continuity and momentum scales S_h and S_{uv} can be seen to increase as the landscape patterning coarsens further. Initially, the same holds for the characteristic boundary scales, before they decrease after $n \approx 180$ and eventually level out.

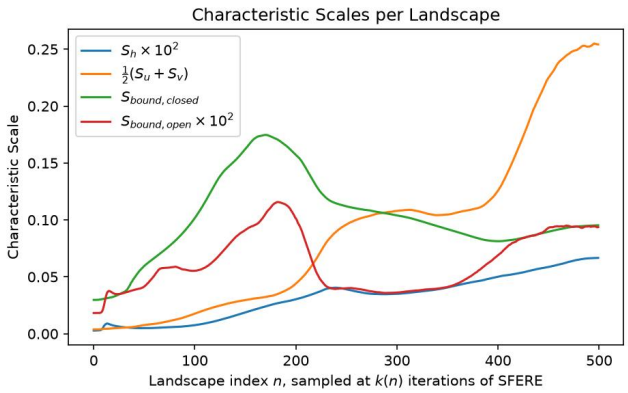


Figure 6.18: Characteristic scale for all landscapes inside the training set, with respect to each of the PDE and boundary loss terms. We present the average characteristic momentum scale of S_u and S_v to remove any directional asymmetries. Landscapes are sampled from SFERE [66], where the landscape with index n was sampled after $k(n)$ iterations.

Figures 6.19, 6.20, 6.21, 6.22 and 6.23 provide the hydrodynamic simulation output of each network over the course of 1000 iterations when evaluated on the last entry of their training set (i.e. the farthest coarsened landscape of that category), in the top right. Additionally, the aforementioned residuals are computed for all landscapes in its training set and the average residual values are presented over simulation time in the bottom right. The figures also illustrate the type of landscape that each network is trained on, by providing an example of the bottom topography and vegetation density from their training sets in the top left. These match with the categorisation from Figure 4.5. The layout of the domain is indicated in the bottom left. Aside from this temporal progression of the residuals per network in this study, Figure 6.24 presents the residual value for each landscape, averaged over 1000 iterations. In the remainder of this section, we discuss our qualitative observations for the solution of each model and immediately cross-reference them with the residuals in this figure to indicate how they reflect our assessment.

Both during training and evaluation of the networks, the bed topography and vegetation density remain fixed between iterations, so that the hydrodynamic solutions should reach an equilibrium. Indeed, all solutions seem to reach a stable equilibrium in Figures 6.19c, 6.20c, 6.21c, 6.22c and 6.23c, as little change is visible between iterations. This stability is supported by the almost constant time-progression of the residuals in Figures 6.19d, 6.20d, 6.21d, 6.22d and 6.23d.

In the most primitive stage of landscape development, **Hydro-PINN-Stage-1** is able to find a reasonable hydrodynamical solution in Figure 6.19c. The water layer accurately matches the uniform bed topography and the horizontal flow velocity u indicates that water accelerates as it moves down the slope. There is almost no flow in the vertical direction, which is expected given the direction of the slope. The open boundary is adequately modelled, judging by the positive and constant horizontal flow in the right margin of the domain. On all other edges of the domain, it is clearly visible that there is almost no flow, signalling a correctly modelled closed boundary. Aside from a circular artifact that appears in both velocity components after 200 iterations, and disappears after 600, the biggest inconsistency seems to appear in the horizontal flow velocity u, as it indicates water flowing uphill in a large portion of the domain. Also, a net non-zero flow can be observed in the vertical direction.

This may indicate that the PINN struggles to identify when and where in the domain the flow should be precisely zero. From a mathematical perspective, the task of our PINN is to find the appropriate *function value* for both the water layer thickness h and the flow components u and v by training on the temporal *rate of change* for those quantities as prescribed by the PDEs. Both the upwards flow in horizontal direction and the net non-zero flow in the vertical direction can be explained as an error that accumulates in the function value of u and v until an equilibrium is reached (i.e. until the network repeatedly produces zero rate-of-change for u and v, as dictated by the PDEs). This error indicates that the velocity components were not sufficiently optimised during training, and this is also reflected by the momentum residual $\mathcal{R}_{uv}$ in Figure 6.24b, where large losses appear for this **Hydro-PINN-Stage-1** between $0 \leq n < 100$.

Hydro-PINN-Stage-2 has more difficulty finding an appropriate hydrological solution, as can be seen in Figure 6.20c. Some structural aspects of the landscape are reflected in the solution, however large amounts of water seem to be flowing away from the open boundary and up the slope. Despite this, some water still drains through the open boundary, as seen by the strong positive flow near the right edge of the domain. The vertical flow reflects some of the channelisation in the soil, but this is much less refined in comparison to the starting condition. Compared to this starting condition, our water layer reflects almost none of the channelisation in the soil.

Comparing Figure 6.19 to Figure 6.20, it is clearly visible that the water layer h in Figure 6.19c forms a much closer match to the corresponding landscape in Figure 6.19a in comparison to how well the water layer in Figure 6.20c matches with the starting condition in Figure 6.20a. This observation is well reflected in the continuity residual in Figure 6.24a, where a sharp increase in normalised loss is observed as we move from **Hydro-PINN-Stage-1** ($0 \leq n < 100$) to **Hydro-PINN-Stage-2** ($100 \leq n < 200$). Figure 6.24b, however, shows that **Hydro-PINN-Stage-2** performs better in comparison to **Hydro-PINN-Stage-1**, contradicting our qualitative observations. Because the three PDEs in this system are coupled, it is possible that the solutions for u and v presented in Figure 6.20c are *more* correct in comparison to those of Figure 6.19c, *given* the respective solutions for h. This may explain why the visible low quality of the velocity solutions is not reflected by the residual $\mathcal{R}_{uv}$ alone.

Hydro-PINN-Stage-3 follows the structuring of the landscape much better in Figure 6.21c, where water clearly gathers inside the channels and consistently flows towards the open boundary to drain from the tidal prism. Water flows faster inside channels and the vertical flow indicates water moving away from elevated areas towards the channels. Compared to **Hydro-PINN-Stage-1** and **Hydro-PINN-Stage-2**, **Hydro-PINN-Stage-3** produces velocity solutions that are much more refined, indicating not only an accurate direction of flow but also more adequate flow magnitudes. The appropriateness of these solutions is reflected by the $\mathcal{R}_h$ and $\mathcal{R}_{uv}$ residuals in Figure 6.24a and Figure 6.24b, respectively, as they indicate similar or lower levels of normalised loss between $200 \leq n < 300$ in comparison to **Hydro-PINN-Stage-2** ($100 \leq n < 200$).

Judging from Figure 6.21c, both the open and closed boundaries seem to be appropriately modelled by **Hydro-PINN-Stage-3**. Flow velocities approach zero as they reach the closed boundary and the water seems to smoothly flow out through the open boundary, as can be seen by the strong positive outflow. Still, the boundary residuals in Figure 6.24c and Figure 6.24d, for the closed and open boundary residuals respectively, disagree with this observation by signalling worse boundary performance in comparison to earlier models. In Section 4.2.3, it is explained how we refrain from modelling separate boundary conditions for the x- and y-directions. As a result, the closed boundary is modelled by assuming zero-gradient in h and zero flow in both x- and y-directions. Similarly, for the open boundary we assume zero-gradient in h in both directions. In Figure 6.21c, it can clearly be seen that these assumptions do not hold in the along-boundary direction. The soil incisions near the open boundary, for example, force the gradient in the water layer along that boundary to be non-zero. In other words, the landscape profile in Figure 6.21a forces the PINN to move away from these false boundary assumptions and accept higher losses for the boundary residuals.

From stage 4 onwards, growing vegetation starts to affect the landscape structuring more significantly and **Hydro-PINN-Stage-4** shows an accurate response to this in Figure 6.22c. Water accumulates in the channels on multiple branching scales. In vegetated regions, there seems to be a drastic slowdown in horizontal flow, and this is an adequate prediction given that bed friction increases with vegetation density. The vertical flow accurately reflects water moving away from uplands, into the nearby channels. Moreover, the vertical flow component v shows zero vertical flow in areas that have fallen dry in the domain. A similar effect can be seen in u, but is less clearly visible. The SWEs tend to turn unstable in (near-)dry areas, with velocity components reaching extremely large values. Section 4.2.2 explained how we mitigated such instabilities by imposing a zero-flow condition in areas where the water level reaches below a critical minimum. Qualitatively, Figure 6.21c indicates that this moving boundary strategy is effective for preventing such instabilities.

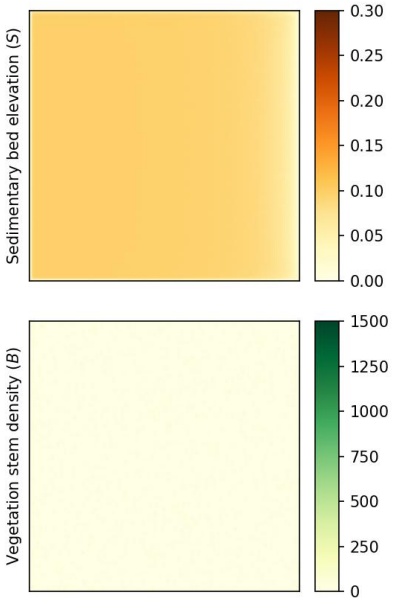
The PDE residuals $\mathcal{R}_h$ and $\mathcal{R}_{uv}$ in Figures 6.24a and 6.24b, respectively, indicate similar performance between **Hydro-PINN-Stage-3** and **Hydro-PINN-Stage-4** models with respect to the continuity and momentum equations. However, we also observe that water more clearly gathers in the channels in Figure 6.22c in comparison to Figure 6.21c, which shows much larger areas of dryness. Also, the

magnitude of flow in Figure 6.21c is generally much larger than in Figure 6.22c. This decrease in flow speed between **Hydro-PINN-Stage-3** ($200 \leq n < 300$) and **Hydro-PINN-Stage-4** ($300 \leq n < 400$) is a natural result of the growing vegetation in the domain and is therefore an accurate prediction. But it can also explain why more water gathers in the channels in Figure 6.22c in comparison to Figure 6.21c. In the latter, water drains from the tidal prism more quickly, following the large flow magnitudes. And because the water injection term H_{in} remains identical between models, this results in less water in the domain and in the channels. These considerations taken together, it is plausible that the continuity and momentum residuals $\mathcal{R}_h$ and $\mathcal{R}_{uv}$ are similar between the two PINNs.

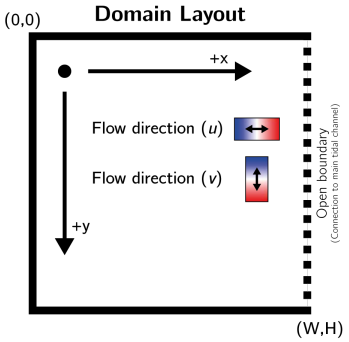
Similar observations can be made in Figure 6.23c, where **Hydro-PINN-Stage-5** was trained on landscapes from the fifth and final category of landscape coarsening. Water gathers in the channels even more clearly, on multiple branching scales, and uplands are accurately predicted to be dry. In those areas, both flow velocity components u and v are modelled close to zero by the PINN, like was also observed in the evaluation of **Hydro-PINN-Stage-4** above. Here, the effect of this moving boundary condition is even more clearly visible in Figure 6.23c. Flow velocities have lowered even further in Figure 6.23c in comparison to Figure 6.22c and Figure 6.21c, and this is an appropriate prediction given the increased vegetation density. Finally, in Figure 6.23c, it can be seen that the flow velocity solutions are less sharply defined in the left portion of the domain. When looking the corresponding hydrodynamic solution from SFERE in this landscape, presented in Figure 6.23c as the starting condition, we observe that the flow magnitude decreases as the channels propagate further into the landscape (i.e. away from the open boundary) and that higher frequency signals appear in the image as the channels become narrower. PINNs, and neural networks in general, have the tendency to learn low-frequency signals first before turning to high-frequency ones [50]. This may explain why our PINN is able to find more accurate solutions in areas where the channels are wider, so that more water can flow with higher speeds, compared to areas with narrow channels and tiny fast-fluctuating flows.

Similar to earlier models, the PDE residuals $\mathcal{R}_h$ and $\mathcal{R}_{uv}$ agree with our observations in Figure 6.24a and Figure 6.24b, respectively, showing approximately equal normalised loss values for both terms in comparison to **Hydro-PINN-Stage-3** and **Hydro-PINN-Stage-4**. The boundary residuals $\mathcal{R}_{\text{bound,closed}}$ and $\mathcal{R}_{\text{bound,open}}$ seem to have slightly decreased for **Hydro-PINN-Stage-4** ($300 \leq n < 400$) and **Hydro-PINN-Stage-5** ($400 \leq n < 500$) in comparison to **Hydro-PINN-Stage-3** ($200 \leq n < 300$) and this is likely a result of the lowered flow velocities in general throughout the landscape that we observed and discussed above.

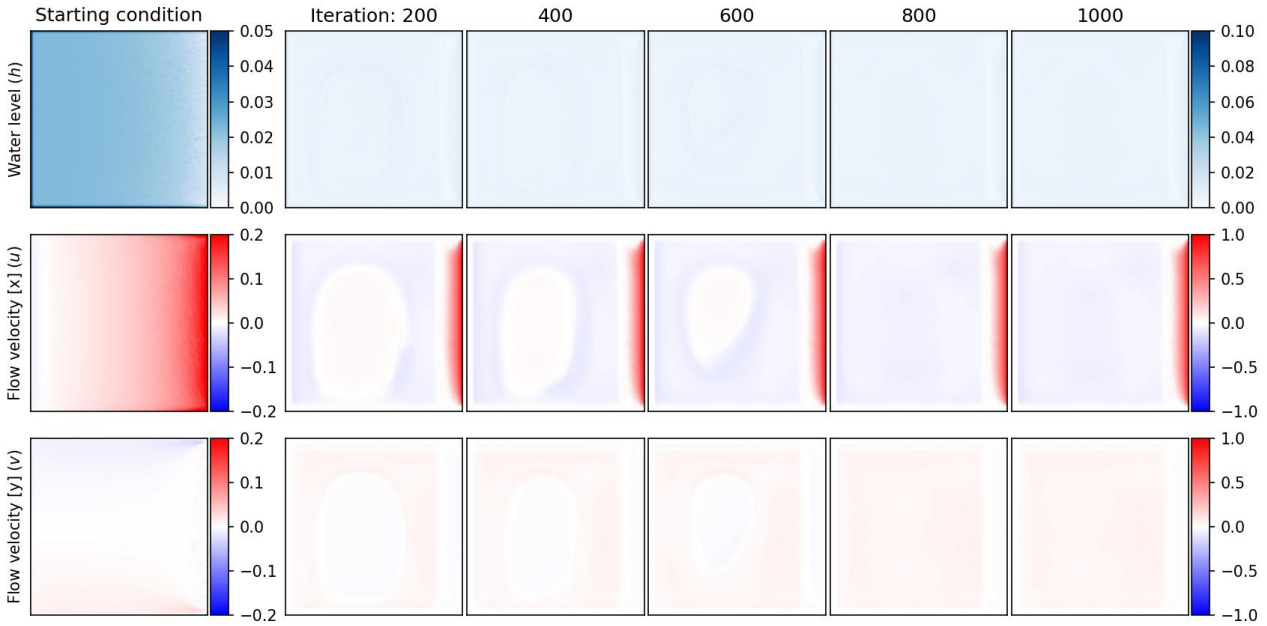
Across all Figures 6.19c, 6.20c, 6.21c, 6.22c and 6.23c, the PINNs consistently predict flow magnitudes that are much larger than the starting condition. A similar effect can be observed in the water layer thickness h as well, most notably in Figures 6.21c, 6.22c and 6.23c. These increased magnitudes are likely a result of the wide value range that was chosen for the in- and outputs of the network for each quantity in the non-dimensionalisation of Section 4.2.5.



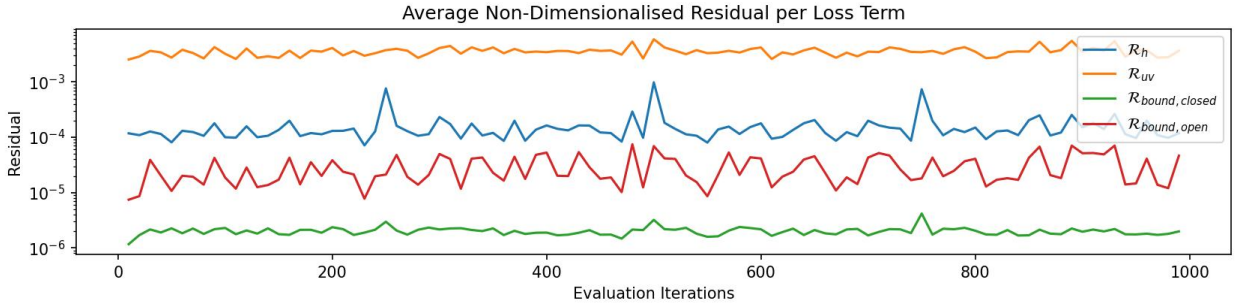
(a) Landscape structuring (Stage 1)



(b) Domain layout

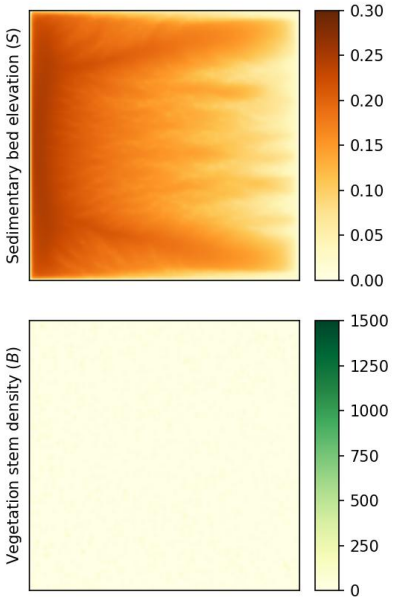


(c) Hydrodynamic solution predicted by **Hydro-PINN-Stage-1** after 200, 400, 600, 800 and 1000 iterations on SFERE [66] sample $k(99)$.

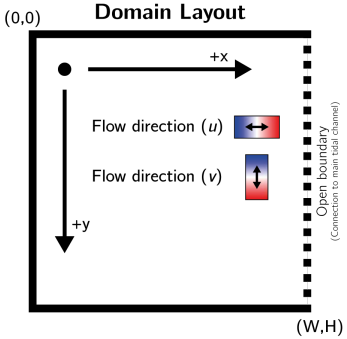


(d) Non-dimensionalised continuity, momentum and boundary residuals, over the course of 1000 iterations, averaged over all 100 landscapes in the training set.

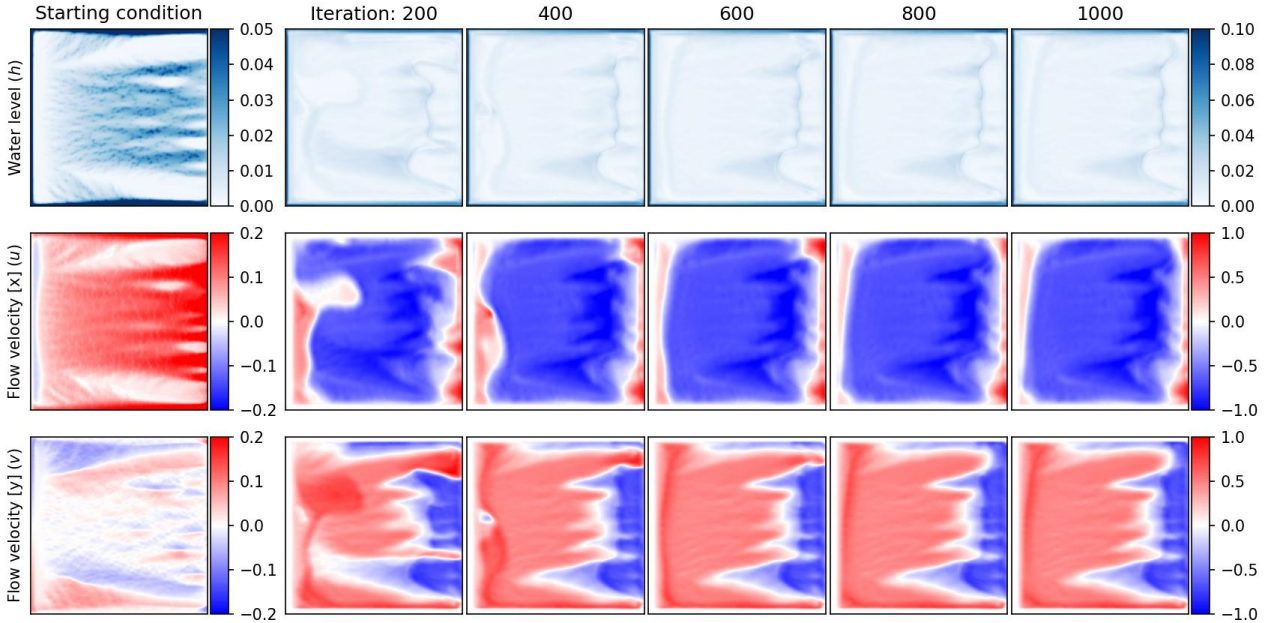
Figure 6.19: Hydrodynamic solution as predicted by **Hydro-PINN-Stage-1** when trained and evaluated on 100 static landscapes, sampled from SFERE [66] inside the first category of landscape coarsening defined in Figure 4.5a.



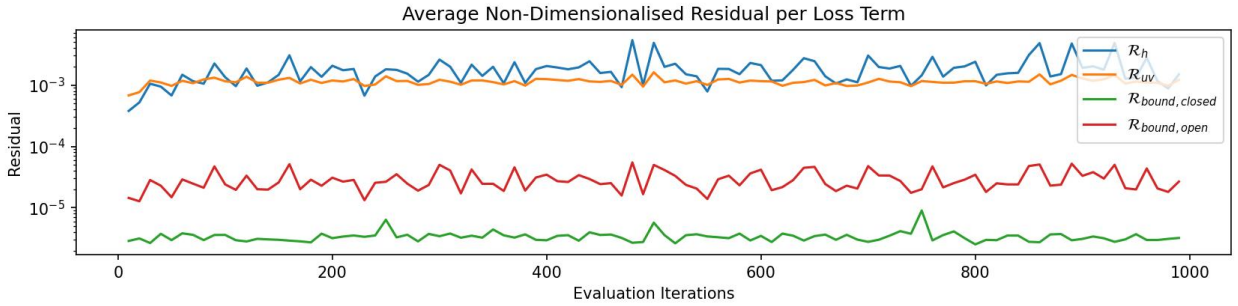
(a) Landscape structuring (Stage 2)



(b) Domain layout

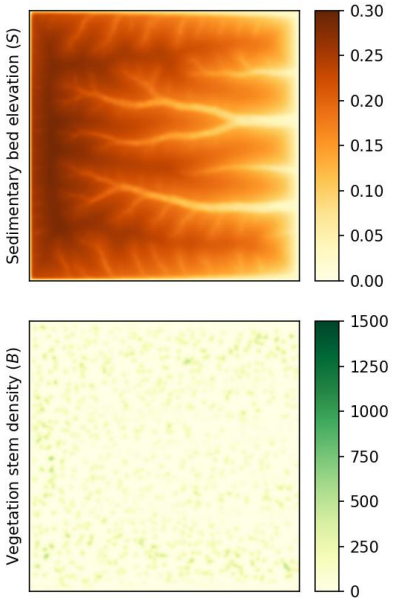


(c) Hydrodynamic solution predicted by **Hydro-PINN-Stage-2** after 200, 400, 600, 800 and 1000 iterations on SFERE [66] sample $k(199)$.

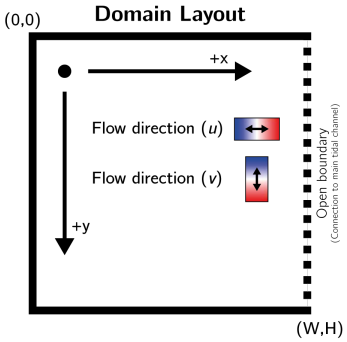


(d) Non-dimensionalised continuity, momentum and boundary residuals, over the course of 1000 iterations, averaged over all 100 landscapes in the training set.

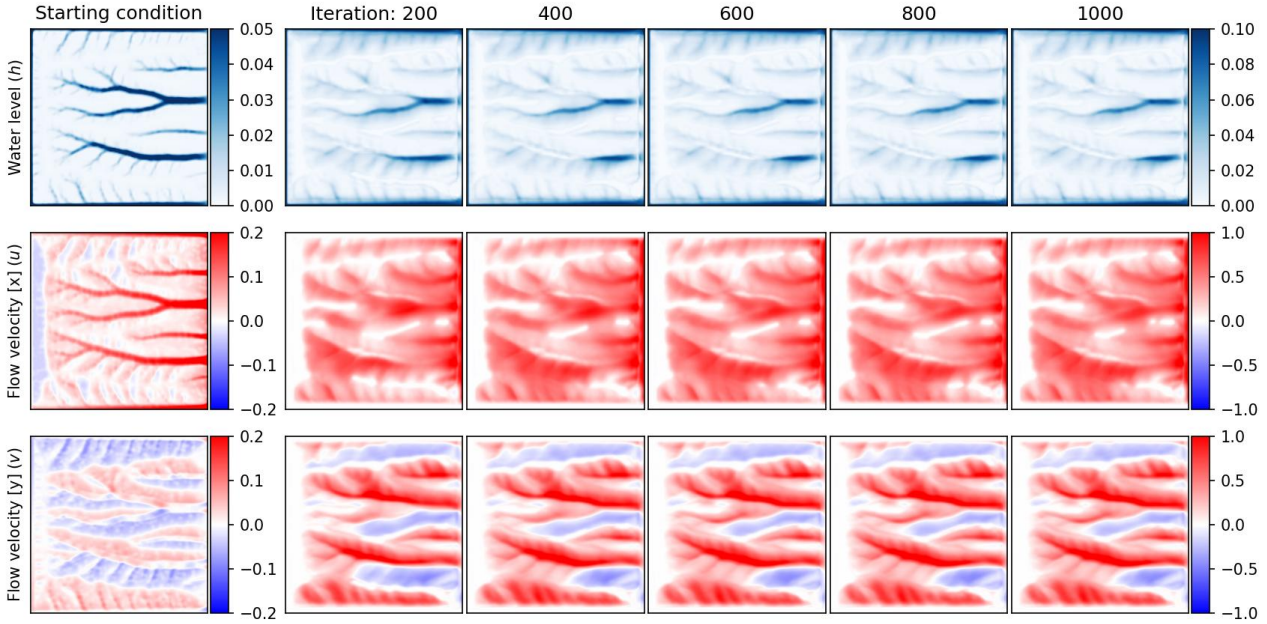
Figure 6.20: Hydrodynamic solution as predicted by **Hydro-PINN-Stage-2** when trained and evaluated on 100 static landscapes, sampled from SFERE [66] inside the second category of landscape coarsening defined in Figure 4.5b.



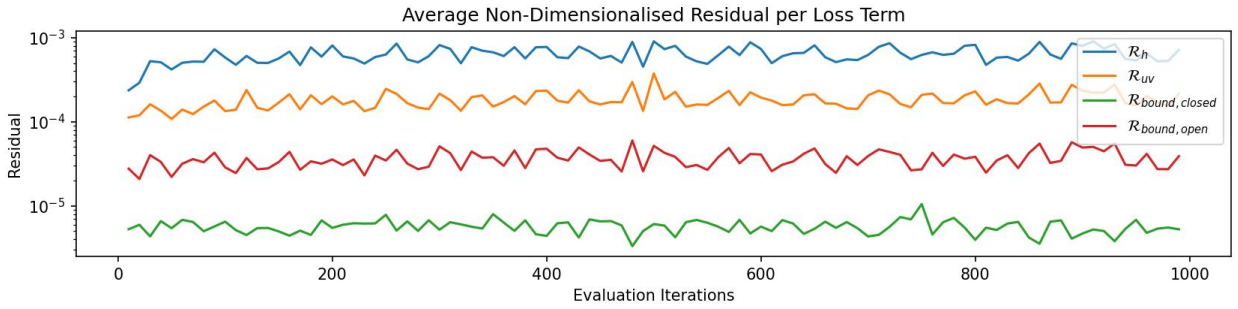
(a) Landscape structuring (Stage 3)



(b) Domain layout

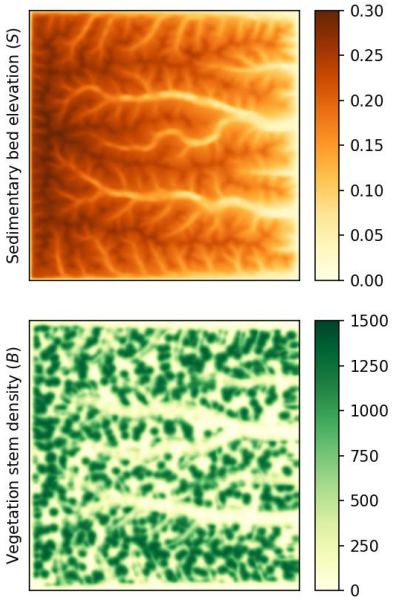


(c) Hydrodynamic solution predicted by **Hydro-PINN-Stage-3** after 200, 400, 600, 800 and 1000 iterations on SFERE [66] sample $k(299)$.

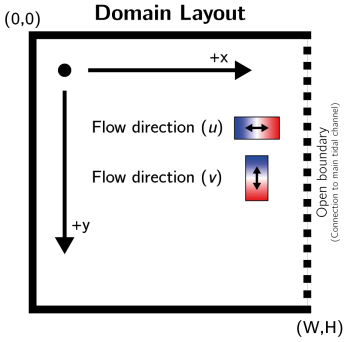


(d) Non-dimensionalised continuity, momentum and boundary residuals, over the course of 1000 iterations, averaged over all 100 landscapes in the training set.

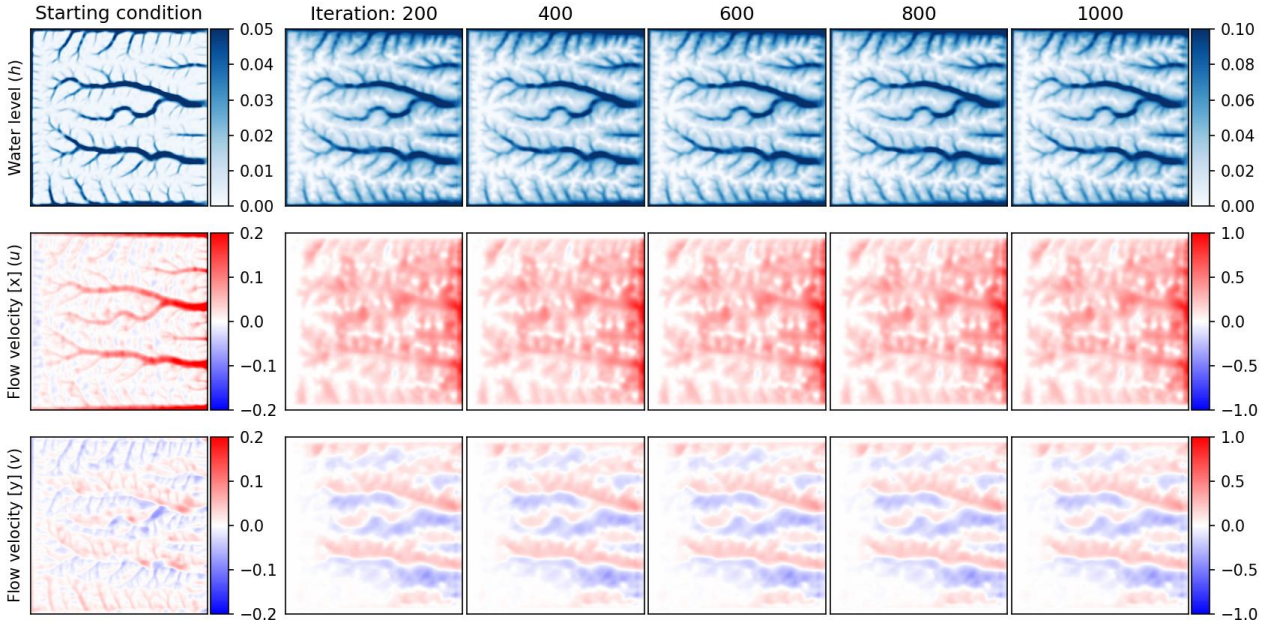
Figure 6.21: Hydrodynamic solution as predicted by **Hydro-PINN-Stage-3** when trained and evaluated on 100 static landscapes, sampled from SFERE [66] inside the third category of landscape coarsening defined in Figure 4.5c.



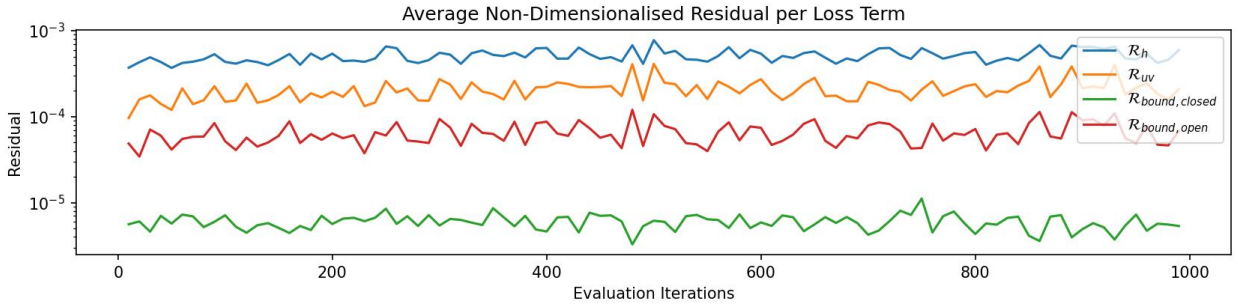
(a) Landscape structuring (Stage 4)



(b) Domain layout

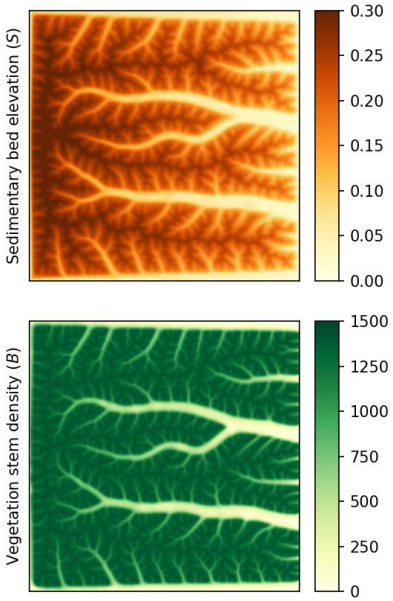


(c) Hydrodynamic solution predicted by **Hydro-PINN-Stage-4** after 200, 400, 600, 800 and 1000 iterations on SFERE [66] sample $k(399)$.

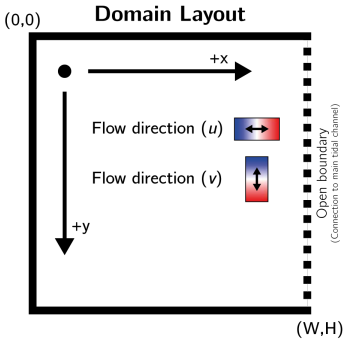


(d) Non-dimensionalised continuity, momentum and boundary residuals, over the course of 1000 iterations, averaged over all 100 landscapes in the training set.

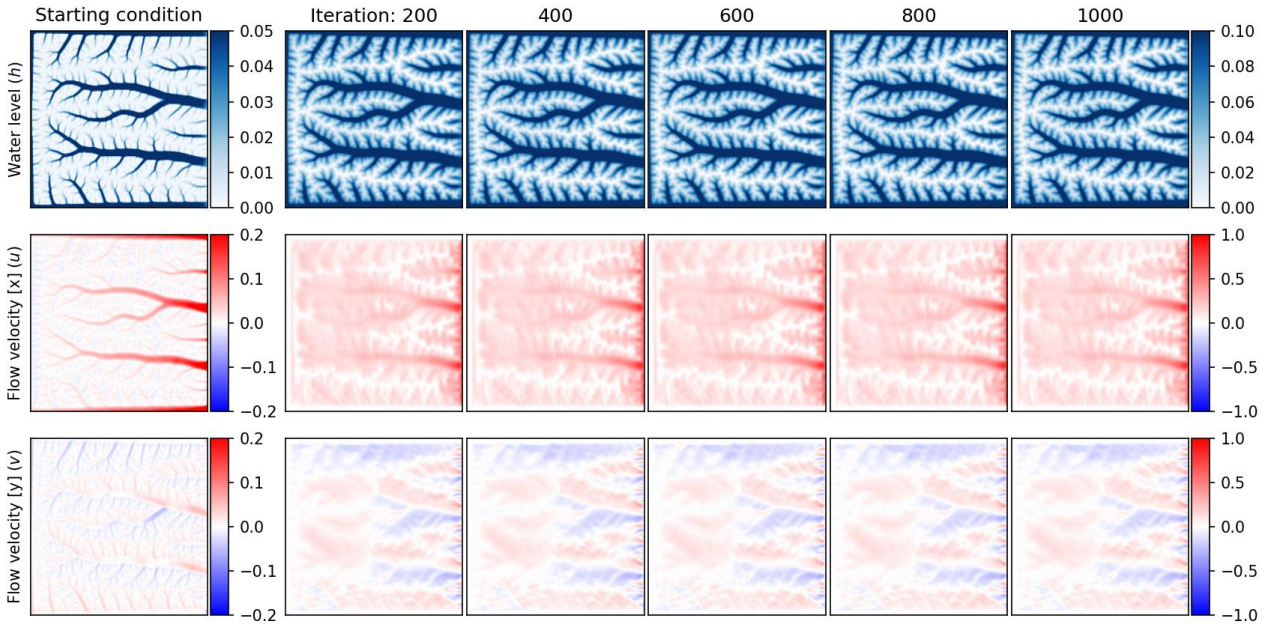
Figure 6.22: Hydrodynamic solution as predicted by **Hydro-PINN-Stage-4** when trained and evaluated on 100 static landscapes, sampled from SFERE [66] inside the fourth category of landscape coarsening defined in Figure 4.5d.



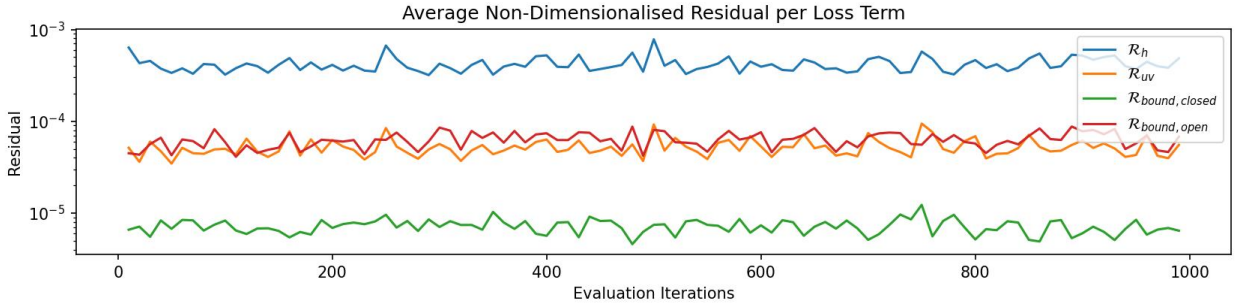
(a) Landscape structuring (Stage 5)



(b) Domain layout

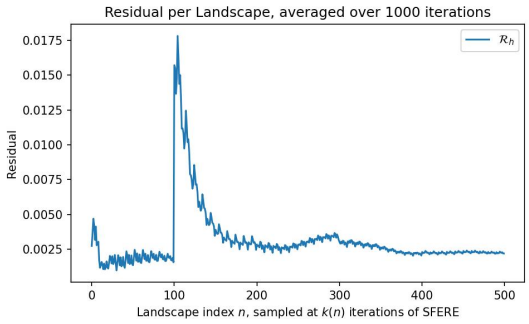


(c) Hydrodynamic solution predicted by **Hydro-PINN-Stage-5** after 200, 400, 600, 800 and 1000 iterations on SFERE [66] sample $k(499)$.

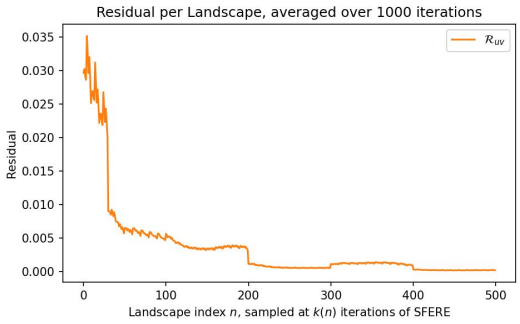


(d) Non-dimensionalised continuity, momentum and boundary residuals, over the course of 1000 iterations, averaged over all 100 landscapes in the training set.

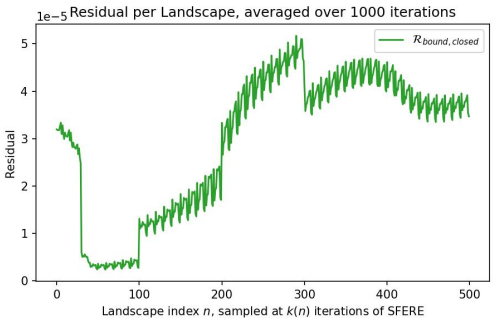
Figure 6.23: Hydrodynamic solution as predicted by **Hydro-PINN-Stage-5** when trained and evaluated on 100 static landscapes, sampled from SFERE [66] inside the fifth category of landscape coarsening defined in Figure 4.5e.



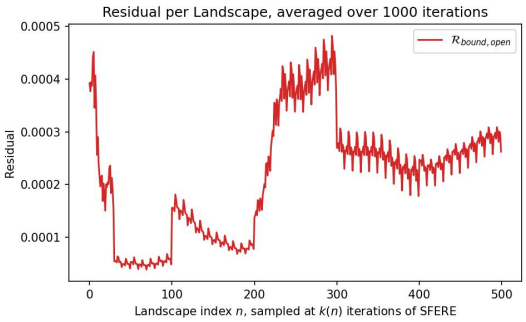
(a) PDE residual for the average continuity loss for each landscape.



(b) Combined PDE residual for the average flow velocity losses for each landscape.



(c) Boundary residual for the average closed boundary loss for each landscape.



(d) Boundary residual for the average open boundary loss for each landscape.

Figure 6.24: Non-dimensionalised residuals for each of the PDE and boundary loss terms, evaluated for all landscapes in the training sets and averaged over 1000 iterations.

The results presented in this section show that landscape structuring has a noticeable impact on the training effectiveness and simulation performance of the Hydro-PINN. Even though all models in this study seem to have converged relatively quickly, their predictions reveal a notable difference in hydrodynamic performance. Convergence is therefore in this setting not a suitable metric to measure training effectiveness. Qualitative assessment of the solutions indicates that the PINN finds more appropriate hydrodynamical solutions after training on landscapes with more complex spatial features. Concretely, **Hydro-PINN-Stage-3**, **Hydro-PINN-Stage-4** and **Hydro-PINN-Stage-5** produce more accurate hydrodynamical predictions compared to **Hydro-PINN-Stage-1** and **Hydro-PINN-Stage-2**, given the type of landscape that they were trained on.

The normalised PDE residuals $\mathcal{R}_h$ and $\mathcal{R}_{uv}$ are in agreement with this qualitative assessment, showing particularly large losses for $\mathcal{R}_{uv}$ in the case of **Hydro-PINN-Stage-1** and for $\mathcal{R}_h$ in the case of **Hydro-PINN-Stage-2**. This matches with the hydrodynamical components that were modelled inappropriately in those cases. However, we have also seen that a low residual for one of the terms does not suffice to conclude that the corresponding component of the solution is accurately modelled. This can happen, because the PDEs are tightly coupled, so that high losses for just one of the residuals can qualitatively negatively influence neighbouring terms.

The boundary residuals $\mathcal{R}_{\text{bound,closed}}$ and $\mathcal{R}_{\text{bound,open}}$, on the other hand, indicate worse performance with respect to the boundary conditions as the networks are trained on further coarsened landscapes. Qualitatively, this increase in boundary loss is not visible and it is likely caused by our naively modelled boundary conditions. From a qualitative standpoint, both the closed and open boundaries seem to be accurately modelled in all five models. This likely means that the network prioritises optimisation of the PDE loss terms, as a result of which the zero-gradient and zero-flow conditions in the direction along the boundary cannot be met, resulting in higher boundary residual values.

In Section 4.2.8, it is hypothesised that the difference in performance would be connected to the rate at which the landscapes drain, as higher drainage rates lead to more dryness in the domain. Across all results presented in this section, however, there appears to be significantly less dryness in the landscapes compared to the preliminary results presented in Section 4.2.8. In fact, these figures indicate that dryness appears *more* frequently in *later* developmental stages, as the ridges between the channels rise above the water level. This is to say that Figure 6.23 contains more and larger areas of dryness while still demonstrating higher training effectiveness. The hypothesis in Section 4.2.8 is therefore rejected and an alternative explanation must be sought.

Classical machine learning theory can offer an explanation for the performance difference, by looking at the data-variance in each of the training sets. In a traditional ML setting, the training performance largely depends on the *variance* of the datasets used for training. By variance, we mean that the entries of the dataset should be sufficiently different to provide the network with rich information. In our PINNs, this concept translates directly to the *spatial variance* of our training environments. Concretely, this means that our Hydro-PINN can learn more effectively on a fully developed salt marsh, with channels at multiple branching scales and vegetation in various patterns, than on a mudflat that has almost no spatial features. More spatial variance in the landscapes can lead to more effective training, and the results presented in this section reflect this reasoning accurately.

The increase in training effectiveness can also be explained by means of the characteristic scales that were introduced in the beginning of this section. Each landscape has such a characteristic scale, depending on the magnitude of the terms inside each of the PDEs. During training, the PINN has to learn the relationship between each of these PDE terms and the outcome of the simulation. Landscapes featuring larger magnitudes of gradients and flow velocities can serve the network with richer information, as the individual terms have a more significant effect on the outcome of the PDE. In Figure 6.18, we saw that farther coarsened landscapes are generally associated with larger characteristic scales for each of the terms, and this can therefore explain why networks trained on these landscapes learn to solve the PDEs more effectively.

6.3.2. Hydrodynamics in Static Landscapes

This section focusses on the hydrodynamical solutions, as produced by the Hydro-PINN that is described in Section 4.2, after training in three different configurations. Firstly, we consider the Hydro-PINNs that were trained as part of the ablation study, which was setup in Section 4.2.8 and evaluated in Section 6.3.1, to investigate how landscape structuring impacts training effectiveness. These five PINNs were trained on sets of *static* landscapes corresponding to the five categories identified in Figure 4.5. By static, we mean that the sedimentary bed elevation and vegetation density remains fixed between iterations. Throughout this section, we collectively refer to these five PINNs and their solutions as the **Hydro-PINN-Static**. Secondly, we consider a Hydro-PINN that was trained alongside a numerical morphological solver, in a hybrid configuration that is described in detail in Section 4.2.9. The numerical solver provides the network with sedimentary bed topographies and vegetation densities that are dynamic over the course of training. We further refer to this model as the **Hydro-PINN-Hybrid**. Finally, the hydrodynamic component of the Saltmarsh-PINN from Section 4.3 is in this section evaluated in isolation. Since the Saltmarsh-PINN is built around an almost identical implementation of the Hydro-PINN, evaluating it separately allows us to compare the performance of the Hydro-PINN after training alongside two PINNs for the morphological quantities versus with a numerical solver. We further refer to this model as the **Hydro-PINN-Saltmarsh**.

These three PINNs are in this section evaluated on static landscapes, with bed topographies and vegetation profiles that remain fixed between iterations. Similar to Section 6.3.1, this means that the solutions should reach an equilibrium. The solutions are evaluated and compared qualitatively, including a comparison with SFERE [66], and the non-dimensionalised residuals, derived in Section 6.3.1, are presented and connected to our qualitative observations. Like before, the networks are separately evaluated on landscapes from each of the categories defined in Figure 4.5. Figure 6.25 presents the water layer thickness h and both flow velocity components u and v for each of these landscape stages, as produced by SFERE [66], and Figures 6.26, 6.27 and 6.28 present the hydrodynamical solutions for the **Hydro-PINN-Static**, **Hydro-PINN-Hybrid** and **Hydro-PINN-Saltmarsh**, respectively, following the same structure. The PDE and boundary residuals are presented for each PINN in Figure 6.29.

Upon evaluating the **Hydro-PINN-Static** as part of the ablation study in Section 6.3.1, we observed that the PINN consistently predicts flows with much larger magnitudes in comparison to SFERE [66]. A similar, but more subtle, effect was observed in the water layer thickness h as well. Figure 6.26 places the various solutions that were covered in that section together and reveals the magnitude difference, in comparison to the predictions made by SFERE in Figure 6.25, even more clearly. The **Hydro-PINN-Hybrid** shows a similar magnitude increase in Figure 6.27. What further becomes apparent is that both SFERE and the **Hydro-PINN-Static** produce stronger flows in stages 2 and 3 in comparison to stages 4 and 5. This follows naturally, as the more extensive vegetation coverage (illustrated in Figure 4.5) leads to higher friction levels in stages 4 and 5. Furthermore, vegetation in the landscape leads to more complex channel networks, with more and narrower channels [66], effectively reducing the flow speed further. In Figure 6.27, it can be seen that the **Hydro-PINN-Hybrid** seems to neglect this vegetation-induced slowdown as it produces flows with similar magnitudes in all simulation stages. In terms of magnitude, the **Hydro-PINN-Saltmarsh** seems to reflect this slowdown in stages 4 and 5 slightly more accurately.

To understand why the **Hydro-PINN-Hybrid** misses the slowdown in water flow, we turn to the type of landscapes that it was trained on. The **Hydro-PINN-Static** is trained on fixed landscapes, meaning that the complexity of the landscape structuring (i.e. the characteristic scale of the individual PDE terms) remains over the course of training. During training of the **Hydro-PINN-Hybrid**, the landscapes are in each iteration updated by the numerical solver that predicts an updated state for the bed topography and vegetation density based on the hydrodynamic solutions predicted by the PINN.

With the large flow-rates predicted by our PINNs, sediment erosion-rates and vegetation mortality-rates increase dramatically. In such conditions, the scale-dependent feedbacks responsible for the formation of (complex) channel networks are disrupted and the recursive channelisation no longer occurs. Moreover, any channel network that was initially present in the landscape is quickly eroded away by the strong flows. With less vegetation in the landscape to stabilise the flow, the elevated flow-speeds are sustained so that the recursive scale dependent feedbacks cannot be recovered. As a result, the overall structural complexity (i.e. characteristic scales) of the landscapes in the training

set is limited by the maximum spatial complexity that can be supported in such strong-flow conditions. Eventually, the training set only contains landscapes with a structural organisation that can be sustained by the stronger flows, and the network can thus not learn to produce weaker flows. Thereby, the network effectively never learns to distinguish between regions where a stronger or weaker flow is appropriate.

Comparing Figures 6.27 and 6.26, it can further be noticed that the **Hydro-PINN-Static** captures the structure of the landscape in the flow velocity components u and v more accurately in stage 3 in comparison to **Hydro-PINN-Hybrid**. In stage 2, however, it is the other way around and **Hydro-PINN-Hybrid** finds a more appropriate solution for u and v compared to **Hydro-PINN-Static**, where water no longer flows uphill. Similar reasoning to the above paragraph can offer an explanation here as well. The **Hydro-PINN-Hybrid** has difficulty distinguishing between strong and weak flows, as a result of its training conditions, and therefore produces flows in the solution for stage 3 with an almost constant magnitude across the landscape. Where **Hydro-PINN-Static** predicts stronger flows in the channels and weaker ones in uplands in Figure 6.26, the **Hydro-PINN-Hybrid** is unable to distinguish between the two.

Both our observations that the **Hydro-PINN-Hybrid** does not capture the reduced flow speeds in stages 4 and 5 and that it is seemingly unable to make an accurate distinction between regions where strong or weak flows are more appropriate, are well reflected in the momentum residual $\mathcal{R}_{uv}$ in Figure 6.29b. Across all stages, it reports a higher normalised momentum loss for the **Hydro-PINN-Hybrid** in comparison to the **Hydro-PINN-Static**. In stage 1, this accurately reflects the qualitatively far less accurate prediction made by the **Hydro-PINN-Hybrid** in Figure 6.27 compared to that of **Hydro-PINN-Static** in Figure 6.26. In stages 2 and 3, the increased loss is likely caused by the inability of the **Hydro-PINN-Hybrid** to distinguish between flow-speeds, and in stages 4 and 5 it is likely because the solutions of the **Hydro-PINN-Hybrid** do not reflect the expected reduction in flow speed caused by extensive vegetation coverage.

Judging by the water layer thickness h, the **Hydro-PINN-Hybrid** seems to consistently produce more accurate solutions in comparison to **Hydro-PINN-Static** from a qualitative standpoint. The water layer predicted by **Hydro-PINN-Hybrid** in Figure 6.27 captures the underlying landscape structuring more sharply and consistently in comparison to the **Hydro-PINN-Static** in Figure 6.26. However, in comparison to the simulation of SFERE [66] in Figure 6.25, the **Hydro-PINN-Hybrid** generally produces noticeably higher water levels, even in areas where dryness can naturally be expected. The **Hydro-PINN-Static** predicts dryness in the water layer h visibly more accurately and this is likely the reason why the continuity residual $\mathcal{R}_h$ reports a higher loss value for **Hydro-PINN-Hybrid** compared to **Hydro-PINN-Static** across all stages in Figure 6.29a. In stages 3, 4 and 5, the **Hydro-PINN-Hybrid** predicts dryness more accurately than in stages 1 and 2, and this is why its continuity residual is lower in those stages.

In the evaluation of the ablation study in Section 6.3.1, we found that the network was able to produce a stable solution for the flow velocities u and v in areas where the water layer thickness approaches zero. It was able to do this, because it managed to successfully capture our zero-flow condition in regions where h reaches below the critical threshold. This part of our wetting-drying strategy is described in detail in Section 4.2.2. Figure 6.26 presents exactly the same results as discussed in Section 6.3.1, only under the collective label **Hydro-PINN-Static**. When we compare this to Figure 6.27, we can see that the **Hydro-PINN-Hybrid** captures this zero-flow condition in dry regions even more sharply in stages 4 and 5. On almost all ridges between the channels in those landscapes does that PINN predict zero flow velocity, where this is not always the case for **Hydro-PINN-Static** in Figure 6.26. This observation can form an explanation for one of our earlier observations, namely where the **Hydro-PINN-Hybrid** is less capable of predicting dryness in the water layer h than **Hydro-PINN-Static**. Dry areas generally appear in regions of the domain where the bed elevation rises above the water level and this mostly occurs on the ridges between the channels. By posing a zero-flow constraint in these regions, we directly restrict the network from evacuating any remaining water from those regions towards lower lying regions. A thin water layer will remain, locally increasing the continuity loss. The observation that **Hydro-PINN-Hybrid** better predicts zero-flow in dry regions compared to **Hydro-PINN-Static** is therefore a part of the explanation why the continuity residual for the former outgrows that of the latter.

Turning to the boundary conditions, it can be seen that the solutions produced by the **Hydro-PINN-Static** seem to adhere to the zero-gradient and zero-flow conditions at the closed boundary slightly better compared to **Hydro-PINN-Hybrid**, in Figures 6.26 and 6.27, respectively. Namely, the **Hydro-PINN-Hybrid** predicts non-zero flows at that boundary in stages 1, 2 and 3, where the **Hydro-PINN-Static** does not. In stages 4 and 5, the models produce seemingly equally well modelled closed boundaries. This is also reflected by the closed boundary residual $\mathcal{R}_{\text{bound,closed}}$ in Figure 6.29c, initially indicating that the **Hydro-PINN-Hybrid** performs worse than **Hydro-PINN-Static** but start to produce similar loss values for later stages. Like in the evaluation in Section 6.3.1, the general increase in boundary loss is likely due to our simplified boundary conditions. Furthermore, the **Hydro-PINN-Hybrid** predicts water flowing into the domain through the open boundary in stages 4 and 5 and this is an inaccurate prediction given the physical properties of the domain. Water is continuously injected by the H_{in} term and has to drain from the tidal prism through the open boundary of the domain, as the sedimentary bed topography is sloped towards that boundary. The **Hydro-PINN-Static** predicts no such inflow at the open boundary, and these observations are accurately reflected by the open boundary residual $\mathcal{R}_{\text{bound,open}}$ in Figure 6.29d, which is consistently higher for the **Hydro-PINN-Hybrid** than for the **Hydro-PINN-Static**.

The predictions made by the **Hydro-PINN-Saltmarsh** are presented in Figure 6.28, where it can be seen that some of the structural aspects of the underlying landscape are captured by the water layer thickness h, with water gathering in channels and dry regions in between channels. However, the water seems to only gather in wider channels, leaving the smaller branches unrecognised by the solution. Furthermore, the solution in general shows more signs of noise and artifacts. From these observations, it is clear that the **Hydro-PINN-Saltmarsh** produces qualitatively less accurate solutions for the water layer h in comparison to the **Hydro-PINN-Static** and the **Hydro-PINN-Hybrid**. The continuity residual $\mathcal{R}_h$ supports this qualitative assessment and reports larger relative losses for the **Hydro-PINN-Saltmarsh** in comparison to both other models. Only for some landscapes in stage 1 does this residual indicate that the **Hydro-PINN-Hybrid** performs worse, but this logically follows from the qualitatively extremely inaccurate solution produced by that PINN for that stage in Figure 6.27.

In contrast to the water layer h, the flow velocities u and v predicted by the **Hydro-PINN-Saltmarsh** in Figure 6.28 respond to the underlying landscape structuring much less accurately. In stages 1 and 2, these solutions are generally not a suitable fit to the landscape, with strong flows in unnatural directions and that do not follow the underlying bed topography. From stage 3 onwards, the channelisation is reflected slightly more accurately. However, large amounts of water seem to be flowing into the domain through the open boundary and following the channels land-inwards. This cannot be an accurate solution given the conditions. From Figure 6.29b, it is clear that the momentum solutions produced by the **Hydro-PINN-Saltmarsh** are indeed notably less accurate in comparison to both other models. The wrongfully modelled open boundary is, furthermore, clearly recognised by the open boundary residual $\mathcal{R}_{\text{bound,open}}$ in Figure 6.29d, as it indicates much larger loss values for this PINN in comparison to the other models. The elevated levels of noise in all components h, u and v are likely part of the reason why the closed boundary residual reports higher loss values for this PINN in those stages in Figure 6.29c in comparison to the other models. From stage 3 onwards, that residual indicates slightly improved performance with respect to the closed boundary constraints, and this is likely due to the drastically weaker flows in the areas surrounding the closed boundaries in those stages.

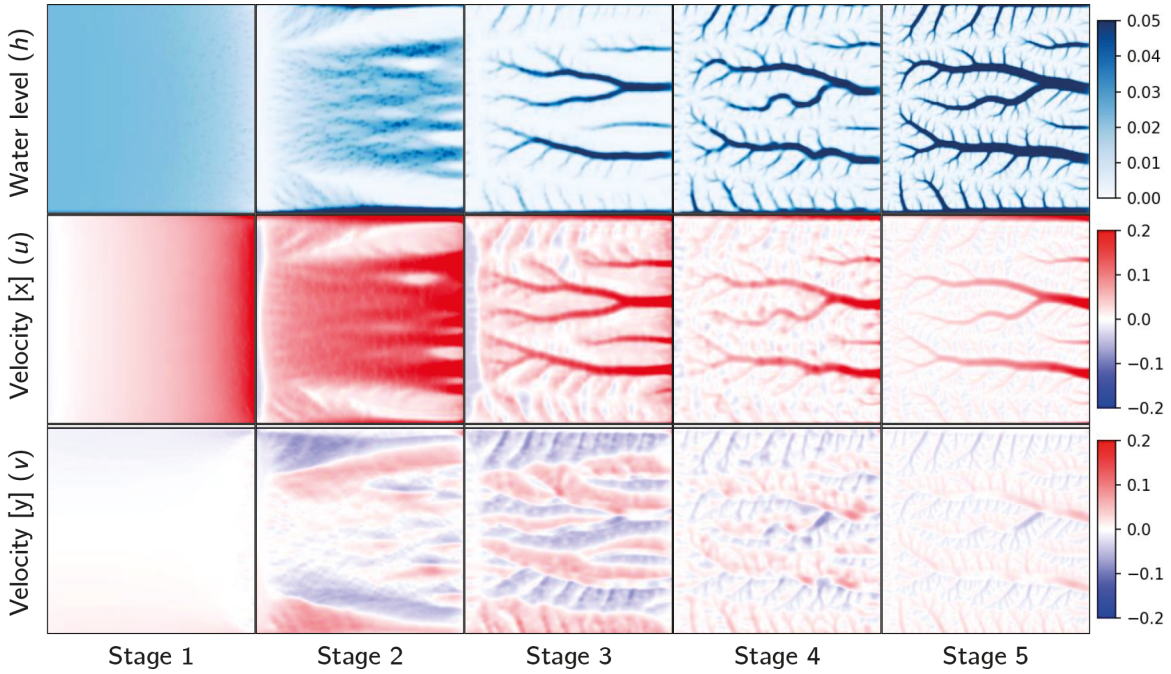


Figure 6.25: Hydrodynamic solution predicted by SFERE [66] for each category of landscape coarsening defined in Figure 4.5.

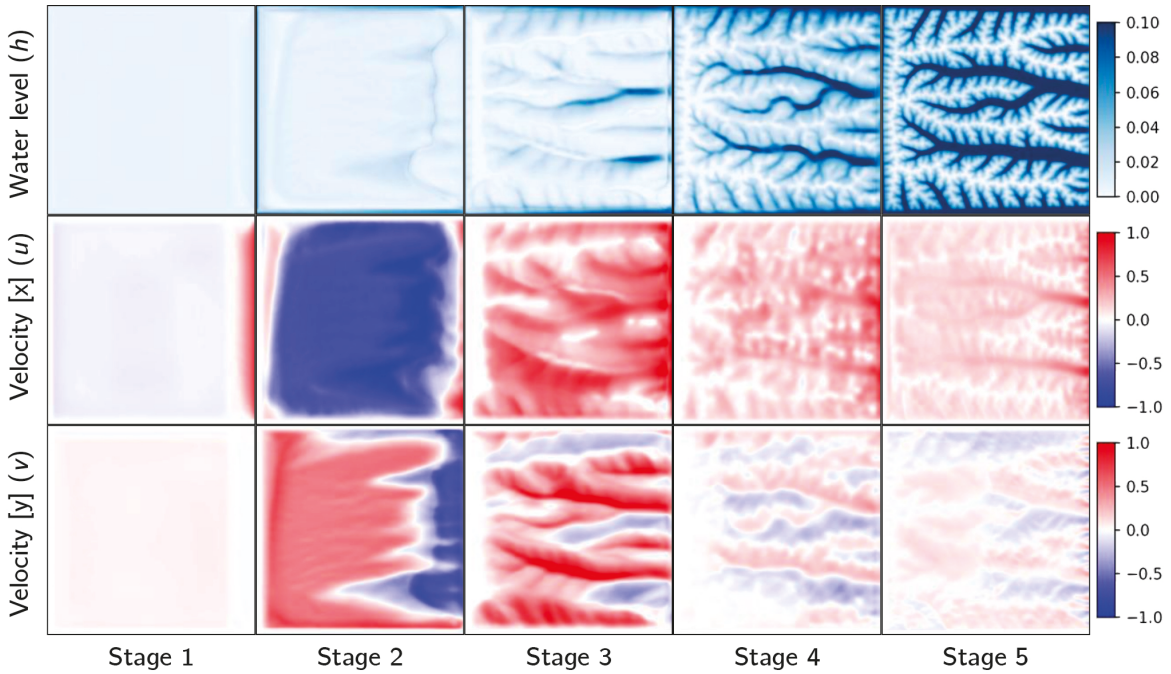


Figure 6.26: Hydrodynamic solution predicted by **Hydro-PINN-Static** for each category of landscape coarsening defined in Figure 4.5.

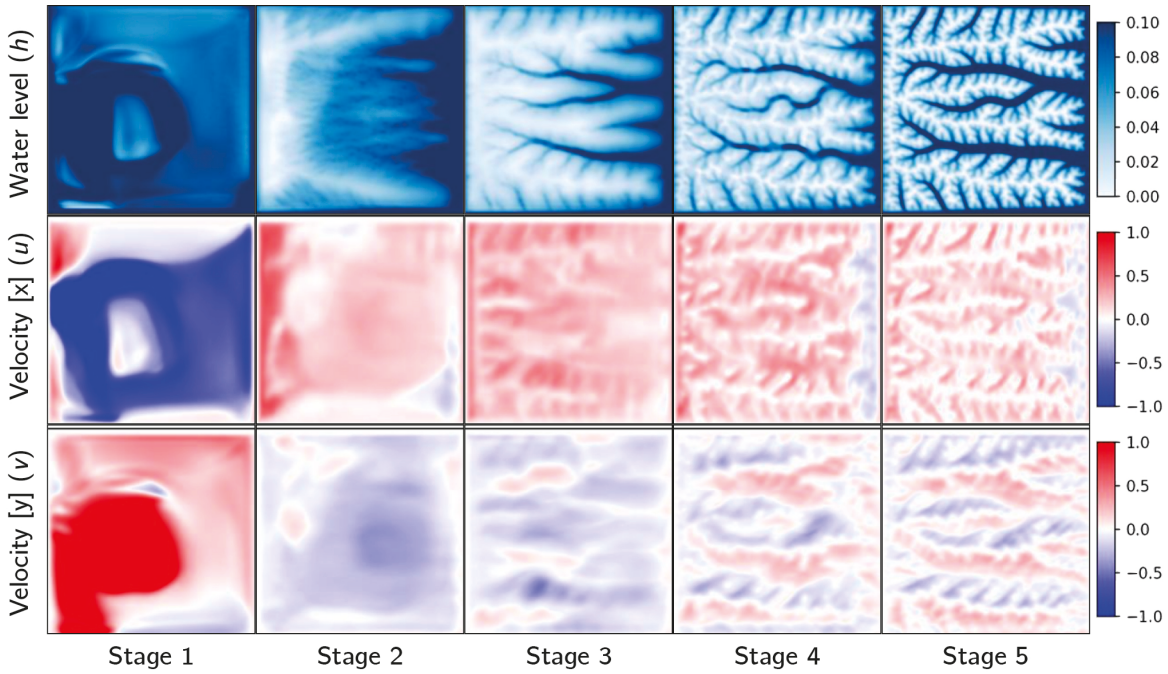


Figure 6.27: Hydrodynamic solution predicted by **Hydro-PINN-Hybrid** for each category of landscape coarsening defined in Figure 4.5.

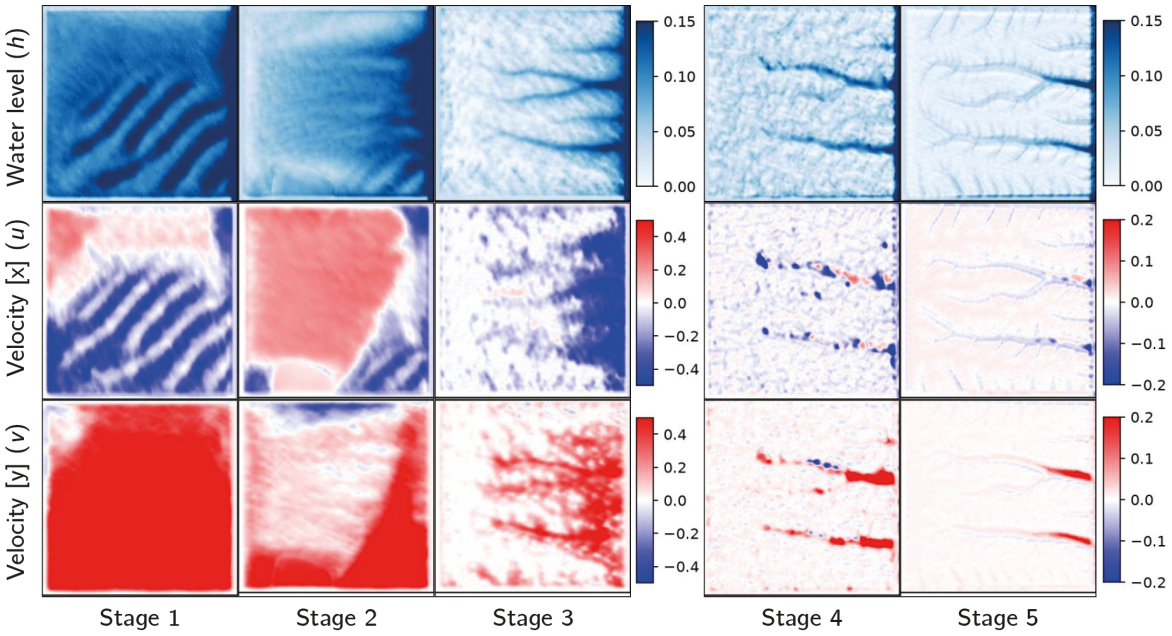
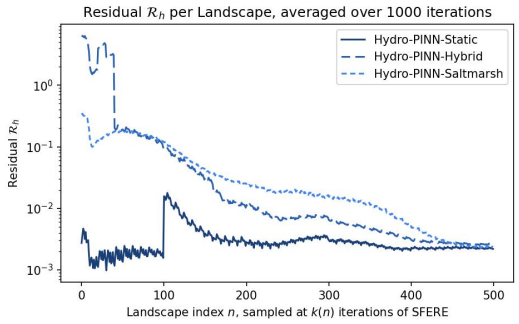
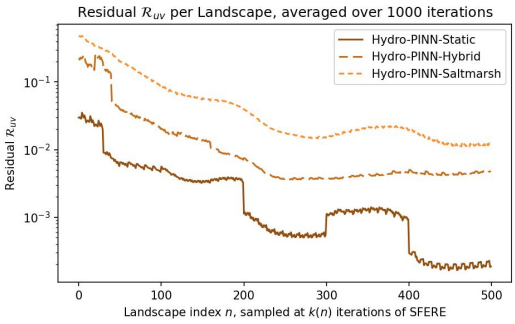


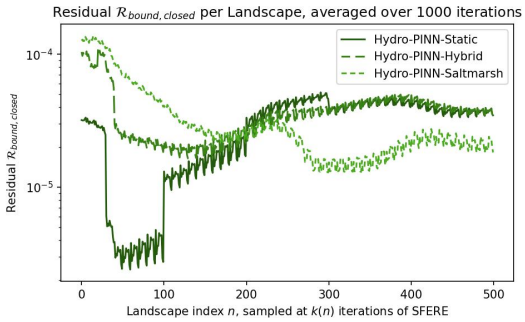
Figure 6.28: Hydrodynamic solution predicted by **Hydro-PINN-Saltmarsh** for each category of landscape coarsening defined in Figure 4.5.



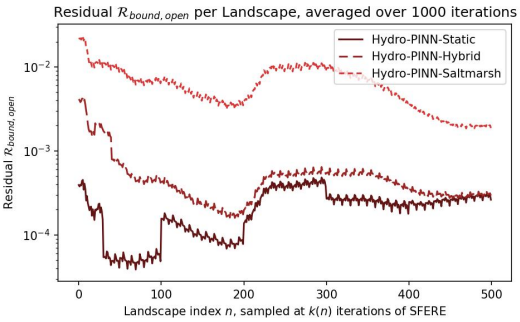
(a) PDE residual for the average continuity loss for each landscape.



(b) Combined PDE residual for the average flow velocity losses for each landscape.



(c) Boundary residual for the average closed boundary loss for each landscape.



(d) Boundary residual for the average open boundary loss for each landscape.

Figure 6.29: Non-dimensionalised residuals for each of the PDE and boundary loss terms, evaluated for all landscapes in the training sets and averaged over 1000 iterations. Residuals are presented for **Hydro-PINN-Static**, **Hydro-PINN-Hybrid** and **Hydro-PINN-Saltmarsh**.

The purpose for the development of the Hydro-PINN was to embed the SWEs, coupled to an underlying bed topography and vegetation in the landscape. The first objective was therefore to expand on the SWE-PINN established in Section 4.1 to allow coupling of the SWEs to an underlying layer of sediment and vegetation, and to model the connection of the landscape to the main tidal channel. This objective was formalised as sub-question **RQ2a**, for which we can formulate the following answer. By assuming that ebb-flow is the dominant contributing factor contributing to the self organisation in coastal wetlands, it is no longer needed to model the complete rising and falling of the tides and this can help to prevent large areas of dryness in the domain. By constantly injecting water in the domain, water can continuously flow through the channels in the landscape towards the connecting boundary with the main tidal channel. This connection can thus be modelled as an open outflow boundary, for which no loss term has to be used during training. Although the water level cannot be fully restricted to purely positive values, by compensating for negative water surface levels in the first spline order we can prevent the establishment of deep underground 'wells'. Scaling of variables before and after passing through the network is important, as large mismatches in scale can lead the network to over-react to larger signals.

For the training of this Hydro-PINN, the follwoing answer to sub-question **RQ2b** can be formulated. Training can be done by making use of the images produced by SFERE [66]. Via an Image Fitting CNN, that transforms pixel images to Hermite spline representation, these images can be used in the training set for this Spline-PINN. We have seen that farther coarsened landscapes can lead to higher training effectiveness, as they present the network with more complex spatial features. Larger spatial variance is beneficial for learning from a classical machine learning perspective. Furthermore, the increased effectiveness can be explained from the perspective of the PDEs themselves, as landscapes with more spatial complexity speak to the terms in the PDEs in more varying configurations and magnitudes. In our work, we captured the degree to which certain landscapes rely on certain terms in the PDE with the characteristics scales derived in Section 6.3.1. The Hydro-PINN can be placed in a hybrid setup, alongside a numerical solver for the morphological quantities, and this leads to spatially more consistent hydrodynamical solutions.

The purpose for the development of the Saltmarsh-PINN was to embed the hydro-morphodynamical system behind tidal channel formation in the framework of our PINN. Because the design for our Saltmarsh-PINN builds on that of the Hydro-PINN, sub-question **RQ3a** can be answered as follows. Because the Hydro-PINN is designed to be compatible with an underlying sedimentary bed topography and vegetation, it design considerations can almost directly be carried over to our Saltmarsh-PINN. Including our strategies for water management, wetting-drying, the modelling of an open boundary and the non-dimensionalisation of network in- and outputs. Two equations are added for the bed elevation and vegetation density, respectively, and they are represented by two additional physics-informed loss terms. The resulting system describes processes that operate on vastly different temporal and spatial scales, for which two important adaptations are necessary. Firstly, the time-scale difference between the hydrodynamics and the landscape processes can be alleviated by incorporating a morphological acceleration factor in the physics-informed loss. Secondly, the hydrodynamics, sedimentary and vegetative dynamics are spread among three separate PINNs, because the spatial-scale difference between the processes can be difficult to capture in a single U-Net. The Hydro-PINN is used to solve the hydrodynamics, and therefore takes a central position in the design of this PINN. Boundary conditions are implemented with separate masks for the x- and y-directions, which is different in comparison to the Hydro-PINN. During training, the three networks are treated as one, by wrapping their collective parameters into a single optimiser. The loss terms therefore do not have to be split among the networks and gradient surgery [77] can still be applied on all components of the system.

To train the three networks that form our Saltmarsh-PINN, the following answer to sub-question **RQ3b** can be formulated. The hydrodynamical component of the Saltmarsh-PINN can be effectively trained if paired with a properly functioning solver for the sedimentary and vegetative processes in the landscape. From our results, it is clear that training the three components simultaneously results in less accurate hydrodynamical solutions in comparison to the Hydro-PINN which was trained alongside an accurate numerical solver for the morphological quantities. Considering that the setup of the Saltmarsh-PINN, with a separate neural network for each process, facilitates asymmetric learning and multi-phase learning, there are training routines imaginable where this requirement can be (partially) fulfilled. For example, by first training the hydrodynamical component alongside a numerical solver for the other processes for a certain number of epochs, before inserting the remaining networks into the optimisation process. Whether the sedimentary and vegetative components can also be effectively trained with a properly functioning hydrodynamical component remains to be answered.

6.4. Physics Informed Morphodynamics in Coastal Wetlands

The previous section focussed on the hydrodynamical solutions, produced by our PINNs upon evaluating them on a series of static landscapes (i.e. with fixed sediment and vegetation). In this section, the focus shifts to a dynamical assessment of the complete hydro-morphodynamics predicted by our PINNs. Specifically, a qualitative assessment is performed on two hydro-morphodynamical models. In Section 6.4.1, the Hydro-PINN from Section 4.2 is evaluated in conjunction with a numerical solver for the morphological quantities, to investigate whether the physics-informed hydrodynamics are sufficiently accurate to trigger self-organisation in the landscape. In Section 6.4.2, the numerical solver is removed and both the hydro- and morpho-dynamics are solved by our Saltmarsh-PINN, which is described in detail in Section 4.3.

6.4.1. Hybrid Morphodynamics

The Hydro-PINN is developed in Section 4.2 to solve the SWEs, coupled to the sedimentary bed topography and vegetation in the landscape. In Section 4.2.9, it is explained how the Hydro-PINN can be placed alongside a numerical processor to solve the sedimentary and vegetative dynamics, both during training and evaluation of the network. The evaluation in Section 6.3.2 focussed on the hydrodynamics that resulted from training our Hydro-PINN in this configuration. This section takes the evaluation further by also considering the morphodynamics produced by this hybrid setup. Activating the numerical solver during evaluation allows us to investigate whether the physics-informed hydrodynamics are sufficiently accurate to trigger the scale-dependent feedbacks responsible for the spatial self-organisation of the landscape. The precise mechanisms behind this self-organisation are described in detail in Section 3.1.1.

Throughout this section, we look for visual characteristics in both the hydrodynamical solution produced by the Hydro-PINN, as well as the landscape structuring that the numerical solver produces in response to that. As the numerical solver processes the sedimentation and vegetation PDEs almost identically to SFERE [66], in this evaluation it is assumed to be sufficiently accurate. Three starting conditions are used for this evaluation. First, the system is evaluated on a flat bottom topography in an environment without vegetation. Then, the starting condition of SFERE is recreated, by randomly placing small tussocks of vegetation, to investigate how vegetation responds to the physics-informed water dynamics. Finally, the system is evaluated on a fully developed salt-marsh, to examine to what extent the PINN can sustain this converged self-organised state.

When there is no vegetation in the landscape, water immediately starts flowing towards the open boundary, where it drains from the tidal prism. As a result, a gradual slope starts to form evenly across the bed topography, as can be seen in Figure 6.30b, and this observation forms the first visual clue to signal an accurate (hydrodynamical) solution. In fact, the formation of a slope in the sediment is the first event that can be observed while running the SFERE [66] simulation and it is important that a slope forms, as it is a prerequisite for the scale-dependent feedbacks to be triggered. After 500 iterations, channel formation can be observed in Figure 6.30c, indicating that indeed a channelisation feedback was triggered in the sediment. This indicates that the physics-informed hydrodynamics are sufficiently accurate to support at least some form of self-organisation. However, the channels predicted by our model are much wider in comparison to SFERE [66], and have a lower degree of branching. This can for example be seen by comparing them to Figure 4.5b.

Across the earlier evaluations presented for this model in Section 6.3.2, we observed that our PINNs consistently predict much larger flow velocities in comparison to SFERE [66]. Likewise, the water levels predicted by our PINN are typically higher. These effects can offer an explanation for the widening of the channels, as well as for the lower branching rates that we observed. The spacing between parallel channels and the degree of channel branching are controlled by the ratio between channel incision and topographic diffusion rates [66]. This ratio is known as the landscape Péclet number [45, 46]. With an increased rate at which channels are incised, relative to the rate of topographic smoothing or diffusion, the spacing between neighbouring channels decreases and their banks steepen, causing the incision of more channel branches, and thereby increasing the channel network complexity [66, 45, 46].

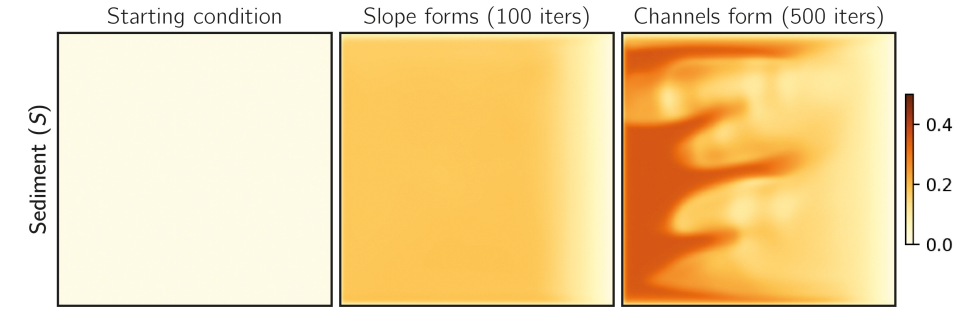
In SFERE [66], and therefore also in our work, the sedimentation rate increases with water depth, as was explained in Section 3.1.2. Higher overall water levels therefore lead to more deposition of sediment. At the same time, the bed friction term τ_b/ρ lowers with increasing depth, thereby reducing erosion rates. Combined, these result in a reduction of the channel incision rate, thereby lowering the Péclet number. The observed increased flow rates lead to higher bed friction and thus more erosion of sediment, but they also lead to more diffused flow, according to the turbulent mixing terms in Equations 3.2 and 3.3 with eddy viscosity D_U. In our results in Section 6.3.2, indeed we can see that the flow is less concentrated in comparison to SFERE [66], and this lowers the rate at which channels are incised. In total, the deeper water levels and stronger flows predicted by our models result in a net lower Péclet number, and can thereby explain the wider channel separation and lowered branching rate that we observed in Figure 6.30.

If random tussocks of vegetation are placed in the landscape, to recreate the starting condition of SFERE [66], a similar slope formation can be observed in Figure 6.31b. In that figure, the vegetation tussocks have grown to larger patches and the bed topography seems to respond to these patches with what seem to be trails behind the vegetation, following the direction of the flow. After 400 iterations, in Figure 6.31c, channels start to develop in the bed topography, with steeper banks and sharper ridges between the channels in comparison to Figure 6.30c. Recall from Section 3.1.1 that, in this context, vegetation refers to sediment-stabilising organisms, such as biofilms [76], algal mats [10] or plants [62, 60] and that interact with the sedimentary channelisation feedback to increase the steepness of banks and increase the overall network complexity [76, 62, 31, 40, 68]. Although the vegetation in Figure 6.31c is relatively sparse, these effects can still result in more concentrated flows and this can be an explanation for the steeper banks and sharper ridges that we observe.

As the simulation continues, vegetation patches start to grow in size, but they seem to grow faster in some directions than in others, resulting in elongated stretches of vegetation instead of evenly distributed patches, as can be seen in Figure 6.31c. After 800 iterations, Figure 6.31d even shows a slightly more complex patterning. Figure 6.31e provides the hydrodynamic solution corresponding to that landscape, where both the water layer and horizontal flow velocity indicate that waves have started to form in the domain. This can be seen by the darker vertical bands spanning in between the channel ridges in the water layer, and the corresponding alternating direction of horizontal flow in those regions. Because these fluctuating flows interrupt the constant outwards stream of water, these waves may drastically impact the self-organising feedbacks. It is not exactly clear why precisely this patterning appears or how it is formed, however it is likely that, among other things, the aforementioned effects have impacted the scale-dependent feedbacks sufficiently to break the multi-scale patterning. Furthermore, it seems as if there is no connection between the vegetation and the bed topography that can directly explain the observed patterning, as the patches of vegetation seem to ignore the ridges in Figure 6.31f. For the purposes of this work, it suffices to derive from these results that the observed vegetation patterning is unexpected given the landscape conditions and that this leads us to conclude that our PINN produces hydrodynamic solutions that are not yet sufficiently accurate to support the self-organising feedbacks with enough detail.

As a final test, the hybrid Hydro-PINN is evaluated on a fully developed salt-marsh landscape. The purpose of this test is to qualitatively determine to what extent our hybrid model can sustain the complex patterning that is present in the starting condition. In Figure 6.32, it can be seen that the complex channel network quickly fades over a period of a few hundred iterations. Already after 100 iterations, the branching network is reduced to a series of topographic ridges in the landscape, as can be seen in Figure 6.32b. In that figure, we can also see that the patterning in the vegetation gradually morphs from a multi-scale branching pattern to a regular string pattern, with bands of dense vegetation oriented perpendicular to the slope. This process continues, and after about 800 iterations, all vegetation is organised in banded patterns throughout the landscape, as in Figure 6.32c.

This type of regular patterning is known to appear in wetland ecosystems under specific circumstances [54]. The type of patterning that forms is influenced by numerous factors, such as the angle of the slope or the water input rate H_{in}, and it is plausible that our solutions contain different values for such parameters, induced by small inaccuracies, resulting in the regular patterning that we observe. The observation that some form of patterning appears can indicate that some aspects of our solution are accurate, and the observation that the patterning is regular, where branching channels were expected,

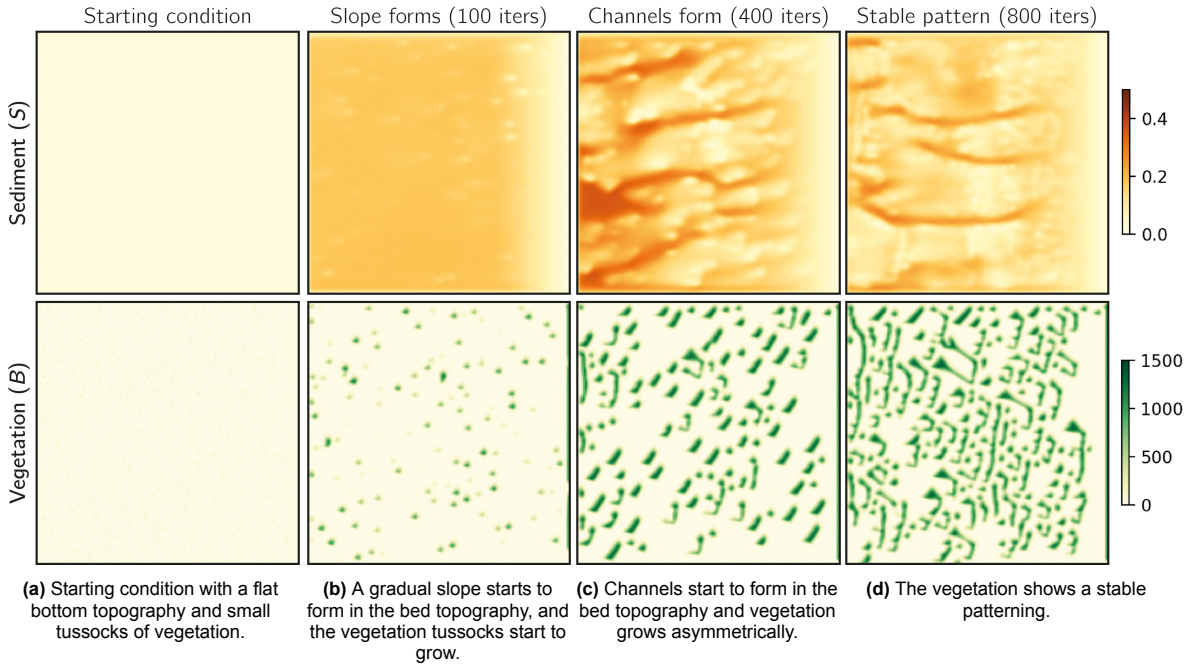


(a) Starting condition with flat bottom topography and no vegetation.

(b) A gradual slope starts to form in the bed topography.

(c) Channels start to form in the bed topography.

Figure 6.30: Sedimentary bed elevation S as predicted by the Hydro-PINN on a starting condition with flat bottom topography and no vegetation.

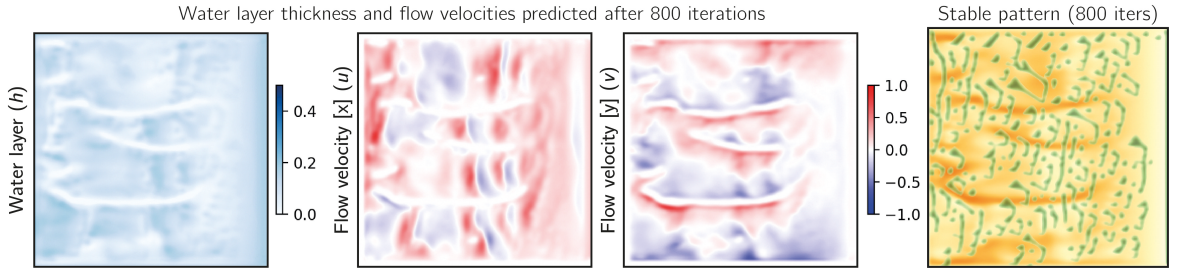


(a) Starting condition with a flat bottom topography and small tussocks of vegetation.

(b) A gradual slope starts to form in the bed topography, and the vegetation tussocks start to grow.

(c) Channels start to form in the bed topography and vegetation grows asymmetrically.

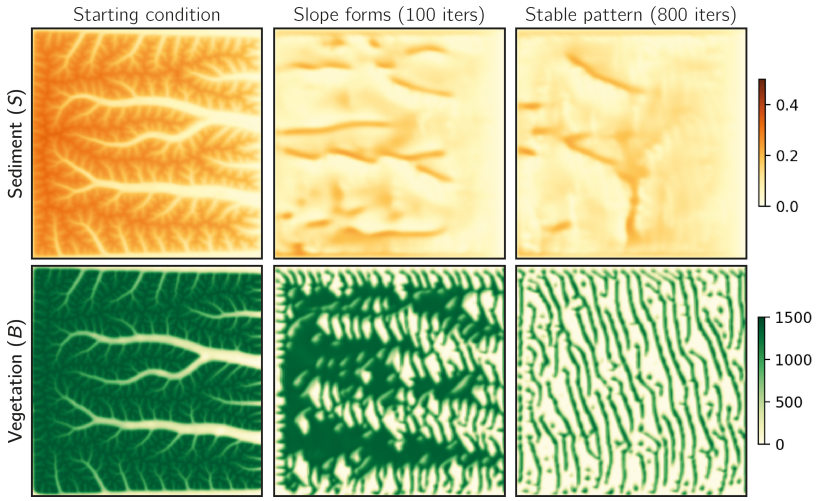
(d) The vegetation shows a stable patterning.



(e) Hydrodynamic solution corresponding to Figure 6.31d.

(f) Vegetation presented in overlay above bed topography, both corresponding to Figure 6.31d.

Figure 6.31: Hydro-morphodynamics as predicted by the Hydro-PINN when evaluated on the starting condition of SFERE [66].



(a) Starting condition with fully formed channel network.

(b) Sharp ridges start to form and the vegetation morphs from a multi-scale pattern to a regular band pattern.

(c) Regular string patterns in the vegetation.

Figure 6.32: Morphodynamics predicted by the Hydro-PINN when evaluated on a starting condition with a converged branching channel network.

indicates that some aspects of our solution are incorrect. Similar to the conclusions derived above, for the purposes of this work it suffices to conclude that the physics-informed hydrodynamics are not yet sufficiently accurate to represent the full array of scale-dependent feedbacks that are needed for the multi-scale pattern formation that would be expected given the conditions.

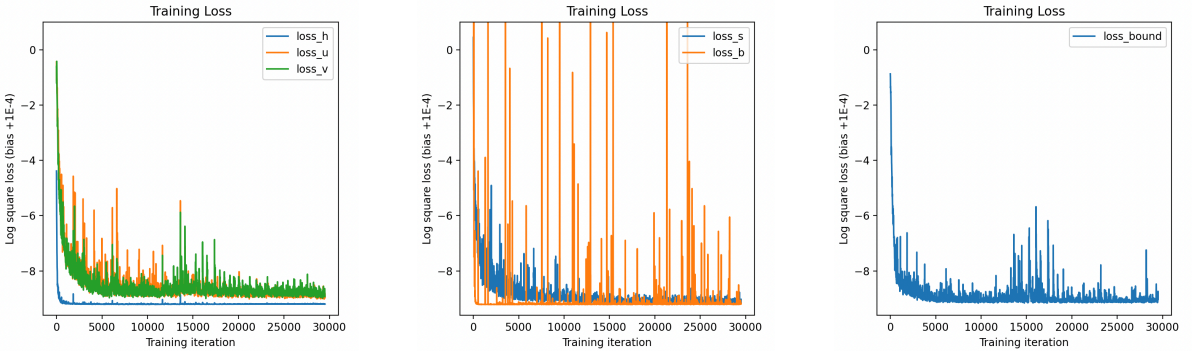
The purpose of evaluating the hydro-morphodynamical solutions produced by this hybrid configuration was to explore whether our Hydro-PINN produces sufficiently accurate hydrodynamics to support the self-organising scale-dependent feedbacks in the landscape. The results presented in this section contain several visual clues to indicate that some aspects of the solutions are adequate, given the starting condition. A gradual slope appears in the landscape, allowing water to smoothly flow towards the open boundary and trigger a channelisation feedback in the sediment. However, certain characteristics indicate that our physics-informed hydrodynamical solutions prevent the landscape from forming the precise patterning that would be expected given the conditions. The channels predicted by our model are considerably wider, and there are no signs that the feedbacks are triggered recursively to form branching channels. We explained that this is likely caused by overall higher water levels, as well as flow magnitudes in the landscape. With identical starting conditions, SFERE [66] predicts narrower channels, as well as multiple branching scales. These considerations combined form an answer to sub-question **RQ2c**.

In Section 6.3.2, the Hydro-PINN was evaluated on static landscapes after training in the hybrid setup. In that section, an answer is formulated to sub-question **RQ2b**, stating that the Hybrid-PINN can be placed alongside a numerical solver for the morphological quantities and that this can help to produce more spatially consistent solutions, but limiting the ability of the network to distinguish between various flow speeds. The insights that we obtained in this section can help to understand why. Therefore, an extension to the answer of sub-question **RQ2b** can be formulated as follows. By placing the Hydro-PINN in conjunction with a numerical morphological solver during training, the hydrodynamics predicted by the PINN directly influence the spatial structuring of the landscapes in the training set. The maximum characteristic scale of the training landscapes is thus limited by the accuracy of the hydrodynamic predictions, and this can in turn limit the training effectiveness of the PINN. In Section 6.3.1, it became clear that more spatial complexity in the training landscapes leads to higher training effectiveness. This, in combination with the lowered spatial complexity of the landscapes that result from training in this hybrid setting, can explain the lowered training effectiveness of the Hydro-PINN in this hybrid setup.

All combined, the answer to the second main question **RQ2** can be formulated as follows. After significant changes to the model and training procedures established in **RQ1**, the resulting Hydro-PINN can be trained on a set of virtual (salt-marsh) landscapes. These images can for example be produced by a numerical simulation, and with higher spatial variance in the training set comes higher training effectiveness. By coupling the Hydro-PINN to a numerical morphological solver, the landscapes can be updated according to the hydrodynamics predicted by the PINN during training. There, the training effectiveness of the PINN is limited by the maximum characteristic scale of the landscapes that results from its own hydrodynamic predictions. In general, the hydrodynamical solutions are accurate enough to trigger a channelisation feedback in the sedimentary bed, however they are not yet accurate enough to trigger all scale-dependent feedbacks responsible for the formation of branching multi-scale channel networks in the landscape.

6.4.2. Saltmarsh-PINN Morphodynamics

The Saltmarsh-PINN is designed in Section 4.3 to solve the system of hydro-morphodynamical equations with an ensemble of three PINNs. The SWEs are solved by an almost identical version of the Hydro-PINN that was evaluated in Section 6.4.1, and the sedimentary and vegetative equations are solved by separate PINNs. Section 4.3.5 explains how the PINNs are trained simultaneously on a series of starting conditions from SFERE [66]. After training in this configuration, the hydrodynamic component of this Saltmarsh-PINN has been evaluated in isolation in Section 6.3.2, and this section extends that evaluation by considering the complete hydro-morphodynamical system. To assess the extent to which the Saltmarsh-PINN can solve this complex multi-scale system of PDEs, this section presents a qualitative assessment of its hydro-morphodynamical solutions on two different starting conditions.



(a) Hydrodynamical loss terms L_h, L_u and L_v

(b) Morphological loss terms L_S and L_B

(c) Shared boundary loss term L_{bound}

Figure 6.33: Training loss of hydrological PDE loss terms (left), morphological PDE loss terms (centre) and boundary loss (right), for the Saltmarsh-PINN trained using the ImFit-CNN on 500 samples of SFERE.

Training of the Saltmarsh-PINN took 36 hours on an NVIDIA A40 and the training losses are indicated in separate figures in Figure 6.33. The hydrodynamical loss terms show fast convergence in Figure 6.33a. And similarly, the sedimentary loss term L_S and the boundary loss term L_{bound} seem to converge relatively quickly in Figures 6.33b and 6.33c, respectively. The vegetation loss term L_B, however, shows considerably more noise during training in Figure 6.33b in comparison to the other terms.

To start, the Saltmarsh-PINN is evaluated on a topographically relatively simple landscape, corresponding to the first category of landscape coarsening in Figure 4.5a. This starting condition already contains a slight slope in the bed topography, in the direction of the open boundary on the right edge of the domain. In Figure 6.34, it can be seen that the PINN is able to maintain some parts of the slope during the first few hundred iterations, but that the topography quickly morphs into a seemingly chaotic state after 400 to 600 iterations. No channelisation seems to occur in the sediment, indicating that the accuracy of the solution is not sufficient to trigger the corresponding scale-dependent feedbacks. Water seems to gather in the topographic lowlands, with some parts of the domain even falling dry in regions of high elevation. In those dry regions, there seems to be no sign of noticeably unstable velocities, which can be an indication that the zero-flow dryness condition described in Section 4.2.2 also works in this Saltmarsh-PINN. In horizontal direction, the water flow seems to follow the topography correctly, with water flowing mostly in downhill direction. However, the flow in vertical direction is not as consistent, with large flows against the direction of the topographic slope. At the open boundary, water seems to be flowing domain-inwards. Considering that a long-stretched valley has formed in the soil near the open boundary, this may actually signal a correctly modelled open boundary, as its boundary condition simply prescribes free flow of water through the boundary. Not necessarily only outflow.

From the vegetation stem density in Figure 6.34, it can firstly be seen that the densities of vegetation are generally much smaller, with values not exceeding 50-100 m^{-2}, where SFERE [66] assumes a maximum carrying capacity of $k = 1500$ m^{-2}. Besides this, the vegetation forms a spotty pattern across the landscape and the spots mostly appear in areas of higher elevation and seem to be eroded away in areas with strong water flows. Spots are of approximately equal size and are evenly spaced, however some are less dense than others.

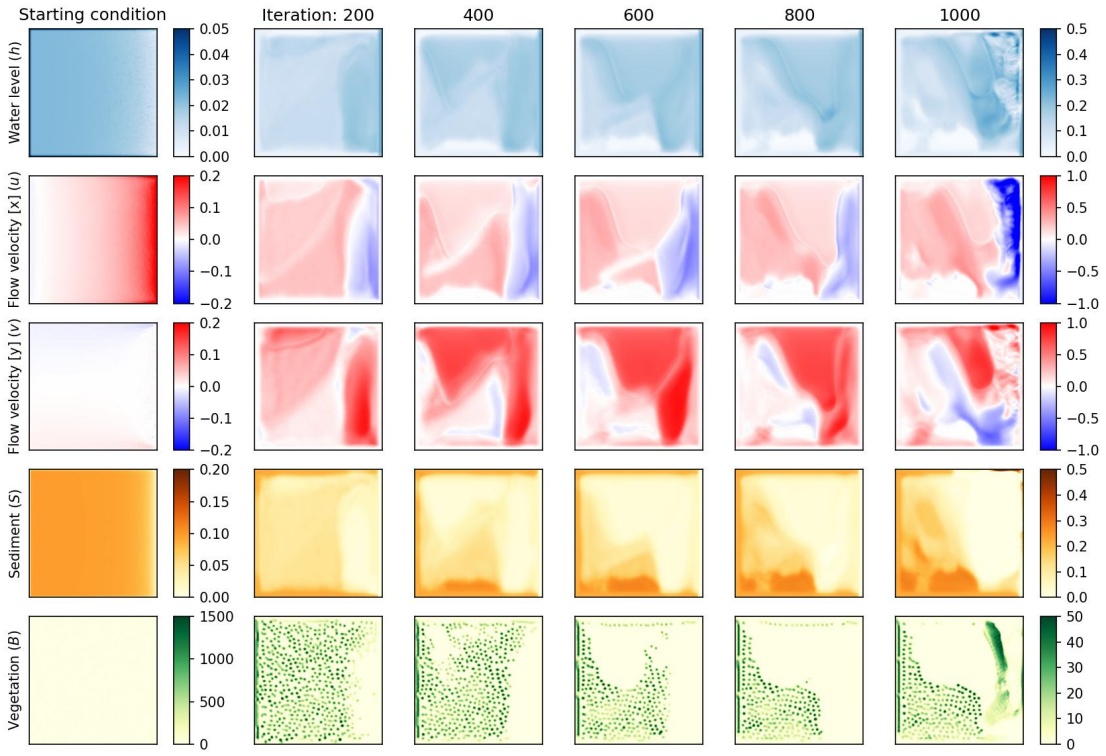


Figure 6.34: Hydro-morphodynamics as predicted by the Saltmarsh-PINN when evaluated on a starting condition corresponding to the first category of landscape coarsening defined in Figure 4.5a.

In Figure 6.35, the Saltmarsh-PINN is evaluated on the fifth and final category of landscape coarsening defined in Figure 4.5e, featuring a fully formed branching channel network. After 200-400 iterations, it can be seen that some aspects of the channel network are sustained by the PINN. However, the channels seem to be much wider and any sign of recursive branching has vanished from the bed topography. After 400 iterations, what remains of the channels quickly deforms into a seemingly chaotic bed topography, much like we saw after a few hundred iterations in Figure 6.34. Similar to that figure, water can be seen to clearly gather in the topographic valleys, with even larger areas of dryness appearing on highlands, in Figure 6.35. The momentum solutions show zero velocity in dry areas, indicating once again that our wetting-drying strategy from Section 4.2.2 can effectively prevent the velocity terms from turning unstable. Initially, the flow fields inside the domain show similar characteristics to Figure 6.34, with the horizontal flow accurately following the down-hill direction and the vertical flow indicating strong downwards flows in lowlands.

Similar to Figure 6.34, the predicted vegetation layer in Figure 6.35 indicates much lower overall vegetation densities in comparison to SFERE [66]. Furthermore, Figure 6.35 shows similar spotty vegetation patterns as in Figure 6.34. In addition to the spots of vegetation, however, this solution also seems to contain Turing-like patterns [64], that appear mostly in the top left of the domain. Although these stripes are seemingly regular, they are clearly different from the patterns that we observed in Figures 6.31d and 6.32c in Section 6.4.1. For the patterns in Figure 6.35, it is much harder to pinpoint what causes their formation. However, we believe that our similar theory for the regular pattern formations in Section 6.4.1 can also be applied here. From our results, it is clear that the overall water layer thickness is much higher in comparison to SFERE [66] and that the flow magnitudes are larger. Furthermore, we have seen in Section 6.3.2 that the **Hydro-PINN-Saltmarsh** produces hydrodynamic solutions that are considerably less accurate in comparison to both the **Hydro-PINN-Static** and **Hydro-PINN-Hybrid**. The patterns that we observe in the vegetation are therefore likely (indirectly) a result of both of these observations.

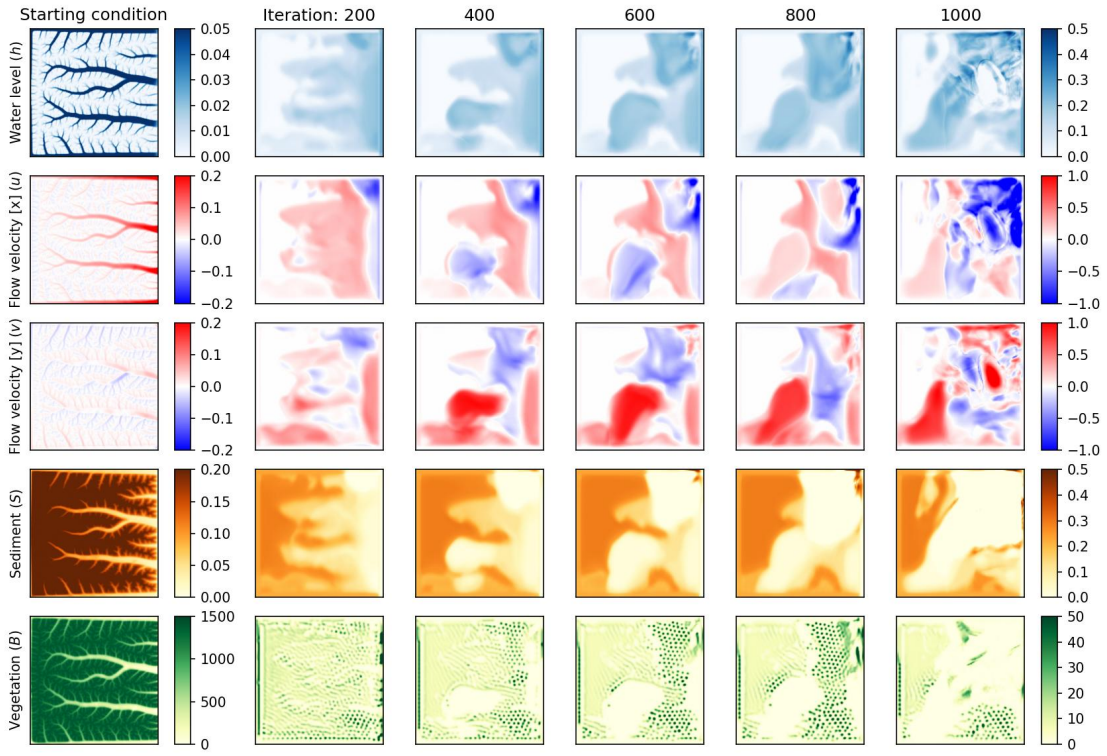


Figure 6.35: Hydro-morphodynamics as predicted by the Saltmarsh-PINN when evaluated on a starting condition corresponding to the last category of landscape coarsening defined in Figure 4.5e.

In total, these results show that not all five PDEs can be optimally solved by this Saltmarsh-PINN. During training, the network has trouble optimising for the vegetation loss term L_B and this is reflected in the simulation by poorly modelled vegetation dynamics. The PINN consistently produces vegetation patterns with much lower densities in comparison to SFERE [66]. Other than that, our PINN predicts mostly regular patterns in the vegetation. This may be an indication that the PINN has captured some self-organisation mechanisms in the landscape, however the patterns that form are drastically different from the reference solution in SFERE. Visual inspection of the hydro-morphological solutions produced by this PINN indicates some degree of physical accuracy. However, significant aspects of the solution are highly inaccurate, like water flowing against the general direction of the slope.

When specifically comparing the hydrodynamical component of this Saltmarsh-PINN to the Hydro-PINN that was trained and evaluated in earlier sections, it is clear that the former was trained less effectively compared to the latter. Given that the Hydro-PINN was trained in a hybrid configuration with a numerical solver for the morphological quantities, this difference in training effectiveness can be explained as follows. The hydrodynamic equations, and therefore the characteristic scale of each term in those PDEs, are tightly coupled to the underlying sedimentary bed topography and the vegetation stem density. It is through this coupling that the performance of one network influences the training efficiency of another. In Sections 6.3.2 and 6.4.1, it became clear how the training effectiveness of the Hydro-PINN is influenced by the spatial structuring of the landscapes in its training set. The same effect applies here. The training effectiveness of one component network heavily depends on the performance (i.e. the degree to which its solutions have been optimised with respect to the corresponding PDEs) of the others. Because the networks are here trained simultaneously, the resulting hydrodynamics are less accurate in comparison to the Hydro-PINN, as it was trained alongside a numerical solver which instantly produces accurate solutions.

With this insight, we can formulate an answer to sub-question **RQ3c**: The training effectiveness of the hydrodynamical component of the Saltmarsh-PINN is largely influenced by the type of landscapes that are used during training. We saw this during the ablation study in Section 6.3.1. We also saw that training of the Hydro-PINN can be more effective if combined with a numerical solver that simultaneously updates the sedimentary and vegetative profiles in the landscape, according to the hydrodynamics predicted by the network. A properly functioning solver for the sedimentary and vegetative processes in the landscape can therefore positively impact the training productivity of the hydrodynamical component of the Saltmarsh-PINN. However, with underdeveloped solvers for these processes, such as the U-Nets that are part of this Saltmarsh-PINN, the training productivity may be negatively impacted, like we saw in our results. The degree to which simultaneous training of the three networks can facilitate effective training in each individual network therefore largely depends on the (complexity of the) landscapes that the system produces.

Everything taken into consideration, the answer to the third and final research question **RQ3** can be formulated as follows. By simultaneously training three separate PINNs for each landscape process in the system, and by using a series of landscape-profiles created by SFERE as starting conditions, the training effectiveness of the individual networks is largely influenced by the predictive capabilities of the surrounding ones. Specifically in our experiments, it became clear that the hydrodynamical component performed noticeably worse when trained alongside two PINNs for the morphological equations compared to a numerical solver. The vegetative component seemed to have the most trouble finding a suitable optimisation path. The solutions produced by our Saltmarsh-PINN instance, in which all three networks were trained simultaneously, are inadequate with respect to a number of visual characteristics. However, some aspects of the simulation seem to be accurate and could be explained by simple physical reasoning. Examples are the formation of single-scale patterns in the vegetation, drastic decay of vegetation in areas with high levels of water, and the accumulating of water in deeper valleys in the bed topography.

7

Discussion

This thesis focussed on the embedding of a complex multi-scale hydro-morphodynamical system for the formation of tidal channel networks in a physics-informed learning framework. For this, we built on the frameworks of Spline-PINN [73] and SFERE [66] to design three physics-informed models. The SWE-PINN, described in Section 4.1, sets the stage by embedding the SWEs in a closed water basin with no bed topography. The Hydro-PINN, described in Section 4.2 extends this conceptualisation by coupling the SWEs to sedimentary bed topography and vegetation and opening the connection between the landscape and the main tidal channel. Finally, the Saltmarsh-PINN, developed in Section 4.3 captures the complete hydro-morphodynamical system by spreading the work across three separate PINNs.

In a closed water basin, our SWE-PINN produces fast wave simulations that accurately predict many of the important wave phenomena, such as reflections and interference patterns. The waves are created by oscillators that can be positioned anywhere in the domain, and their surface disturbances are the main driving force in the training process. The use of gradient surgery [77] during optimisation improves the training effectiveness by avoiding conflicts between the PDE and boundary objectives. Without it, the boundary conditions are heavily under-prioritised, resulting in waves permeating boundaries and overall lower training efficiency. Our model generalises towards domains unseen during training, including ones with complex boundary- and obstacle-shapes. We can also simulate wave frequencies and amplitudes other than those used during training.

In a partially open coastal wetland the underlying sedimentary and vegetative layers become the drivers of training. By training on images generated by SFERE [66], it became clear that landscapes with more complex spatial features can lead to more effective training of the Hydro-PINN. Our Image Fitting CNN, described in Section 4.2.7, facilitates the use of pixel images (e.g. those from SFERE) as starting conditions and it allows the PINN to be trained in conjunction with a numerical solver for the morphological quantities. This numerical solver updates the sedimentary bed topography and vegetation densities inside the training landscapes, to present the network with a much larger pool of training conditions. Training in this hybrid configuration results in spatially more consistent solutions that are sufficiently accurate to trigger a channelisation feedback in the sediment. However, because our PINN consistently predicts larger flow magnitudes and higher water levels, in comparison to SFERE, the resulting channels are much wider and the degree of branching is lower, resulting in landscapes with less spatial complexity. Because these landscapes are recycled during training, this reduction in spatial complexity eventually spreads across the entire training set and this negatively impacts the training effectiveness of the PINN. In landscapes with vegetation, our PINN predicts regular string patterns instead of the multi-scale branching channel networks and this is likely also a result of the changed conditions that result from the limited accuracy of our hydrodynamic predictions.

The Saltmarsh-PINN essentially replaces the numerical morphological solver with two PINNs for the sedimentary bed elevation and vegetation density and this negatively impacted the training effectiveness of the Hydro-PINN. This indicated that the simulation performance of one network in the Saltmarsh-PINN can directly affect the training effectiveness of another. Our results indicate that the morpholog-

ical acceleration factor used in many numerical models can be effectively embedded in the physics-informed loss, but that the spatial scale difference between the processes is harder to capture. The hydro-morphodynamical solutions produced by this Saltmarsh-PINN are not sufficiently accurate to trigger the scale-dependent feedbacks necessary for self-organisation in the landscape. Some form of regular patterning could be observed in the vegetation, however we refrain from deriving any strong conclusions from these observations.

7.1. Implications for Coastal Wetland Modelling

This section covers the most important implications that our findings can (potentially) have for the field of coastal wetland modelling. Specifically, these implications can be relevant for works like SFERE [66] and Delft3D [11, 12], but also for other hydro-morphodynamical systems. Our physics-informed models have several properties that could potentially translate to useful improvements for these existing coastal wetland simulations. Our PINNs have demonstrated to be capable of solving the SWEs at a time-step that is orders of magnitude larger than a typical numerical simulator. Our simulations advance at a rate of one second per iteration and, combined with the fast inference of CNNs, this could improve the hydrodynamical simulation speed significantly. Besides the speed improvement alone, this faster pace can help narrow the time-scale gap between the hydro- and morphodynamical processes in the simulation, which may allow them to work with smaller morphological acceleration factors for improved stability and accuracy.

Many numerical simulations rely on simplifications of the model, to make the simulations more stable and more easily computable. SFERE [66], for example, neglects the lateral advective sediment transport in the modelling of the sedimentary bed elevation. As a result, the meandering of rivers is not captured and the simulated channels are mostly straight. By accelerating the hydrodynamics and narrowing the time-scale gap between the landscape processes, these physics-informed models may not require as many extensive simplifications and could therefore lead to more accurate representations of the physical system. Concretely in what areas we could potentially remove such simplifications requires further research.

Besides this, because our simulations make use of Hermite Splines for their mesh-less representation, our solutions contain an implicit damping factor that may increase the stability of the solutions. Any type of shock or sudden gradients are immediately washed away, because the Hermite splines cannot capture them fully. The degree to which sudden gradients can be captured by the splines depends on the chosen spline order, and precision can therefore be tuned. Larger spline orders may require more collocation points during training, however, which can lead to longer training times. Such damping can affect the accuracy of the hydrodynamic solutions, for which a trade-off emerges between accuracy and longer training times for higher spline orders.

For these hydro-morphodynamical systems, the hydrodynamics are particularly limiting in the sense that they form the most sensitive component of the system. Water dynamics are relatively extremely volatile, making small time-steps and a morphological acceleration factor inevitable. In this thesis, we have designed a hybrid setup between a hydrodynamical PINN and a numerical morphological solver by making use of a CNN that transforms the pixel images produced by the numerical processor to spline latent space that can be processed by the PINN. Such a hybrid setup removes the need to embed all five equations in a PINN, so that many of the aforementioned potential benefits could already be achieved by only embedding the hydrodynamical equations. What is more, the numerical solver does not have to rely on Finite Differences to approximate gradients and can instead make use of the differentiable splines. Important to realise is that, in this hybrid configuration, also the morphological quantities are affected by the implicit damping that is caused by Hermite splines. Also, the accuracy of the Image Fitting CNN becomes relevant to the stability and accuracy of the final solutions. From the results obtained by our Saltmarsh-PINN, it is clear that embedding of the complete system of hydro-morphodynamical equations in a physics-informed setup is challenging. Therefore, further developments of our Hydro-PINN in combination with our hybrid design may be more beneficial for coastal wetland simulations on the short term.

It is important to mention that the multidisciplinary nature of many of the aforementioned potential applications may impact the degree to which specialists understand the simulations that they are working with. Numerical solvers have been extensively used and taught in almost every profession, making them a well understood principle. The behaviour of PINNs is much less widely understood and this could impact interpretation of the results by specialists in fields other than that of computer science. Gaining wide-spread understanding about the behaviour of these models is therefore crucial if we want to apply them widely.

7.2. Implications for Physics-Informed Modelling

Foundational to our work is Spline-PINN [73], which demonstrated that the ability of a PINN to produce continuous and differentiable solutions with little training data can be combined with the inference speeds and generalisation capabilities of CNNs, by making use of Hermite Splines. This allowed them to learn flexible-boundary solutions to the incompressible Navier Stokes equations and a damped wave equation, by only training on the PDEs. This thesis takes the approach behind Spline-PINN further, by embedding a more complex hydro-morphodynamical system consisting of five tightly coupled and non-linear PDEs. This results in more learning objectives and more boundary conditions that the network should capture in order to produce accurate solutions.

For an accurate solution to the system, it is important that the network optimises the solution with respect to each of the learning objectives equally. In general in machine learning, training a network to solve multiple objectives simultaneously can be challenging, for example because it requires careful weighing of terms. From the results of our SWE-PINN, it is clear that the PDE objectives had dominated the boundary task, thereby drastically impacting both the accuracy of the solution and the training effectiveness of the PINN itself. Our results further show that such domination can be effectively avoided with gradient surgery [77] to dynamically avoid conflicts between multiple learning objectives during training. This reduces the need for tedious tuning of balancing parameters.

For the framework of Spline-PINN [73], and for works based on Spline-PINN, this gradient surgery strategy [77] may allow learning of larger PDE systems without compromising boundary objectives. In comparison to the two-equation Navier Stokes and wave equation-systems solved by Spline-PINN, our Hydro-PINN could effectively learn to solve a system of three PDEs with strict conservation laws and with an open boundary. Thereby, our results demonstrate that this more complex system could also be learned without the use of training data from the field, by training directly on the PDEs and boundary conditions of the system. This could potentially set the stage for other large and complex PDE systems, where the speed and flexibility of CNN-based PINNs could offer useful benefits.

However, our Saltmarsh-PINN results show that training becomes notably more difficult when the PDE system becomes even more complex. The complete hydro-morphodynamical system behind SFERE [66] could not be captured by a single U-Net, because of the temporal and spatial scale-difference between the landscape processes. Therefore, we opted for a multi-network setup consisting of one hydrodynamical, one sedimentation and one vegetation PINN. Our results demonstrate that this multi-network configuration is unable to accurately capture the dynamics of the complete system if the networks are trained simultaneously. Based on our Hydro-PINN results, we reasoned that the hydrodynamics produced by the PINN can directly impact its own training effectiveness by altering the structuring of the training landscapes via the numerical solver. In our case, the large fluxes predicted by our PINN drastically lowered the spatial complexity of the bed topography and vegetation in the landscape. And the results from our ablation study show that such reduction in spatial complexity can negatively impact the training effectiveness. The Saltmarsh-PINN essentially replaces the numerical solver with a physics-informed alternative and our results demonstrate that this leads to even lower training effectiveness of the Hydro-PINN, likely caused by the under-optimised intermediate morphological simulations that appear at earlier training phases.

These results combined imply that one PINN in an ensemble of multiple solvers (not necessarily only PINNs) can impact its own training by producing solutions whose characteristics influence the predictions made by neighbouring solvers. Furthermore, they imply that simultaneous training of multiple networks in such an ensemble is ineffective, because of inaccurate intermediate solutions produced in early training. Both these implications reason about the impact of the *solution* of one network on the

ability of other networks to learn effectively. Therefore, these results further imply that this problem cannot be resolved by merging the dynamics into a single neural network. A single network, responsible for the solution of both the hydro- and morphodynamics will produce similarly poor solutions because the physical component-solutions influence the training effectiveness of the network with respect to neighbouring components. It is therefore, in this case, not about the fact that we use multiple networks to solve a combined problem, but about the fact that we try to optimise for the three landscape processes simultaneously.

The above reasoning can impact field of physics-informed modelling by implying, supported by derivations from our results, that simultaneous learning of all equations in a complex PDE system may not always be possible. The design of our Saltmarsh-PINN facilitates multi-phase or asynchronous learning, and it is our belief that such strategies are unavoidable when training a physics-informed model to solve such a large and complex system. Regardless of whether we use a single network or an ensemble of networks.

It is important to realise that the above reasoning is tightly connected to the training strategy that we opt for. In our setup, the solutions of one PINN only influence the training effectiveness of others, because we recycle the landscapes back into the training set after each iteration. Alternative strategies could be designed where this is not the case, and in such approaches it may still be possible to optimise for multiple processes simultaneously. For example, we could lower the rate at which landscapes are recycled and more frequently insert pre-rendered landscapes of which we can control how their spatial complexity influences training. We could also opt for a training strategy that uses both a numerical solver and a PINN for the morphodynamics simultaneously, so that some landscapes in the training set are rendered numerically (i.e. with controlled accuracy) and others by the PINN. Both these examples contain some form of asynchronous training, by not always letting the PINNs work on component solutions produced by neighbouring ones, and more research effort is required to derive precisely how this affects learning efficiency.

By separating the PDEs across multiple PINNs, as in our Saltmarsh-PINN, we limit the ability of the PINN to learn the exact coupling between the equations. In our architecture, the hydrodynamics only interface with the morphological quantities at the in- and output layers of the network and this obstructs the network from learning how the dynamics are coupled at deeper (hidden) layers. Alternative architecture can be imagined where the individual components are more connected across the different layers, while still supporting multi-phase or asynchronous training. For example, we could opt for a U-Net architecture with one encoder and multiple decoders, so that the encoder captures the coupling between the components of the system and the decoders learn how that influences the solution of each of the individual PDEs. Multi-phase and asynchronous learning are still supported by that architecture, as we can temporarily disable some of the decoders to focus on the learning of one of the system components. Temporarily freezing certain parts of the network is a technique that is also referred to as staged or modular training. Furthermore, they can be compatible with *curriculum learning* [44] for multi-task optimisation, where the network learns to solve one task before the other, possibly while freezing parts of the network that are responsible for the remaining tasks. In such multi-task networks, the difference in absolute scaling between the variables in the system may, for example, be alleviated by applying batch normalisation [25] to prevent internal layers from receiving inputs with varying value ranges throughout training.

7.3. Implications for Scientific Computing

This section outlines the possible implications that the findings in our work may bring to the more general field of scientific computing, for example to the field of scientific machine learning (Sci-ML) which combines traditional physics-based modelling with machine learning techniques. Many Sci-ML approaches rely on the availability of training data to build reliable models. By training on field data from ground measurements, cameras, satellite imagery, or historical data, these models try to make a prediction about future states of a system. Because these models are trained on data, their predictions can be hard to interpret because the trained model essentially forms a black box.

The physics-informed models presented in this thesis, including the original Spline-PINN [73], do not rely on training data from the field to make accurate predictions. By training directly on the equations of the system, the resulting predictions are more easily interpretable and we can reason about the accuracy of the solutions by referring back to those equations. Combined with the simulation speed and boundary flexibility of our models, this could potentially help us build better understanding about real-world systems. In such applications, our models could, for example, be applied as a digital twin, and could potentially even allow real-time interaction with the system using our fast inference speeds.

As was demonstrated by Spline-PINN [73], and also by our own SWE-PINN, the physics-informed models considered in this thesis have the ability to work with complex domain geometries by simply providing the network with one or more boundary masks. Such flexibility could be of added value for computing problems outside those considered in this thesis. It may allow geometries to be directly imported from, for example, satellite data and there is no need to re-discretise the grid representation. Because our networks accept images as boundary masks, and because they work with mesh-less solutions, the simulation speed is not affected by increased complexities of domain geometries. Our strategies may also support moving boundaries, or even allow scientists and engineers to interact with the boundary conditions in a real-time simulation.

As part of this thesis, we have demonstrated that a hybrid system can be created between a physics-informed model and a classical numerial solver by making use of the locality of Hermite Splines. A simple CNN is sufficient to transform numerical outputs to our mesh-less representations and this allows us to directly combine the two modelling techniques. In fact, the numerical simulator does not have to rely on Finite Differences to approximate gradients and can instead make use of our auto-differentiation capabilities. A core concept of Sci-ML is to embed physical knowledge in a machine learning context, and hybrid systems can be an important part of that concept. The results from our Hydro-PINN experiments could suggest that not all PDEs in the hydro-morphodynamical system behind SFERE [66] necessarily have to all be solved by a PINN. Similar trade-offs exist in surrounding fields of scientific computing, and our CNN-hybrid architecture based on Hermite Splines may potentially improve simulations in such fields as well.

As was mentioned earlier, a standing challenge in physics-informed modelling using the methods described in this thesis is that the solutions are challenging to quantitatively and accurately verify. Because the models are trained on the PDEs of the system, we can interpret and reason about the solutions, but much more rigorous metrics are needed to determine the accuracy of our solutions before they can be reliably used in many of the aforementioned applications. Within scientific machine learning, there may potentially be ways to make the outcomes of our predictions more reliable and quantifiable. The fast inference of our models could be useful in cases where many different simulations need to be run. For example to create accurate uncertainty quantifications, or to predict outcomes following a large set of parameter configurations. This may have use-cases in both research and engineering, but only if the accuracy of the predictions themselves can be measured.

7.4. Implications for Real World Challenges

This section explains how the work presented in this thesis (or extensions thereof) can potentially impact some relevant and important real-world challenges. Accurate and fast simulations of hydrodynamics in general form the cornerstone of many important systems and the safety of entire communities can depend on these systems. To name a few examples: The safety of millions of people can depend on our ability to make fast and accurate predictions about an incoming tsunami. These hydrodynamical simulations are challenging, because a tsunami wave brings various numerical challenges, some of which we have also seen in this thesis. For example, a suitable wetting-drying technique must be chosen, to simulate what path the tsunami takes across sections of land that were first dry. A second important example is related to predictions about sea-level rise and how it may impact our dikes, bridges and delta-works. Of course, the sea level rises at a much slower pace in comparison to an incoming tsunami, but fast and accurate hydrodynamical models are still needed, for example because the development or improvement of delta-works can take years. Other coastal events, such as storm-surges, rely on faster simulations and can have an equal impact on infrastructure. As a final example, fast and accurate shallow water simulations can be used for the engineering of marine vessels, as ships need to withstand powerful waves.

In each of these fields, management and engineering decisions are based on the simulations of a hydro(-morpho)dynamical solver. Engineers need to understand these simulations, and managers need to be able to interpret them, in order to make important decisions that can directly involve the safety of citizens. Expanding our proof of concept to a working PINN model for the simulation of two-dimensional hydrodynamics, with coupling to an underlying bed topography and possibly other factors, could impact these applications as well. For example, faster hydrodynamics could significantly reduce the time needed to construct a large-scale evacuation plan in case of a tsunami, and the type of wetting-drying strategy proposed by our models could alleviate some of the other challenges in tsunami-modelling.

Faster models could also facilitate real-time simulation, which would allow us to more carefully manage the opening and closing of important delta-works, for example. Spline-PINN [73] demonstrated that interactive boundary conditions can be implemented, by dragging an obstacle across the domain. Similar interactions could be applied in our models, making the simulations more interpretable. Furthermore, the flexible boundary shapes that can possibly be simulated with these PINNs, like in our SWE-PINN, could provide simulations of real-world conditions, so that the predictions could immediately translate to the given scenario.

As was mentioned before, all of these applications rely on extremely accurate and reliable hydro(-morpho)dynamical predictions, because they can result in executive decisions that impact hundreds or thousands of lives. The suggestion that our models could improve the simulations behind these decisions is therefore purely made from a motivational standpoint. By no means are our hydrodynamic solutions accurate enough to facilitate (engineering) decision making on these levels. Further research is needed to improve the accuracy of our models, as well as to design methods to verify that their predictions are reliable.

What is also important, is that the numerical solvers that are used for these applications nowadays are widely understood by engineers, managers, students and teachers. For these societal fields, this is important misinterpretations could lead to inappropriate decisions that could potentially impact many people. Before these physics-informed models can be widely used in those areas, we should try to look for ways to make the behaviour of these models more easily understandable for people.

7.5. Limitations and Recommendations

Several measures ensure the reliability and reproducibility of the methods and analysis presented in this work. Firstly, by building upon the works of Spline-PINN [71], we ensured that the design of our PINNs and their training strategies are founded on working and validated models. They present a well-tested and reproducible method to train a CNN based on Hermite Splines on a small system of PDEs. We ensured that the readability, understandability and reproducibility of the code remained despite scaling to a larger system of PDEs, and our strategy to achieve this has been explained in Section 5.2.1. Secondly, many design decisions regarding the physical hydro-morphodynamical system are based on SFERE [66], an accurate and widely used numerical simulator for the formation of tidal channels. By constantly referring back to SFERE, we ensured that our design choices are based on expected outcomes of the simulation (e.g. given a certain starting condition) and that our hypotheses and analysis are based on (comparisons with) a working and validated model. All our code is publicly available [15, 16] and is built to work on various GPU architectures for improved reproducibility. We used fixed seeds during both training and evaluation, ensuring identical outcomes across repeated runs. Finally, we used automated storage management and parameter logging to ensure that trained models are easily traceable and stored alongside the specific parameter values that were used during training.

The aforementioned are measures that we took to ensure that our findings are based on reliable methods that are verifiable. This does not mean that our methods can be expected to work under conditions other than those presented. In the following, we outline the boundaries of our research. An important limitation of (physics-informed) neural networks in general is that their predictions are difficult to exactly verify. Earlier in this section, we mentioned how the solutions are more easily interpretable because we solely train on the PDEs of the system, but this does not mean that the solutions are easily verifiable. To quantitatively assess how accurate the solutions are, given a certain starting condition and parameter configuration, is a much more complicated and is a standing problem with physics-

informed models. Because the predictions are difficult to quantitatively verify, we opted for a qualitative assessment that is supported by normalised residual values (i.e. non-dimensionalised loss terms) for improved interpretability of our results. These residuals do not directly form an accurate representation of the physical accuracy of the solutions, and they should also not be interpreted that way. The work presented in this thesis should therefore be interpreted as a conceptualisation or a proof-of-concept for the targeted models.

The simulations made by our models cannot be used in settings that depend on high accuracy of the predictions. As mentioned, the methods proposed in this thesis could form a new basis for hydrodynamical simulations in the future. However, if a working system in such context was to be replaced by one of our PINNs, reliable and testable metrics are a necessity to verify that their solutions can be safely and responsibly used for predictions in these areas. The metrics used to assess the quality and accuracy of solutions in this thesis are not sufficient in that regard. We recommend that research towards reliable verification metrics or PINNs is therefore continued, and specifically in the context of hydrodynamical and morphodynamical simulations.

The results presented in this thesis were obtained by training on a relatively small set of starting conditions, consisting of 500 landscapes generated by SFERE [66]. Training our models on larger datasets could improve the reliability of our conclusions, by training and testing the systems on more samples, and could also improve the training performance of the networks in general. However, a core principle of these physics-informed models is that they can be trained without the use of extensive training sets. For this reason, we refrained from generating more training landscapes and instead opted for a training strategy where the PINN would generate its own training data by using a (numerical) morphological solver for the landscape processes. This idea is highly inspired by Spline-PINN [73], where such recycling of landscapes into the training set was an important aspect to improve the effectiveness of training.

Throughout this work, the landscapes used for training were produced by SFERE [66] using one specific parameter configuration that is detailed in Table 3.1. We recommend that this parameter space is extended to include landscapes with varying physical properties for the behaviour of the sediment and vegetation in the landscape. By considering various parameter values for these terms, we can produce landscapes with vastly different channel widths, levels of recursive branching, and vegetation behaviour. This will increase the spatial variance in the training set, thereby elevating the potential learning effectiveness. In doing so, it could even be attempted to work with spatially varying parameters. In relation to the previous paragraph, adding landscapes to the training set can be very valuable if they are generated with different parameter configurations for SFERE [66]. The parameter choices for our instance of SFERE were based on the original implementation. For future works we recommend to instead choose parameters based on, for example, the effective size of the domain.

As the model complexity increased between the three main conceptualisations in this work (i.e. the SWE-PINN, the Hydro-PINN and the Saltmarsh-PINN), we have identified several aspects of the learning strategy to be impactful. Firstly, the use of gradient surgery (or possibly a similar multi-objective learning strategy) fulfils an almost foundational role in the learning of a multi-PDE system. Secondly, the physical domains need to be sufficiently *perturbed* for the PINN to effectively learn. Think of, for example, the oscillators as the drivers of training in the case of the SWE-PINN or the more spatially varying sedimentary and vegetation layers in the case of the Hydro-PINN. Thirdly, there is a trade-off in the modelling of the tides. Only modelling ebb-flow (like was done in SFERE and also in this thesis) leads to equilibrium solutions, because of which there is almost no temporal variation in the training set. But modelling the complete rising and falling motion of the tides requires a more complex formulation of the problem for which there is no (numerical) reference. Next, severing the closed boundary and allowing water to flow in and out of the domain requires the model to capture the conservation laws inside the landscape without the possibility of inserting hard constraints. Furthermore, large regions of dryness that resulted from this open boundary required the implementation of a wetting-drying approach. This was challenging after the implicit field descriptor through the use of Hermite Splines. And of course, because the sedimentary and vegetative layers in the landscape had become the drivers of the training process, we had to look for a way to make these landscapes dynamic without relying on the ability of the PINN to already be capable of solve the morphological PDEs. The latter is something that is inherently difficult to achieve, as the water dynamics drive the self-organisation. Training a

PINN to solve these morphological equations is therefore impossible without a working hydrodynamical model. This resulted in the development of the Image Fitting CNN that allowed us to place a numerical morphological solver alongside the PINN during training.

Each step in this process contains several important design decisions. Throughout this work, we tried to choose, as much as possible, the approach that would stay as close as possible to the working mechanisms. The design was setup in such a way that we could always make a reference to SFERE [66] or the working mechanisms behind the original Spline-PINN [73]. This proved helpful for hypothesising, but also to find a working explanation why certain aspects did or did not work. However, simply because our choices were based on working mechanisms, does not mean they are necessarily the best fit for the PINN. In this thesis, we strived to arrive at a working PINN model for the simulation of tidal channels in salt-marshes within the available time frame. In doing so, the design decisions that we made in the different design stages have been targeted towards a conceptualisation or proof of concept of the model. The quality and effectiveness of the resulting simulations have likely been affected by suboptimal choices in these stages.

Our main recommendation is therefore to re-iterate over these design decisions and to investigate to what extent it has impacted the quality and accuracy of both the physical solutions as well as the training efficiency of the PINN. The main focus of a continuation of this work should be on the hydrodynamics inside these coastal landscapes. They form the driving mechanism for the scale-dependent feedbacks in the wetland and they are the most limiting factor in numerical simulation. If we could produce a PINN that accurately solves these hydrodynamics, with steps of one second per iteration, this would already significantly impact future simulation possibilities. A concrete recommendation worth mentioning is that the explicit modelling of the rising and falling motion of the tides at the connection of the wetland to the main tidal channel could be beneficial for training of these PINNs. By directly controlling the water level at the open boundary, we gain complete control over the volume of water in the domain and the conservation laws can likely more easily be grasped by the PINN. Moreover, the tidal motion of the water will prevent the solutions from reaching an equilibrium and this could positively impact training by providing much larger degrees of spatial variance and information-providing physical events inside the domain. In such scenario, wetting-drying will likely become an important design aspect.

8

Conclusion

This thesis set out to embed a complex system of five non-linear multi-scale hydro-morpho-dynamical equations into the mechanism of a PINN. The system contains three PDEs to capture the hydrodynamics inside a partially open coastal wetland and two additional PDEs describe the evolution of the sedimentary bed elevation and the vegetation stem density inside the landscape. This overarching goal was subdivided into three research objectives. First, we strived to build a conceptual approach for a PINN to solve the SWEs in the simplified setting of closed water basins. Then, we aimed to extend this conceptualisation into the realm of hydrodynamics in a coastal wetland. Finally, we aimed to model the complete five-equation system with a PINN.

From the first and simplest closed-basin approach, our results showed that gradient surgery [77] has a significant and positive impact on the ability of the PINN to capture the boundary conditions. We further showed that the use of oscillators to introduce disturbances on the water surface facilitates effective learning of the SWEs across the entire domain and along the boundaries. We showed that our PINN can accurately solve the SWEs with large time-steps of one second per iteration, and in complexly shaped domains.

Analogous to these oscillators, learning of the SWEs inside a partially open coastal wetland was done by using various landscape profiles from the SFERE simulation [66]. By tasking the PINN to solve the SWEs in these fixed landscapes (i.e. with a fixed bed topography and vegetation), we showed that landscapes with more spatial variance lead to more effective training. We designed a CNN to transform numerical morphological solutions to the spline latent space used by our PINN. This enabled training in a hybrid setup to facilitate such temporal landscape development. Through this setup, we showed that landscapes that develop morphologically during training can further aid the training process. Our implementations for wetting-drying and the modelling of open boundaries seemed effective, however alternatives are worthwhile exploring. With our methods, training of hydrodynamics alongside a numerical morphological solver leads to hydrological solutions that are accurate enough to trigger spontaneous channelisation in the soil. However, the channelisation mainly occurs on one scale, with no recursion.

Embedding all five equations is difficult inside a single U-Net. This is caused by the temporal and spatial scale-difference between the processes in the landscape. Therefore, we modelled an ensemble of three PINNs to solve the hydrodynamics, the sedimentary dynamics and the vegetation dynamics separately. Our results indicate that simultaneous training of these three PINNs leads to less accurate dynamics. This signalled that the performance of one network in such an ensemble can directly affect the training effectiveness of another. With our setup, it is possible to implement a form of multi-stage training, to allow each network to learn separately. However, this work does not test the effectiveness of such strategy.

References

[1] Apr. 2026. URL: https://visualpde.com/fluids/shallow_water.html.

[2] Philip Ball. *The Self-Made Tapestry: Pattern formation in nature*. Oxford University Press, Oct. 1998. ISBN: 9780198502449. DOI: 10.1093/oso/9780198502449.001.0001. URL: https://doi.org/10.1093/oso/9780198502449.001.0001.

[3] Juan A Bonachela et al. "Termite mounds can increase the robustness of dryland ecosystems to climatic change". In: *Science* 347.6222 (2015), pp. 651–655.

[4] Matthew W. Brand et al. "Multi-decadal simulation of estuarine sedimentation under sea level rise with a response-surface surrogate model". In: *Advances in Water Resources* 150 (2021), p. 103876. ISSN: 0309-1708. DOI: https://doi.org/10.1016/j.advwatres.2021.103876. URL: https://www.sciencedirect.com/science/article/pii/S0309170821000312.

[5] Rüdiger Brecht, Elsa Cardoso-Bihlo, and Alex Bihlo. "Physics-informed neural networks for tsunami inundation modeling". In: *Journal of Computational Physics* 536 (2025), p. 114066.

[6] Giovanni Coco and A Brad Murray. "Patterns in the sand: From forcing templates to self-organization". In: *Geomorphology* 91.3-4 (2007), pp. 271–290.

[7] Loreta Cornacchia et al. "Vegetation traits and biogeomorphic complexity shape the resilience of salt marshes to sea-level rise". In: *Communications Earth & Environment* 5.1 (2024), p. 658.

[8] John Day et al. "Coastal wetlands in the Anthropocene". In: *Annual Review of Environment and Resources* 49.1 (2024), pp. 105–135.

[9] Susanna Dazzi. "Physics-Informed Neural Networks for the Augmented System of Shallow Water Equations With Topography". In: *Water Resources Research* 60.10 (2024). e2023WR036589 2023WR036589, e2023WR036589. DOI: https://doi.org/10.1029/2023WR036589. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2023WR036589. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2023WR036589.

[10] Roeland C van De Vijsel et al. "Estuarine biofilm patterns: Modern analogues for Precambrian self-organization". In: *Earth Surface Processes and Landforms* 45.5 (2020), pp. 1141–1154.

[11] Deltares. *Delft3D-FLOW: Simulation of multi-dimensional hydrodynamic flows and transport phenomena, including sediments*. User Manual, Version 4.07.02. Deltares. 2026. URL: https://oss.deltares.nl/web/delft3d/manuals.

[12] Deltares. *Delft3D-WAVE: Simulation of short-crested waves with SWAN*. User Manual, Version 4.07.02. Deltares. 2026. URL: https://oss.deltares.nl/web/delft3d/manuals.

[13] Rashmi C. Desai and Raymond Kapral. "Lifshitz–Slyozov–Wagner theory". In: *Dynamics of Self-Organized and Self-Assembled Structures*. Cambridge University Press, 2009, pp. 87–95.

[14] James Donnelly, Soroush Abolfathi, and Alireza Daneshkhah. "A physics-informed neural network surrogate model for tidal simulations". In: *UNCECOMP 2023 : 5th ECCOMAS Thematic Conference on Uncertainty Quantification in Computational Science and Engineering*. European Community on Computational Methods in Applied Sciences (ECCOMAS), Oct. 2023, pp. 836–844. DOI: 10.7712/120223.10379.19908. URL: https://doi.org/10.7712/120223.10379.19908.

[15] R. Dur. *Deep Learning of Vegetation Channel Networks in Coastal Wetlands*. 2026. URL: https://github.com/RensDur/Deep-Learning-of-Vegetation-Channel-Networks-in-Coastal-Wetlands (visited on 09/01/2026).

[16] R. Dur. *SFERE-PyTorch*. 2026. URL: https://github.com/RensDur/SFERE-PyTorch (visited on 09/01/2026).

[17] Majdi Fanous et al. “Deep probabilistic surrogate modelling for uncertainty quantification in mangrove hydro-morphodynamics”. In: *Frontiers in Marine Science* Volume 12 - 2025 (2026). ISSN: 2296-7745. DOI: 10.3389/fmars.2025.1624244. URL: https://www.frontiersin.org/journals/marine-science/articles/10.3389/fmars.2025.1624244.

[18] Amer Farea, Olli Yli-Harja, and Frank Emmert-Streib. “Understanding physics-informed neural networks: Techniques, applications, trends, and challenges”. In: *Ai* 5.3 (2024), pp. 1534–1557.

[19] Mohammad Ahmadi Gharehtoragh and David R Johnson. “Surrogate Models to Predict Wave Hydrodynamics on Evolving Landscapes”. In: *arXiv preprint arXiv:2510.12986* (2025).

[20] Mohammad Ahmadi Gharehtoragh and David R. Johnson. “Using surrogate modeling to predict storm surge on evolving landscapes under climate change”. In: *npj Natural Hazards* 1.1 (2024), p. 33. DOI: 10.1038/s44304-024-00032-9. URL: https://doi.org/10.1038/s44304-024-00032-9.

[21] Xavier Glorot and Yoshua Bengio. “Understanding the difficulty of training deep feedforward neural networks”. In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. 2010, pp. 249–256.

[22] Lucas Goehring. “Pattern formation in the geosciences”. In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 371.2004 (2013).

[23] Abigail J Hillen-Schiller et al. “Using photorealistic 3D visualization to convey ecosystem restoration to the public”. In: *Frontiers in Climate* 7 (2025), p. 1525331.

[24] Zoe J Hughes. “Tidal channels on tidal flats and marshes”. In: *Principles of tidal sedimentology*. Springer, 2012, pp. 269–300.

[25] Sergey Ioffe and Christian Szegedy. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. In: *Proceedings of the 32nd International Conference on Machine Learning*. Ed. by Francis Bach and David Blei. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, July 2015, pp. 448–456. URL: https://proceedings.mlr.press/v37/ioffe15.html.

[26] Md Amirul Islam et al. “Position, Padding and Predictions: A Deeper Look at Position Information in CNNs”. In: *International Journal of Computer Vision* 132.9 (2024), pp. 3889–3910.

[27] Ameya D Jagtap and George Em Karniadakis. “Extended physics-informed neural networks (XPINNs): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations”. In: *Communications in Computational Physics* 28.5 (2020).

[28] Ameya D Jagtap, Ehsan Kharazmi, and George Em Karniadakis. “Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems”. In: *Computer Methods in Applied Mechanics and Engineering* 365 (2020), p. 113028.

[29] Liang Jiang et al. “A Parameterized Prediction Method for Turbulent Jet Noise Based on Physics-Informed Neural Networks”. In: *Journal of Vibration and Acoustics* 147.2 (2025).

[30] Norbert Juergens. “The Biological Underpinnings of Namib Desert Fairy Circles”. In: *Science* 339.6127 (2013), pp. 1618–1621. DOI: 10.1126/science.1222999. eprint: https://www.science.org/doi/pdf/10.1126/science.1222999. URL: https://www.science.org/doi/abs/10.1126/science.1222999.

[31] William S. Kearney and Sergio Fagherazzi. “Salt marsh vegetation promotes efficient tidal channel networks”. In: *Nature Communications* 7.1 (2016), p. 12287.

[32] Dongjin Kim and Jaewook Lee. “A review of physics informed neural networks for multiscale analysis and inverse problems”. In: *Multiscale Science and Engineering* 6.1 (2024), pp. 1–11.

[33] Shigeru Kondo. “The reaction-diffusion system: A mechanism for autonomous pattern formation in the animal skin”. In: *Genes to Cells* 7.6 (2002), pp. 535–541.

[34] Shigeru Kondo and Takashi Miura. “Reaction-diffusion model as a framework for understanding biological pattern formation”. In: *science* 329.5999 (2010), pp. 1616–1620.

[35] Johan van de Koppel et al. “Scale-dependent feedback and regular spatial patterns in young mussel beds”. In: *The American Naturalist* 165.3 (2005), E66–E77.

[36] Yann LeCun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[37] Shicheng Li, Can Ding, and James Yang. “Surrogate modeling for rapid estimation of spatially-resolved flood damage: Application to a coastal region”. In: *Journal of Hydrology* (2025), p. 134763.

[38] Quan-Xing Liu et al. “Pattern formation at multiple spatial scales drives the resilience of mussel bed ecosystems”. In: *Nature communications* 5.1 (2014), p. 5234.

[39] Kuang Luo et al. “Physics-informed neural networks for PDE problems: a comprehensive review”. In: *Artificial Intelligence Review* 58.10 (2025), p. 323.

[40] G. Mariotti, W.S. Kearney, and S. Fagherazzi. “Soil creep in salt marshes”. In: *Geology* 44.6 (June 2016), pp. 459–462. ISSN: 0091-7613. DOI: 10.1130/G37708.1. eprint: https://pubs.geoscienceworld.org/gsa/geology/article-pdf/44/6/459/3550366/459.pdf. URL: https://doi.org/10.1130/G37708.1.

[41] Hans Meinhardt. *The algorithmic beauty of sea shells*. Springer Science & Business Media, 2009.

[42] AB Murray et al. “Biomorphodynamics: Physical-biological feedbacks that shape landscapes”. In: *Water Resources Research* 44.11 (2008).

[43] Siddharth Narayan et al. “The Value of Coastal Wetlands for Flood Damage Reduction in the Northeastern USA”. In: *Scientific Reports* 7.1 (2017), p. 9463.

[44] Anastasia Pentina, Viktoriia Sharmanska, and Christoph H. Lampert. “Curriculum Learning of Multiple Tasks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2015.

[45] J Taylor Perron, James W Kirchner, and William E Dietrich. “Formation of evenly spaced ridges and valleys”. In: *Nature* 460.7254 (2009), pp. 502–505.

[46] J Taylor Perron et al. “The root of branching river networks”. In: *Nature* 492.7427 (2012), pp. 100–103.

[47] Robert M Pringle et al. “Spatial pattern enhances ecosystem functioning in an African savanna”. In: *PLoS biology* 8.5 (2010), e1000377.

[48] PyTorch. *PyTorch*. 2026. URL: https://pytorch.org (visited on 09/01/2026).

[49] Xin Qi, Gustavo A.M. de Almeida, and Sergio Maldonado. “Physics-informed neural networks for solving flow problems modeled by the 2D Shallow Water Equations without labeled data”. In: *Journal of Hydrology* 636 (2024), p. 131263.

[50] Nasim Rahaman et al. “On the Spectral Bias of Neural Networks”. In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, Sept. 2019, pp. 5301–5310. URL: https://proceedings.mlr.press/v97/rahaman19a.html.

[51] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations”. In: *Journal of Computational physics* 378 (2019), pp. 686–707.

[52] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. “Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations”. In: *arXiv preprint arXiv:1711.10561* (2017).

[53] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. *Physics Informed Deep Learning (Part II): Data-driven Discovery of Nonlinear Partial Differential Equations*. 2017. arXiv: 1711.10566 [cs.AI]. URL: https://arxiv.org/abs/1711.10566.

[54] Max Rietkerk and Johan van de Koppel. “Regular pattern formation in real ecosystems”. In: *Trends in Ecology & Evolution* 23.3 (2008), pp. 169–175. ISSN: 0169-5347. DOI: https://doi.org/10.1016/j.tree.2007.10.013. URL: https://www.sciencedirect.com/science/article/pii/S0169534708000281.

[55] Max Rietkerk et al. “Evasion of tipping in complex systems through spatial pattern formation”. In: *Science* 374.6564 (2021), eabj0359. DOI: 10.1126/science.abj0359. eprint: https://www.science.org/doi/pdf/10.1126/science.abj0359. URL: https://www.science.org/doi/abs/10.1126/science.abj0359.

[56] Max Rietkerk et al. “Self-organized patchiness and catastrophic shifts in ecosystems”. In: *Science* 305.5692 (2004), pp. 1926–1929.

[57] Andrea Rinaldo et al. “Tidal networks: 2. Watershed delineation and comparative network morphology”. In: *Water Resources Research* 35.12 (1999), pp. 3905–3917.

[58] Ignacio Rodriguez-Iturbe and Andrea Rinaldo. *Fractal river basins: chance and self-organization*. Cambridge University Press, 1997.

[59] Eleonora Saccon. “Tidal freshwater wetlands under salt intrusion”. In: (2025).

[60] Christian Schwarz et al. “Self-organization of a biogeomorphic landscape controlled by plant life-history traits”. In: *Nature Geoscience* 11.9 (2018), pp. 672–677.

[61] Christine C Shepard, Caitlin M Crain, and Michael W Beck. “The protective role of coastal marshes: a systematic review and meta-analysis”. In: *PloS one* 6.11 (2011), e27374.

[62] Stijn Temmerman et al. “Vegetation causes channel erosion in a tidal landscape”. In: *Geology* 35.7 (2007), pp. 631–634.

[63] J. ARTHUR THOMSON. “On Growth and Form”. In: *Nature* 100.2498 (1917), pp. 21–22. DOI: 10.1038/100021a0. URL: https://doi.org/10.1038/100021a0.

[64] Alan Turing. “The chemical basis of morphogenesis (1952)”. In: *B Jack Copeland* 519 (1952).

[65] A Valsamidou et al. “A hydrological switch drives the transition from saltmarsh to reedland ecosystem”. In: *14th Symposium on River, Coastal and Estuarine Morphodynamics, RCEM 2025*. Universitat Politècnica de Catalunya. 2025, pp. 237–237.

[66] Roeland C Van de Vijsel et al. “Vegetation controls on channel network complexity in coastal wetlands”. In: *Nature communications* 14.1 (2023), p. 7158.

[67] R.C. van de Vijsel. *SFERE*. 2023. URL: https://github.com/RCvandeVijsel/SFERE (visited on 09/01/2026).

[68] Roeland C. van de Vijsel et al. “Algal-Induced Biogeomorphic Feedbacks Lay the Groundwork for Coastal Wetland Development”. In: *Journal of Geophysical Research: Biogeosciences* 126.10 (2021). e2021JG006515 2021JG006515, e2021JG006515. DOI: https://doi.org/10.1029/2021JG006515. eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/2021JG006515. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2021JG006515.

[69] Roeland C. van de Vijsel et al. *SFERE*. Version v1.0. Oct. 2023. DOI: 10.5281/zenodo.8428792. URL: https://doi.org/10.5281/zenodo.8428792.

[70] Roeland C. van de Vijsel et al. *Vegetation controls on channel network complexity in coastal wetlands - Dataset*. 2023. DOI: 10.4121/8d361887-ec02-4472-a8eb-a9d0f3eacfd6.v1.

[71] N. Wandel et al. *Spline_PINN*. 2022. URL: https://github.com/vc-bonn/Spline_PINN (visited on 09/01/2026).

[72] Nils Wandel, Michael Weinmann, and Reinhard Klein. “Learning Incompressible Fluid Dynamics from Scratch–Towards Fast, Differentiable Fluid Models that Generalize”. In: *arXiv preprint arXiv:2006.08762* (2020).

[73] Nils Wandel et al. “Spline-PINN: Approaching PDEs without Data Using Fast, Physics-Informed Hermite-Spline CNNs”. In: vol. 36. Cited by: 51; All Open Access, Gold Open Access. 2022, pp. 8529–8538. DOI: 10.1609/aaai.v36i8.20830. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85142866743&doi=10.1609%2faaai.v36i8.20830&partnerID=40&md5=149f8754b4e1518af0a14870294d2e3b.

[74] Sifan Wang et al. “An expert’s guide to training physics-informed neural networks”. In: *arXiv preprint arXiv:2308.08468* (2023).

[75] Yini Wang and Mingliang Zhang. “Modeling Hydrodynamic and Hydrological Processes in Tidal Wetlands”. In: *Wetlands* 42.1 (2021), p. 1. DOI: 10.1007/s13157-021-01519-1. URL: https://doi.org/10.1007/s13157-021-01519-1.

[76] Ellen J Weerman et al. “Spatial self-organization on intertidal mudflats through biophysical stress divergence”. In: *The American Naturalist* 176.1 (2010), E15–E32.

[77] Tianhe Yu et al. “Gradient Surgery for Multi-Task Learning”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 5824–5836.

[78] Vincent T. M. van Zelst et al. “Cutting the costs of coastal protection by integrating vegetation in flood defences”. In: *Nature Communications* 12.1 (2021), p. 6533.

[79] Richard Zhang et al. “The Unreasonable Effectiveness of Deep Features as a Perceptual Metric”. In: *CVPR*. 2018.

[80] Yinhao Zhu et al. “Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data”. In: *Journal of Computational Physics* 394 (2019), pp. 56–81.

[81] Zhenchang Zhu et al. “Historic storms and the hidden value of coastal wetlands for nature-based flood defence”. In: *Nature Sustainability* 3.10 (2020), pp. 853–862.

Figure Credits

Cover photo and Figure 1.4 – Salt marsh river network. Photo by Edwin Paree (NIOZ), cropped.

Figure 1.1 – Photo ”Sand ripples 2015” by Philip Ball, cropped. Licensed under CC BY 4.0 (https://creativecommons.org/licenses/by/4.0/deed.en). https://doi.org/10.1098/rstb.2014.0218

Figure 1.2 – Photo ”Romanescu” by Rlunaro, cropped. Licensed under CC BY-SA 3.0 (https://creativecommons.org/licenses/by-sa/3.0/deed.en).

Figure 1.3 – Photo ”Boxfish Aracana ornata patterns” by the Birch Aquarium at Scripps, cropped. Licensed under CC BY 4.0 (https://creativecommons.org/licenses/by/4.0/deed.en).

Figure 1.5 – Photo ”Mangrove” by Thomas Peschak, cropped. Licensed under CC BY-SA 4.0 (https://creativecommons.org/licenses/by-sa/4.0/deed.en).

Figure 1.6 – Photo by Norbert Dankers (Wageningen Marine Research), cropped.

Figure 3.1 – Figure adapted from van de Vijsel et al. (2023), ”Vegetation controls on channel network complexity in coastal wetlands”, Nature Communications 14, 7158, https://doi.org/10.1038/s41467-023-42731-3, licensed under CC BY 4.0 (https://creativecommons.org/licenses/by/4.0/).

AI Statement

Generative Artificial Intelligence and Large Language Models (LLMs) are becoming increasingly abundant and they can be great tools for many types of tasks. I recognise that they can be helpful in many scenarios and I find these models to be mostly helpful as a sparring partner to review ideas and to look for inspiration or pointers to relevant literature. I have always looked at the response of the AI tools with a critical eye, and I have never copied any content as part of the thesis. Moreover, I have not used any of these models to delegate parts of the work, such as generating code or designing parts of the research strategies which could simply be copied or executed. For example, I have made use of Claude and ChatGPT to learn more about the structural formation of the landscapes and how it impacts the functioning of the ecosystem. Such knowledge related to ecosystems and their behaviour is relatively new to me and I found the sparring conversations with these models about those topics to be very informative. Again, in those discussions I was always critically minded and never took Claude or ChatGPT for their word. For example, they helped me get used to the specific terminology, which in turn helped me understand literature more quickly and in more detail. As another example, I have made use of Cursor as a way to validate parts of my implementation. In those queries, I would explicitly ask the agent not to alter any code and I have made sure that no AI-generated code ended up in my repositories. But in cases where a lot of detailed changes needed to be made across several files, I found such validation tool to be useful for early-catching and preventing of mistakes.

A

Additional Figures and Alternative Approaches

This appendix contains results and figures that are supplementary to the work presented in the thesis. Additionally, this appendix includes some interesting alternative approaches that could be considered in future continuations of this work.

A.1. SWE Flux Formulation

Throughout this thesis, a flow velocity-based (non-conservative) formulation of the SWEs was used. In 2D, this system models one continuity equation and two equations for the flow velocities in x- and y-direction. Alternatively, one could consider a formulation that models fluid fluxes instead of velocities and the two formulations have different properties that mostly turn up in regions of the solutions that are non-smooth, such as shock. The non-conservative form offers no exact guarantee for conservation of mass and momentum and, because they model velocities directly, this formulation tends to turn unstable in regions where $h \to 0$ because the velocity terms explode. In conservative formulation, which models fluid *fluxes* hu and hv, the conservation laws are exactly guaranteed and the equations remain stable in near-dry regions as $hu, hv \to 0$ as $h \to 0$. This section presents the SWEs in conservative form and explains how they can be coupled to the sedimentary bed and vegetation in the landscape, in much the same way as SFERE [66].

Depending on the conditions in which the equations should hold, different formulations can be chosen. When directly derived from the conservation of mass and conservation of linear momentum (Navier-Stokes), one obtains three equations to describe the water column height h and the horizontal and vertical flow velocity components u and v, like used in [49]:

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}(\mathbf{U})}{\partial x} + \frac{\partial \mathbf{G}(\mathbf{U})}{\partial y} = \mathbf{S}(\mathbf{U}) \tag{A.1}$$

where

$$\mathbf{U} = \begin{bmatrix} h \\ hu \\ hv \end{bmatrix}, \mathbf{F}(\mathbf{U}) = \begin{bmatrix} hu \\ hu^2 + \frac{1}{2}gh^2 \\ huv \end{bmatrix}, \mathbf{G}(\mathbf{U}) = \begin{bmatrix} hv \\ huv \\ hv^2 + \frac{1}{2}gh^2 \end{bmatrix}, \mathbf{S}(\mathbf{U}) = \begin{bmatrix} 0 \\ gh(s_{ox} - s_{fx}) \\ gh(s_{oy} - s_{fy}) \end{bmatrix} \tag{A.2}$$

These equations are directly subject to the fundamental conservation laws, as they describe the rate of change of the water fluxes hu and hv in the domain. In this system, g is the gravitational acceleration constant, $s_{ox} = -\partial S/\partial x$ and $s_{oy} = -\partial S/\partial y$ are the bed slopes in x and y directions, respectively, and s_{fx} and s_{fy} are the bed friction components in x and y direction, respectively, and are given by Equation A.3, where n is Manning's coefficient.

$$\begin{aligned} s_{fx} &= n^2 u\sqrt{u^2+v^2}h^{-4/3} \\ s_{fy} &= n^2 v\sqrt{u^2+v^2}h^{-4/3} \end{aligned} \tag{A.3}$$

By introducing the Chézy coefficient as defined in [66] and rewriting, $\mathbf{S}(\mathbf{U})$ can be rewritten as:

$$\mathbf{S}(\mathbf{U}) = \begin{bmatrix} 0 \\ gh(s_{ox}-s_{fx}) \\ gh(s_{oy}-s_{fy}) \end{bmatrix} = \begin{bmatrix} 0 \\ -gh\frac{\partial S}{\partial x} - \frac{g}{C_z^2}u\sqrt{u^2+v^2} \\ -gh\frac{\partial S}{\partial y} - \frac{g}{C_z^2}v\sqrt{u^2+v^2} \end{bmatrix} \tag{A.4}$$

And by considering the bed shear stress components τ_{bx}/ρ and τ_{by}/ρ as defined in [66], it can be further simplified:

$$\mathbf{S}(\mathbf{U}) = \begin{bmatrix} 0 \\ -gh\frac{\partial S}{\partial x} - \frac{\tau_{bx}}{\rho} \\ -gh\frac{\partial S}{\partial y} - \frac{\tau_{by}}{\rho} \end{bmatrix} \tag{A.5}$$

Finally, by taking the equations out of vector form, we arrive at a system of equations that is directly connected to the conservation of mass and momentum and which allows coupling to the sedimentary bed elevation S, as well as vegetation stem density through the bed shear stress components τ_{bx}/ρ and τ_{by}/ρ:

$$\begin{aligned} \frac{\partial h}{\partial t} &= -\frac{\partial (hu)}{\partial x} - \frac{\partial (hv)}{\partial y} \\ \frac{\partial (hu)}{\partial t} &= -gh\frac{\partial S}{\partial x} - \frac{\partial}{\partial x}\left(hu^2 + \frac{1}{2}gh^2\right) - \frac{\partial}{\partial y}(huv) - \frac{\tau_{bx}}{\rho} \\ \frac{\partial (hv)}{\partial t} &= -gh\frac{\partial S}{\partial y} - \frac{\partial}{\partial y}\left(hv^2 + \frac{1}{2}gh^2\right) - \frac{\partial}{\partial x}(huv) - \frac{\tau_{by}}{\rho} \end{aligned} \tag{A.6}$$

A.2. Preliminary Ablation Study Results

To create the preliminary results of Figures A.1, A.2, A.3, A.4 and A.5, the water source term H_{in} was part of the continuity equation and therefore had to be learned. Furthermore, any negative water surface levels were not compensated for, leading to deep 'wells' below ground level that required a significant amount of water to fill. This likely amplified the negative effects of drought on the training productivity. In addition, the open boundary was poorly modelled by making use of the Laplacians of the flow velocity components $\nabla^2 u$ and $\nabla^2 v$:

$$\begin{aligned} \mathcal{L}_{\text{bound,open}} &= \int_{\partial\Omega_{\text{open}}} \left[\|\nabla^2 u\|^2 + \|\nabla^2 v\|^2 \right] \\ &= \int_{\partial\Omega_{\text{open}}} \left[\left\| \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right\|^2 + \left\| \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right\|^2 \right] \\ &\leq \int_{\partial\Omega_{\text{open}}} \left[\left\| \frac{\partial^2 u}{\partial x^2} \right\|^2 + \left\| \frac{\partial^2 u}{\partial y^2} \right\|^2 + \left\| \frac{\partial^2 v}{\partial x^2} \right\|^2 + \left\| \frac{\partial^2 v}{\partial y^2} \right\|^2 \right] \end{aligned}$$

to approximate the unified open boundary condition. This was done for practical reasons of implementation. Lastly, training of the individual networks was slightly unfair as the training sets only contained snapshots of SFERE with an interval of 10.000 iterations. Due to the logarithmic slowdown in the landscape structuring, later stages of landscape coarsening contained significantly more snapshots than earlier ones.

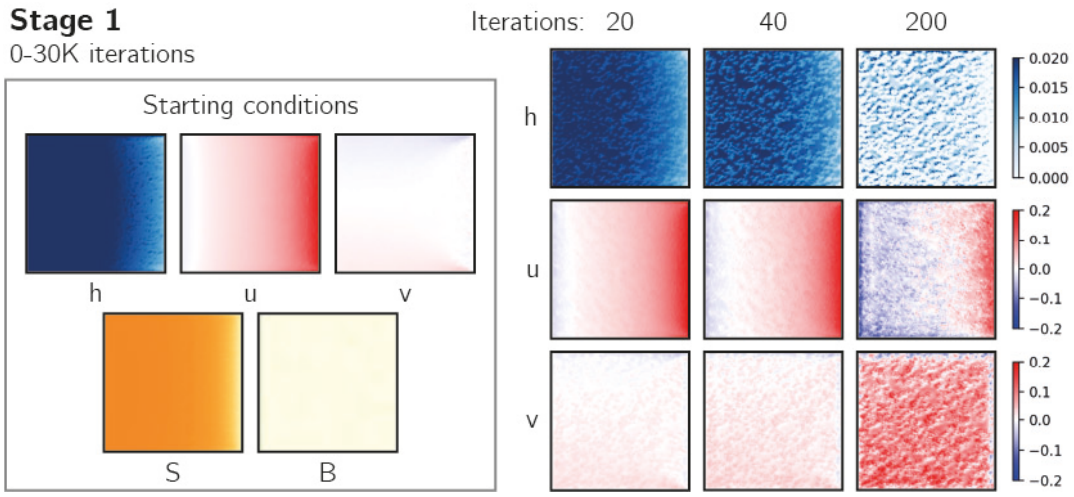


Figure A.1: Preliminary hydrodynamical predictions of **Hydro-PINN-Stage-1** after 20, 40 and 200 iterations.

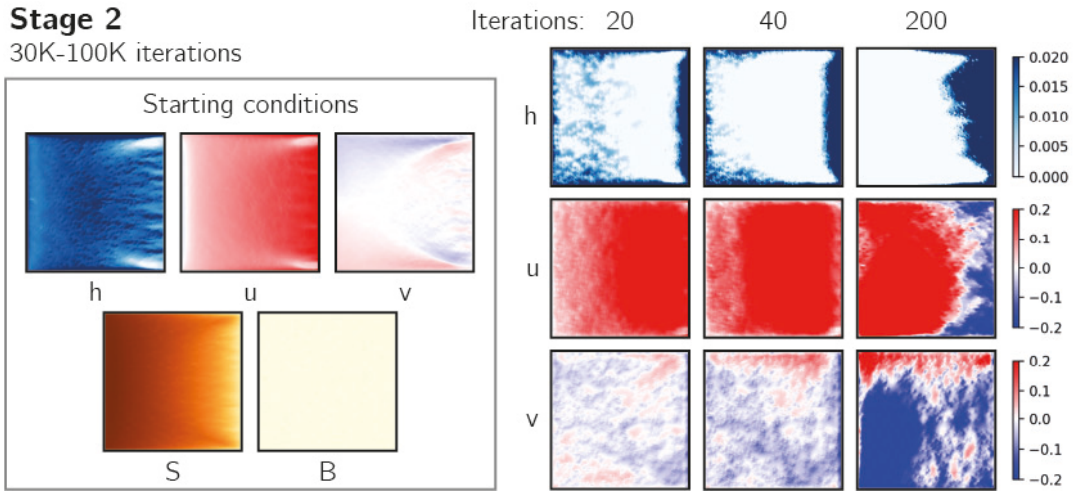


Figure A.2: Preliminary hydrodynamical predictions of **Hydro-PINN-Stage-2** after 20, 40 and 200 iterations.

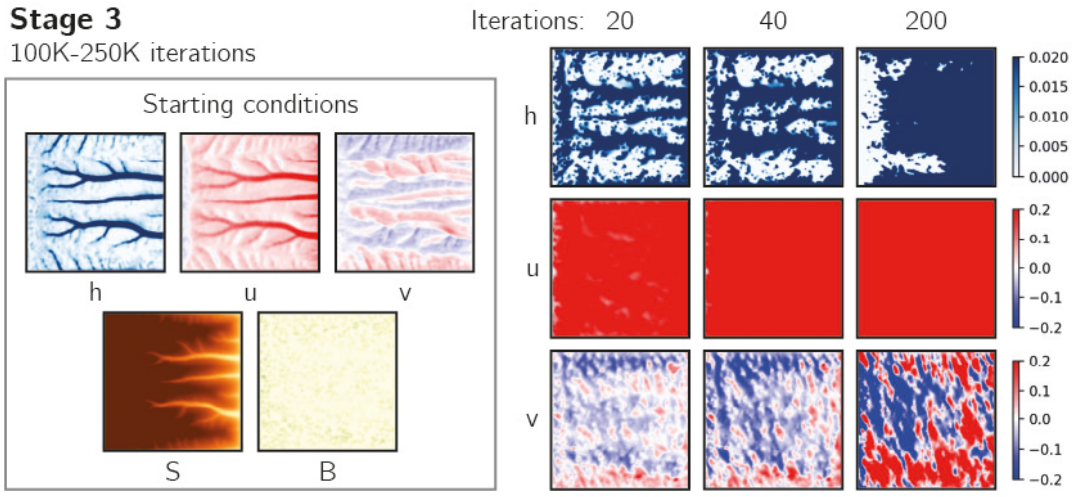


Figure A.3: Preliminary hydrodynamical predictions of **Hydro-PINN-Stage-3** after 20, 40 and 200 iterations.

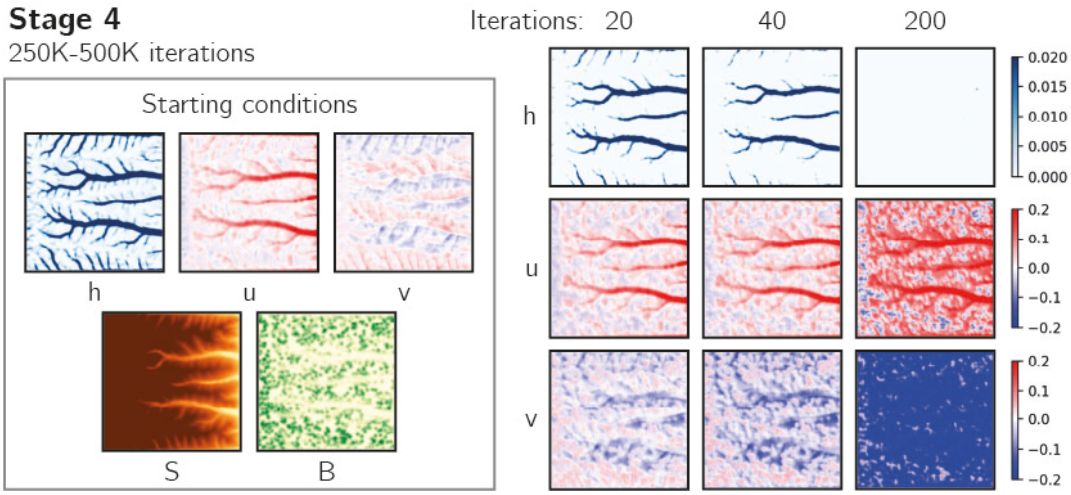


Figure A.4: Preliminary hydrodynamical predictions of **Hydro-PINN-Stage-4** after 20, 40 and 200 iterations.

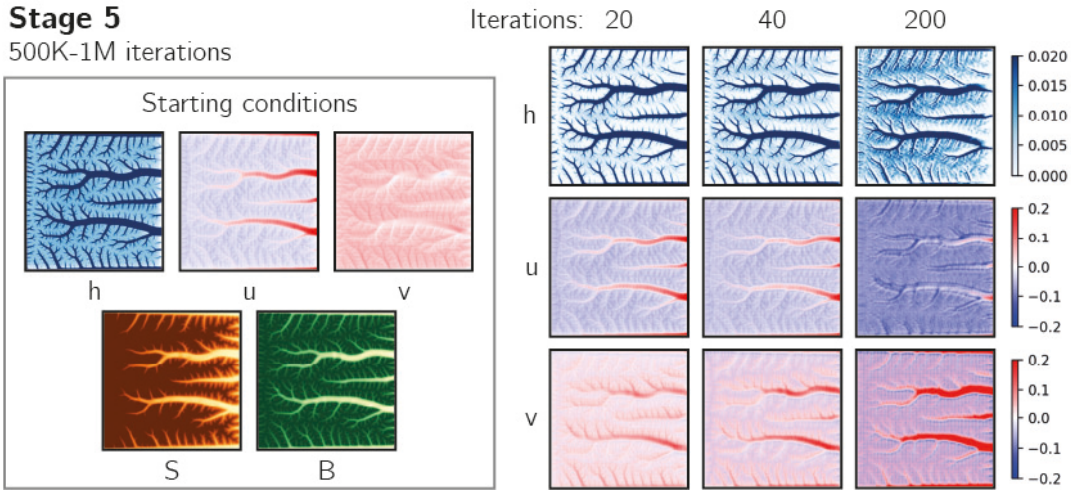
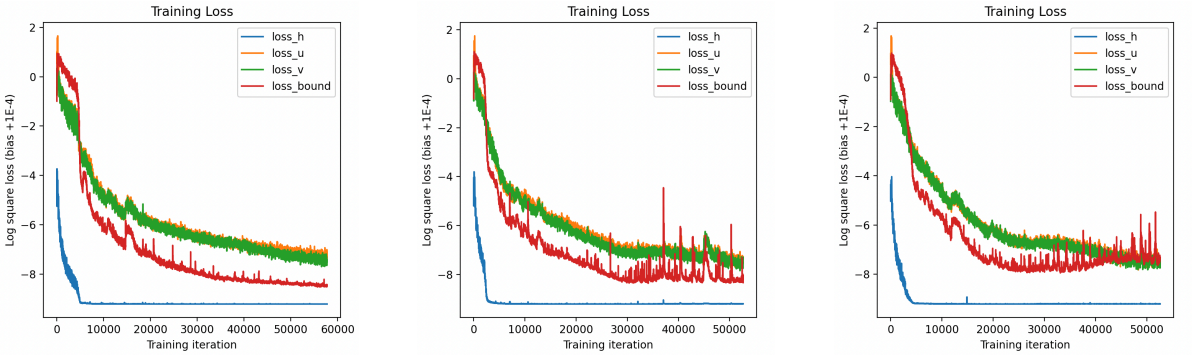


Figure A.5: Preliminary hydrodynamical predictions of **Hydro-PINN-Stage-5** after 20, 40 and 200 iterations.

A.3. Analysing the Impact of Non-Dimensionalisation

In Section 4.2.5, the importance of non-dimensionalisation is explained. To bring to light how the scaling of network in- and outputs could affect training, the five Hydro-PINNs **Hydro-PINN-Stage-1**, **Hydro-PINN-Stage-2**, **Hydro-PINN-Stage-3**, **Hydro-PINN-Stage-4** and **Hydro-PINN-Stage-5** are in this section re-trained with a naive scaling of variables and the results are compared to those in Section 6.3.1. In this naive approach, all network inputs are not pre-scaled and the output scaling is configured so that the solutions can at least reach the necessary output ranges for each of the variables in the system. Concretely, the first spline order of both flow velocity components is set to 10 and the first spline order of the vegetation stem density to 1500. All other spline orders are scaled with a factor of 2.

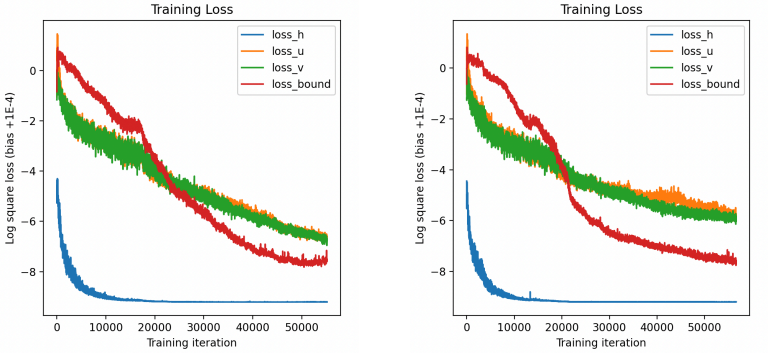
From the training loss in Figure A.6, it is immediately clear that training without non-dimensionalisation is significantly less effective. Where all Hydro-PINNs in the previous section converged after at most 10.000 iterations in Figure A.6, some of these models were only able to reach convergence after training for five times as many cycles. Moreover, it is clearly visible that **Hydro-PINN-Stage-4** and **Hydro-PINN-Stage-5** have not reached convergence at all.



(a) Training loss for **Hydro-PINN-Stage-1** with naive non-dimensionalisation.

(b) Training loss for **Hydro-PINN-Stage-2** with naive non-dimensionalisation.

(c) Training loss for **Hydro-PINN-Stage-3** with naive non-dimensionalisation.



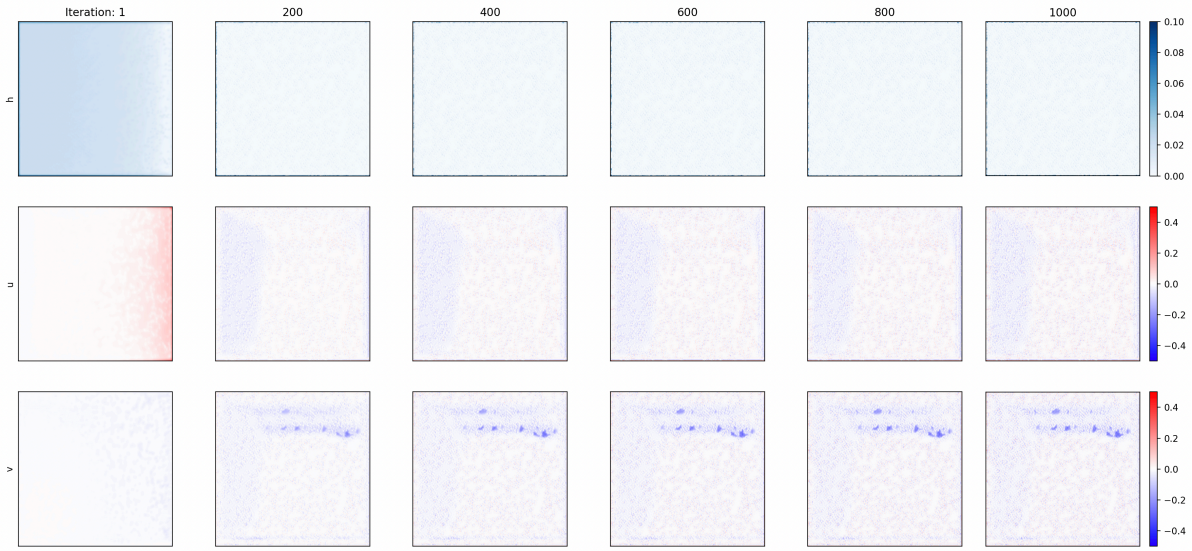
(d) Training loss for **Hydro-PINN-Stage-4** with naive non-dimensionalisation.

(e) Training loss for **Hydro-PINN-Stage-5** with naive non-dimensionalisation.

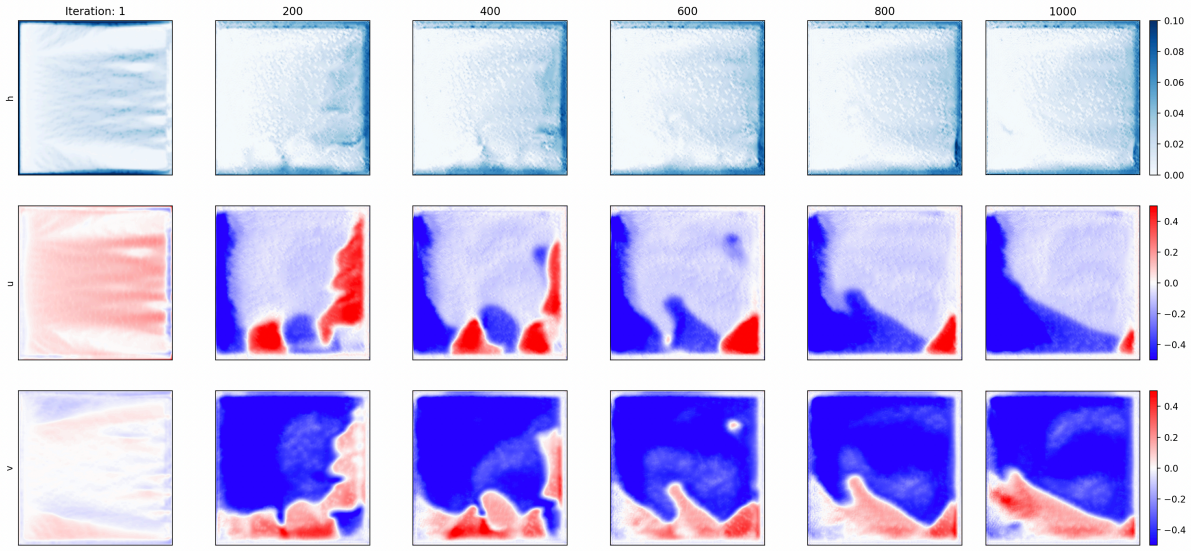
Figure A.6: Training loss for five identical Hydro-PINNs that are trained on independent datasets of equal size, extracted from SFERE, under a naive non-dimensionalisation of network in- and outputs.

Without non-dimensionalisation, **Hydro-PINN-Stage-1** produces a less defined output with more noise in Figure A.7a compared to Figure 6.19. The outflow through the open boundary is less strongly modelled and, especially in both flow velocity fields, the domain is filled with spotty artefacts. These artefacts are likely created by the randomly placed vegetation tussocks in the initial condition of SFERE. These tussocks are initialised as a single pixel where the vegetation density is set to maximum carrying capacity, which corresponds to a value of $1500\ m^{-2}$. This vegetation profile therefore contains significantly larger input values than other channels, causing the network to *over-react* to these signals.

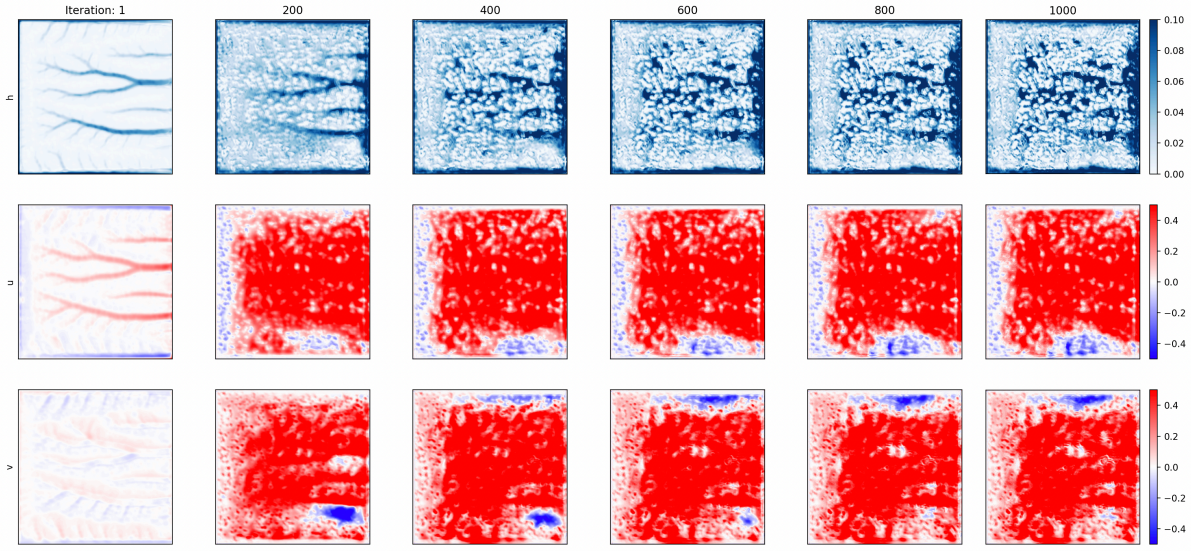
Similar vegetation-induced distortions can be observed in almost all other naively scaled experiments as well and they seem to become more significant as the vegetation starts growing faster in the domain. In Figure A.7c, it is clearly visible that all hydrological quantities are significantly impacted by the patches of vegetation and that the overall solution is much less adequate compared to the properly scaled



(a) Hydrodynamic solution, as predicted by **Hydro-PINN-Stage-1**, without appropriate non-dimensionalisation, after 1, 200, 400, 600, 800 and 1000 forward passes.

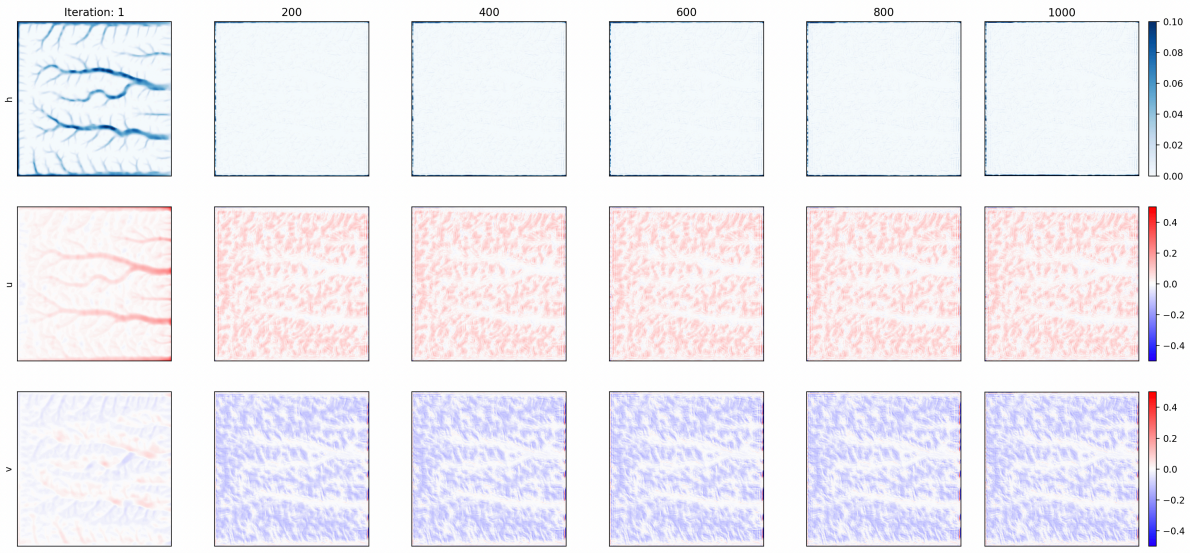


(b) Hydrodynamic solution, as predicted by **Hydro-PINN-Stage-2**, without appropriate non-dimensionalisation, after 1, 200, 400, 600, 800 and 1000 forward passes.

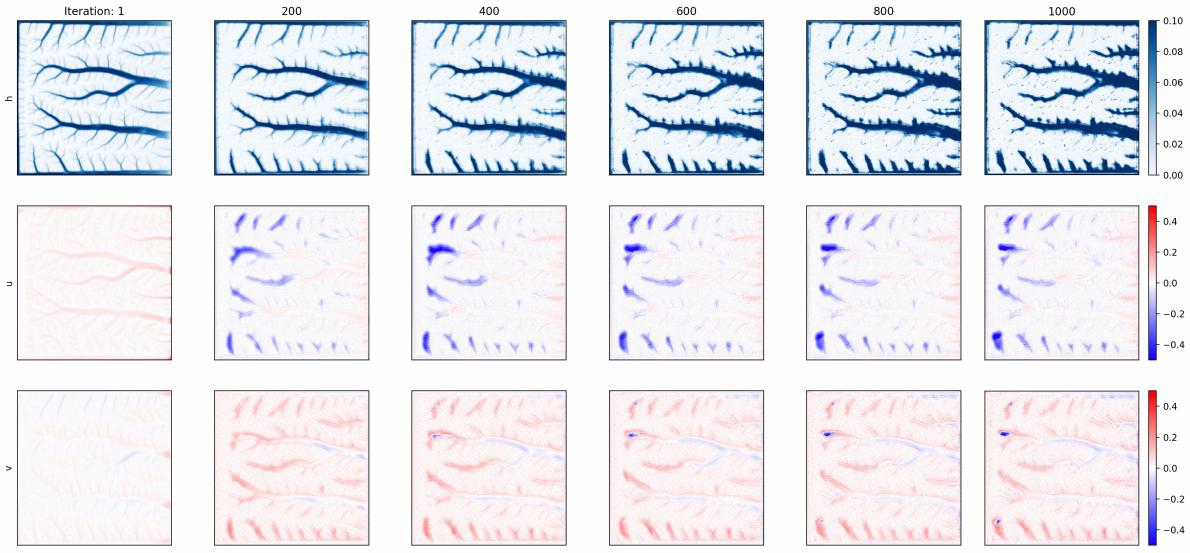


(c) Hydrodynamic solution, as predicted by **Hydro-PINN-Stage-3**, without appropriate non-dimensionalisation, after 1, 200, 400, 600, 800 and 1000 forward passes.

Figure A.7: Hydrodynamic solution, as predicted by five identical Hydro-PINNs with naive implementation of non-dimensionalisation that were trained on SFERE samples from each of the landscape coarsening categories defined in Figure 4.5.



(d) Hydrodynamic solution, as predicted by **Hydro-PINN-Stage-4**, without appropriate non-dimensionalisation, after 1, 200, 400, 600, 800 and 1000 forward passes.



(e) Hydrodynamic solution, as predicted by **Hydro-PINN-Stage-5**, without appropriate non-dimensionalisation, after 1, 200, 400, 600, 800 and 1000 forward passes.

Figure A.7: Hydrodynamic solution, as predicted by five identical Hydro-PINNs with naive implementation of non-dimensionalisation that were trained on SFERE samples from each of the landscape coarsening categories defined in Figure 4.5.

version in Figure 6.21. It impacts the solution so much that the horizontal and vertical flow velocities are seemingly unable to recognise the channelisation in the underlying topography, something that **Hydro-PINN-Stage-3** was much more capable of in the original study.

One could argue that the solutions produced by the naive **Hydro-PINN-Stage-5** in Figure A.7e are somewhat adequate. Water seems to gather in the channel network, although with less precision compared to Figure 6.23, and the water seems to accelerate as it moves along the channels towards the open boundary. However, the vertical flow field does not properly reflect water moving away from uplands and into the channels. This can be seen by the positive (downwards) flow below each of the channels, an area where water flows in the negative direction (upwards) in the SFERE simulation and also in the original **Hydro-PINN-Stage-5** in Figure 6.23.

A.4. Hydrodynamics in Static Landscapes

In Section 6.3.2, our Hydro-PINN is trained in hybrid with a numerical morphological solver and evaluated on static landscapes. Also, the resulting hydrodynamics are compared to the ablation study results and the hydrodynamics produced by the Saltmarsh-PINN. This section presents additional figures corresponding to the results presented in that section. Training of the Hydro-PINN in hybrid took 36 hours on an NVIDIA A40 and the training loss is indicated in Figure A.8, where it can be seen that training has converged.

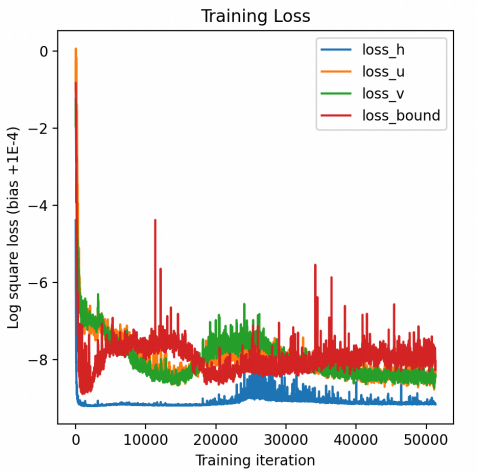
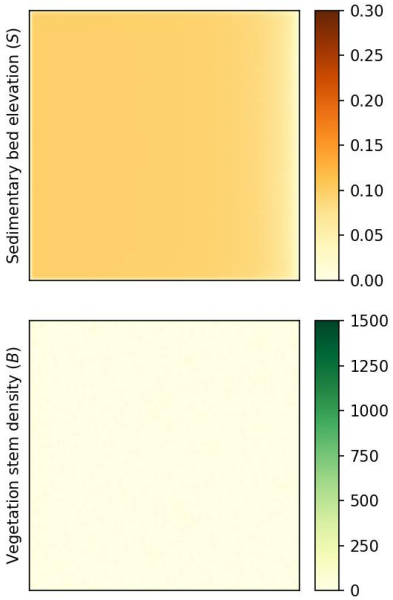
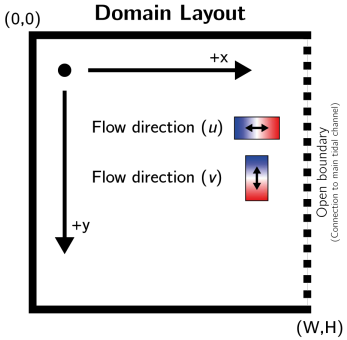


Figure A.8: Training loss for the Hydro-PINN when trained in a hybrid setup with a numerical solver for sediment and vegetation on SFERE sample across all landscape categories

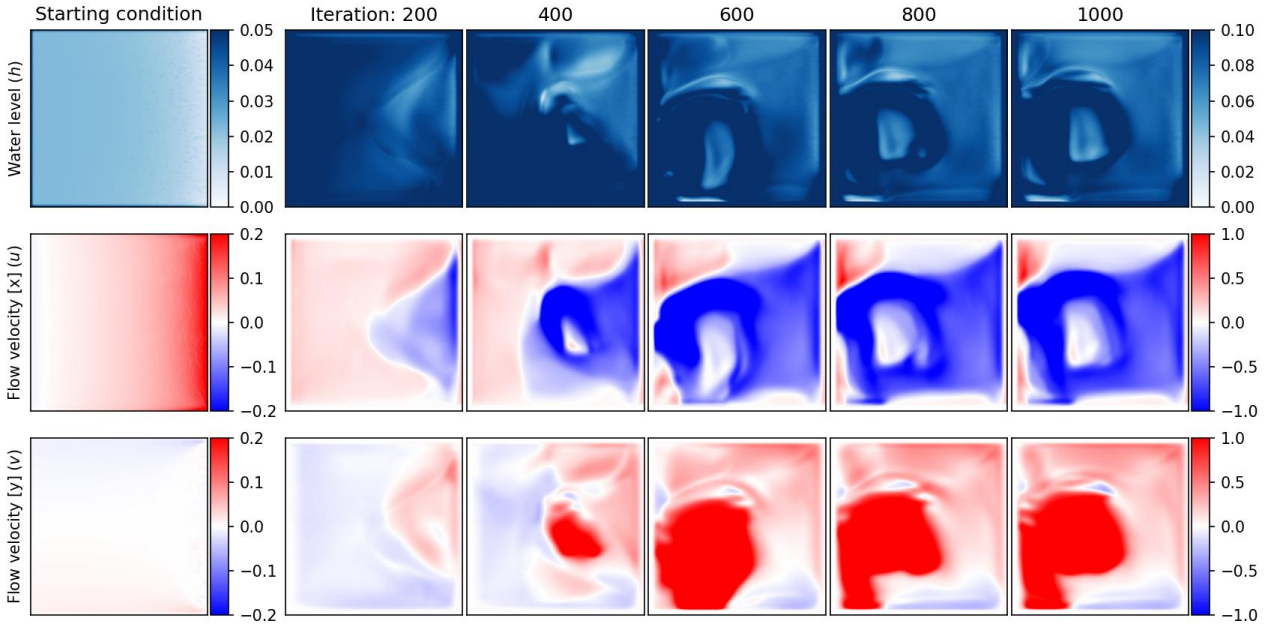
Figures A.9, A.10, A.11, A.12 and A.13 correspond to the results presented in Figure 6.27 and Figures A.14, A.15, A.16, A.17 and A.18 correspond to the results presented in Figure 6.28.



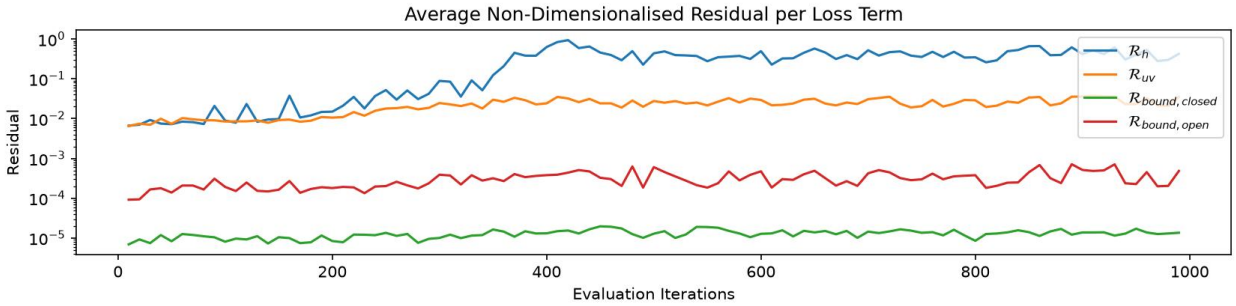
(a) Landscape structuring (Stage 1)



(b) Domain layout

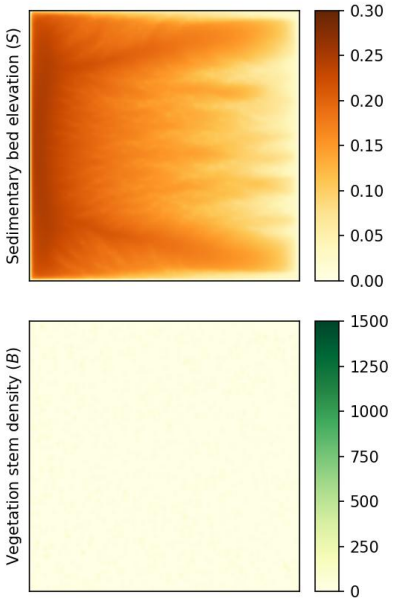


(c) Hydrodynamic solution predicted by the PINN after 200, 400, 600, 800 and 1000 iterations.

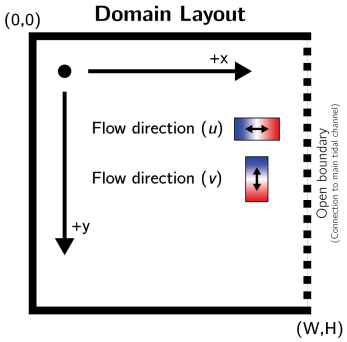


(d) Average Non-Dimensionalised Residual calculated for each term separately.

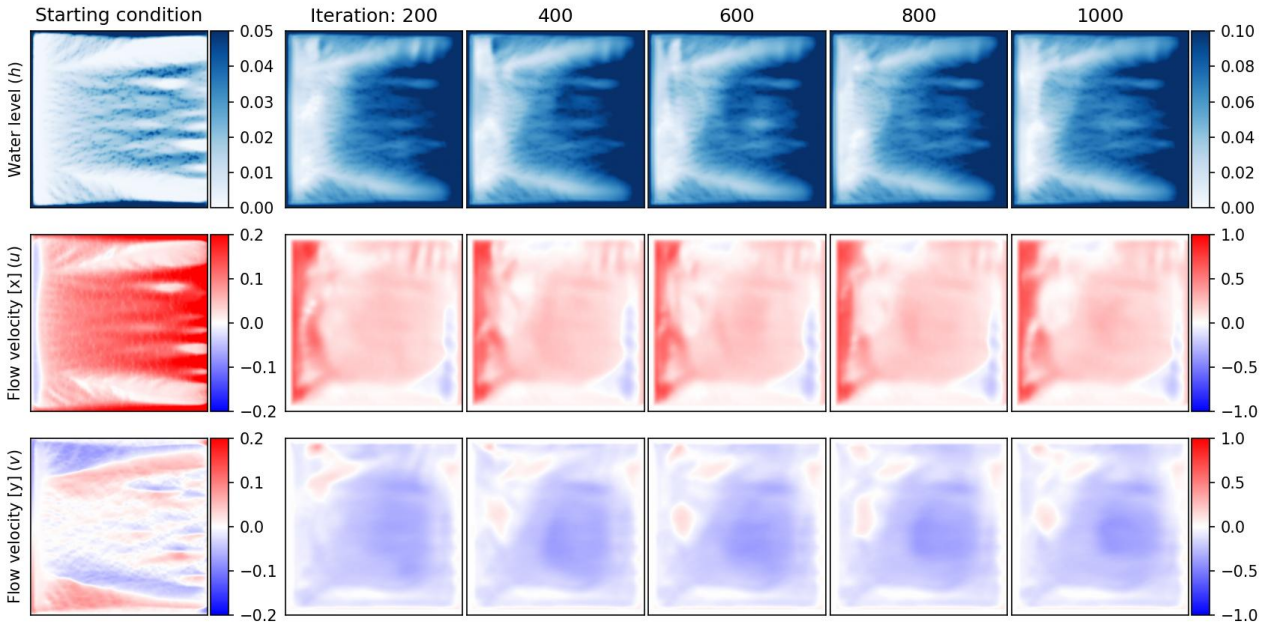
Figure A.9: Hydro-PINN trained on all 500 samples in the training set alongside a numerical solver for the morphological quantities. The PINN is evaluated on 100 samples corresponding to the first stage of landscape coarsening.



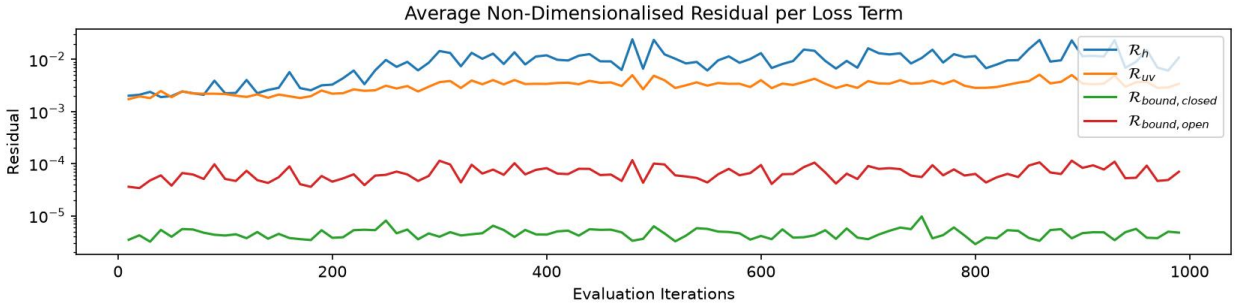
(a) Landscape structuring (Stage 2)



(b) Domain layout

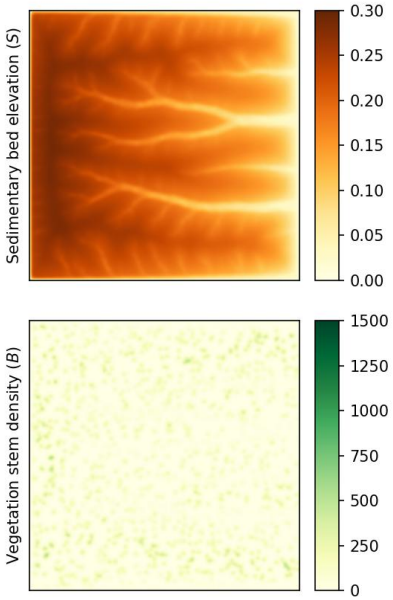


(c) Hydrodynamic solution predicted by the PINN after 200, 400, 600, 800 and 1000 iterations.

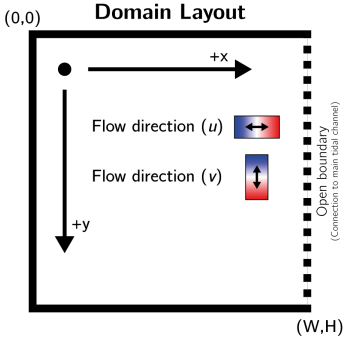


(d) Average Non-Dimensionalised Residual calculated for each term separately.

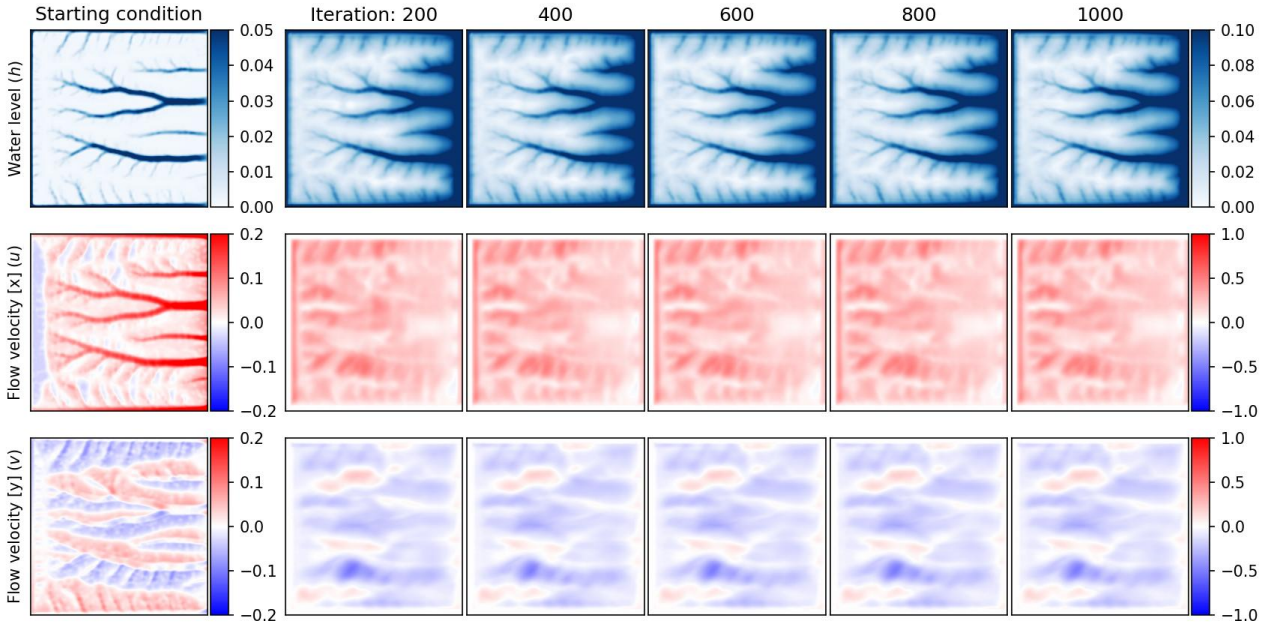
Figure A.10: Hydro-PINN trained on all 500 samples in the training set alongside a numerical solver for the morphological quantities. The PINN is evaluated on 100 samples corresponding to the second stage of landscape coarsening.



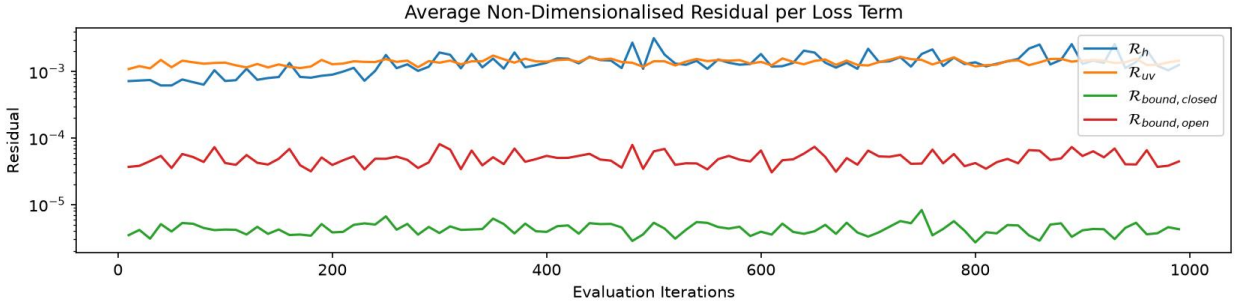
(a) Landscape structuring (Stage 3)



(b) Domain layout

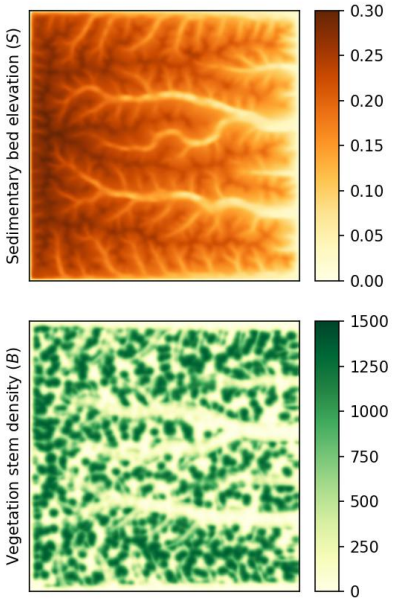


(c) Hydrodynamic solution predicted by the PINN after 200, 400, 600, 800 and 1000 iterations.

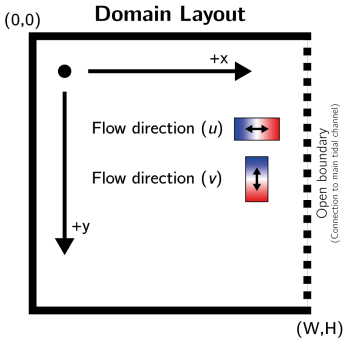


(d) Average Non-Dimensionalised Residual calculated for each term separately.

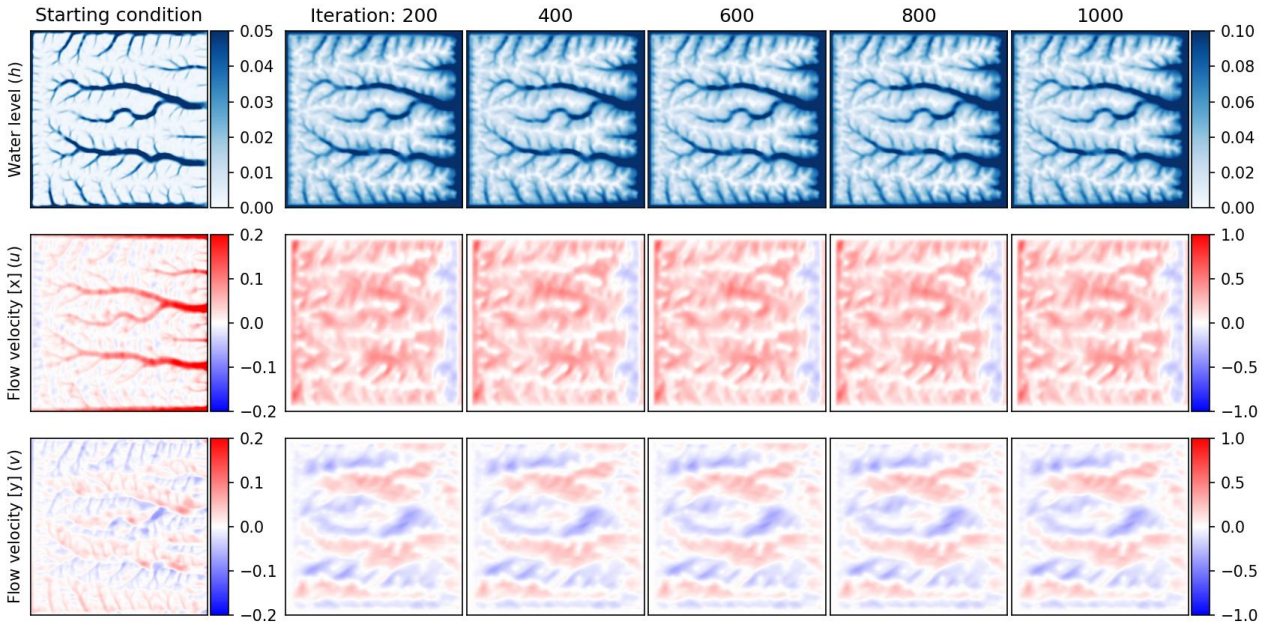
Figure A.11: Hydro-PINN trained on all 500 samples in the training set alongside a numerical solver for the morphological quantities. The PINN is evaluated on 100 samples corresponding to the third stage of landscape coarsening.



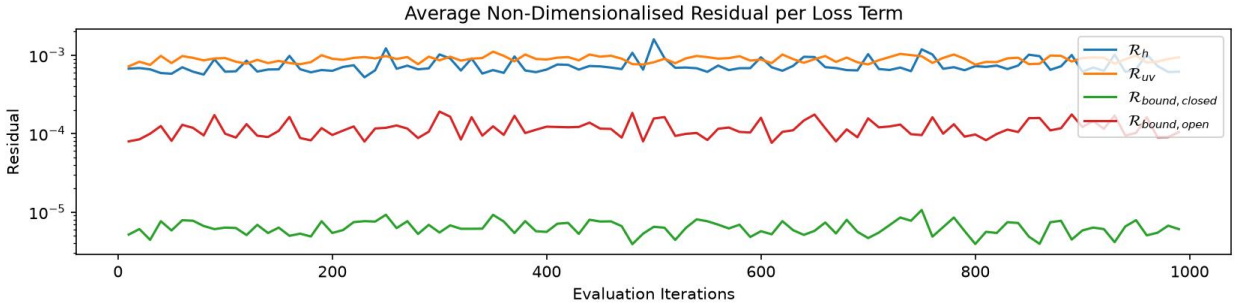
(a) Landscape structuring (Stage 4)



(b) Domain layout

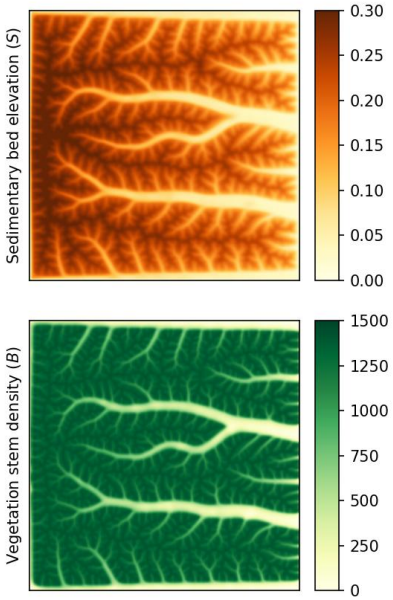


(c) Hydrodynamic solution predicted by the PINN after 200, 400, 600, 800 and 1000 iterations.

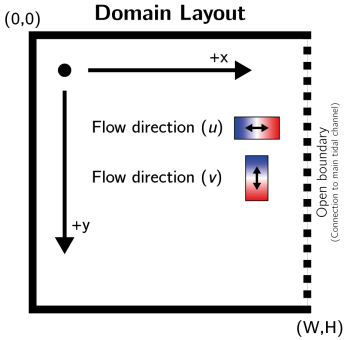


(d) Average Non-Dimensionalised Residual calculated for each term separately.

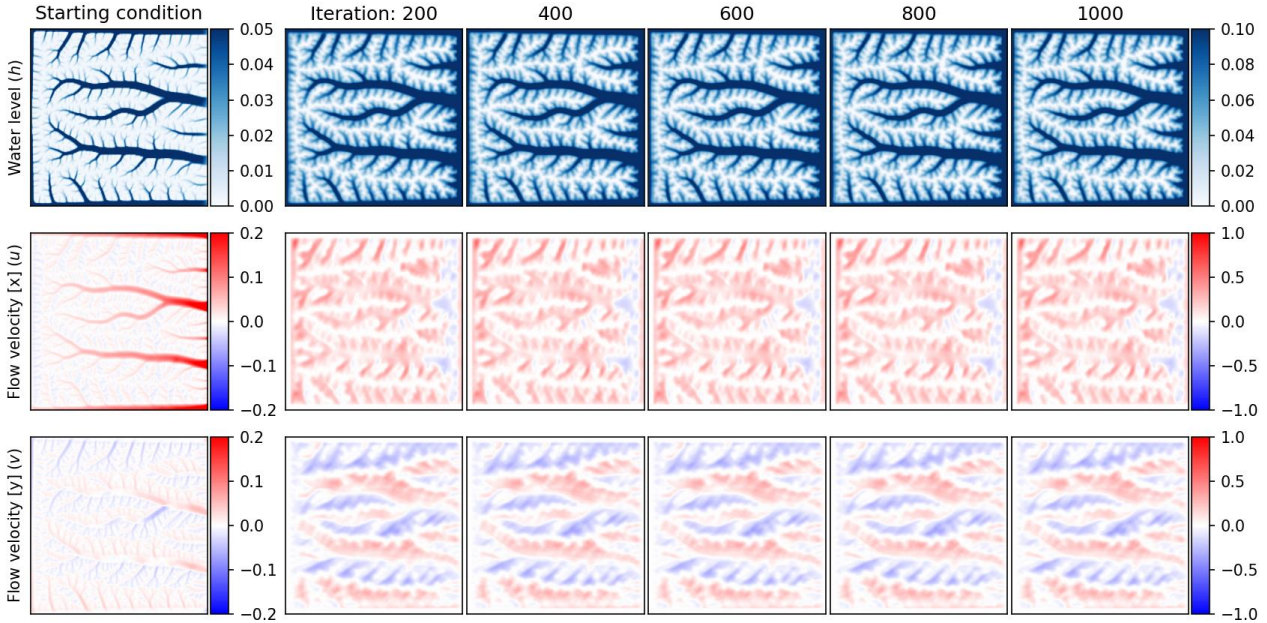
Figure A.12: Hydro-PINN trained on all 500 samples in the training set alongside a numerical solver for the morphological quantities. The PINN is evaluated on 100 samples corresponding to the fourth stage of landscape coarsening.



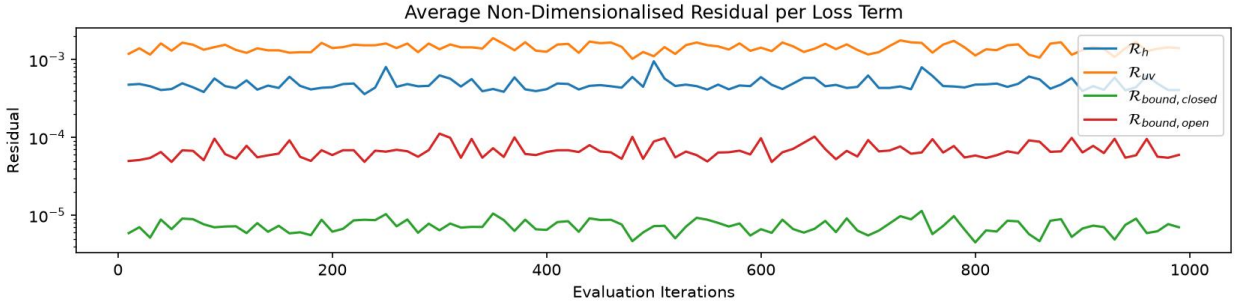
(a) Landscape structuring (Stage 5)



(b) Domain layout

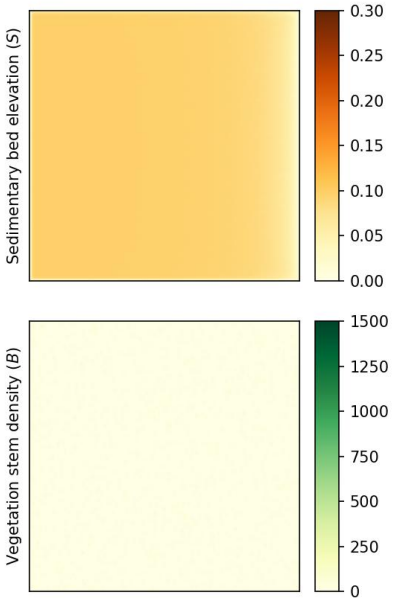


(c) Hydrodynamic solution predicted by the PINN after 200, 400, 600, 800 and 1000 iterations.

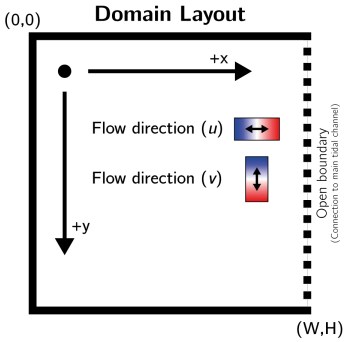


(d) Average Non-Dimensionalised Residual calculated for each term separately.

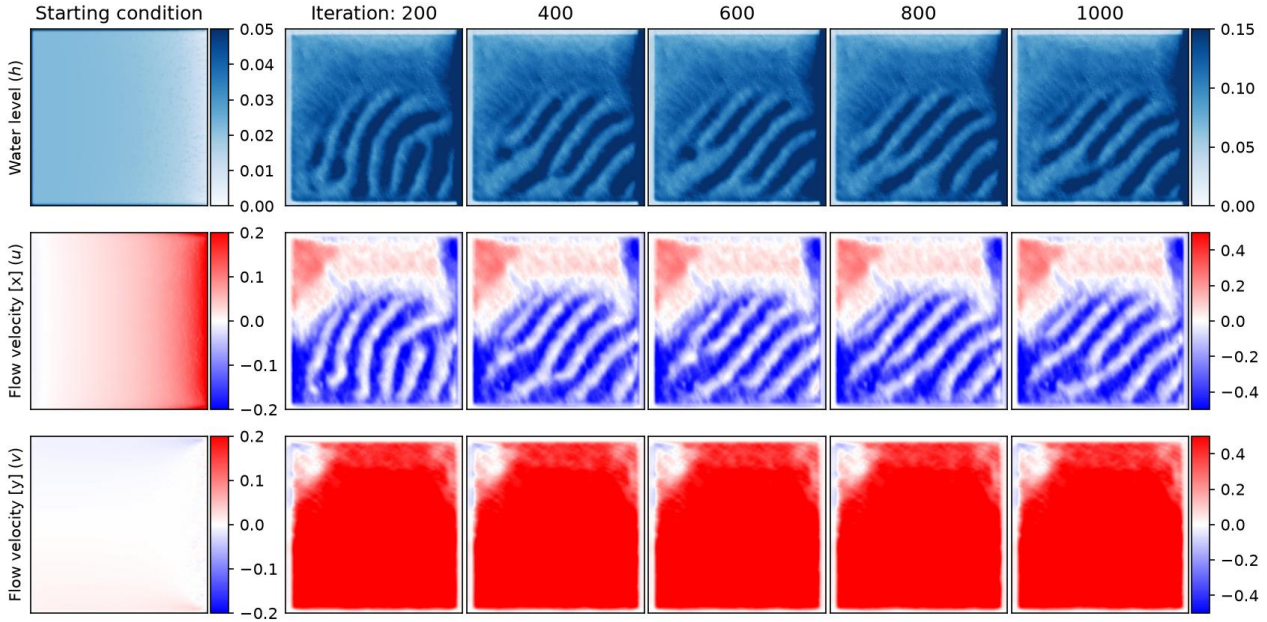
Figure A.13: Hydro-PINN trained on all 500 samples in the training set alongside a numerical solver for the morphological quantities. The PINN is evaluated on 100 samples corresponding to the fifth stage of landscape coarsening.



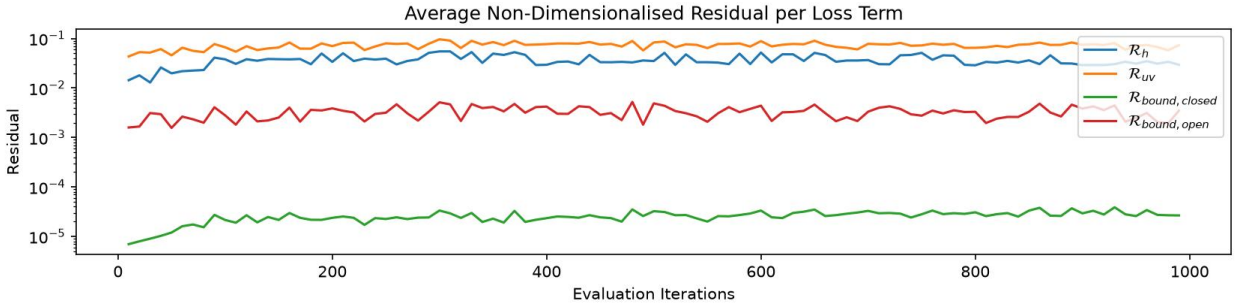
(a) Landscape structuring (Stage 1)



(b) Domain layout

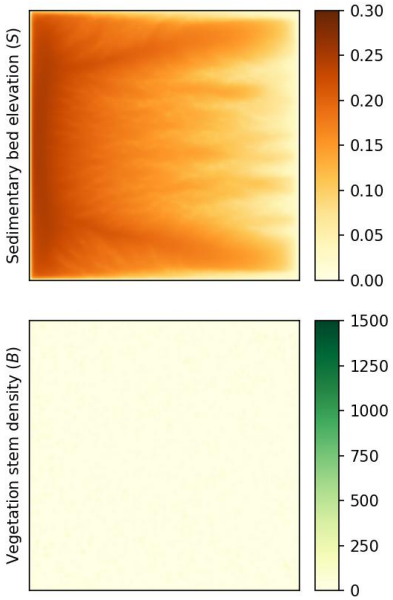


(c) Hydrodynamic solution predicted by the PINN after 200, 400, 600, 800 and 1000 iterations.

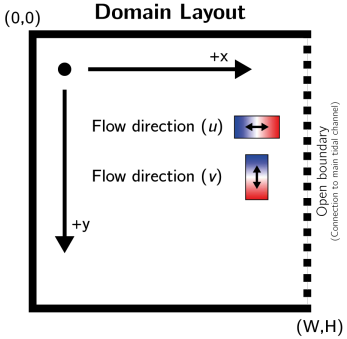


(d) Average Non-Dimensionalised Residual calculated for each term separately.

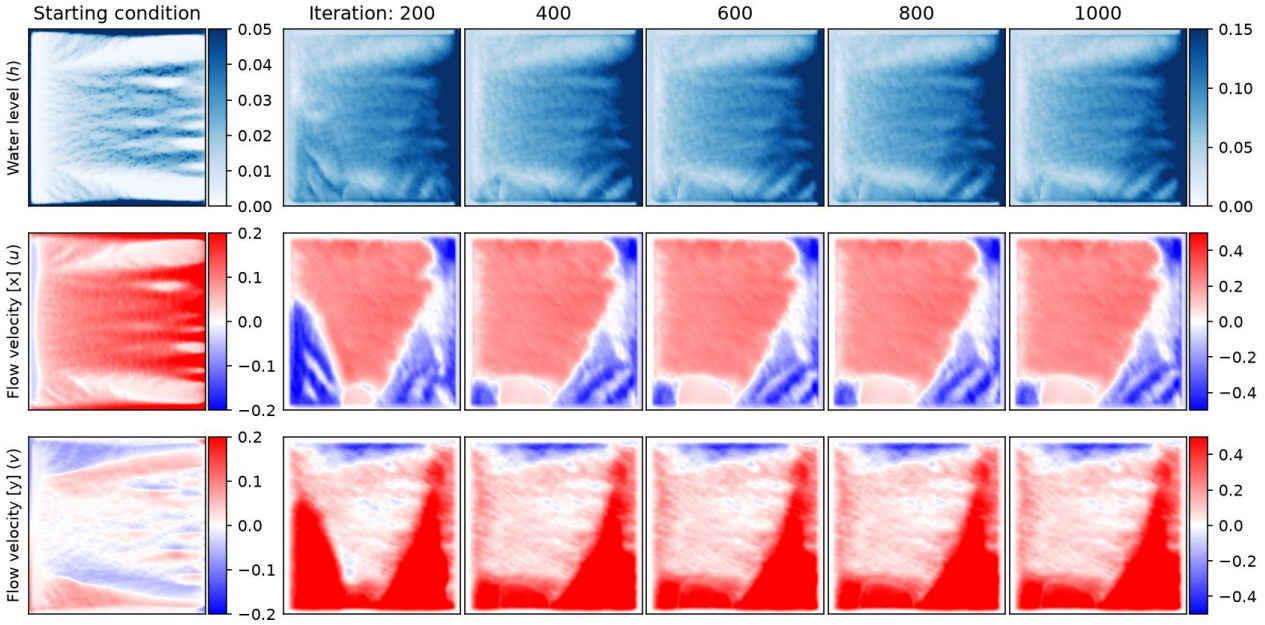
Figure A.14: Hydrodynamic component of the Saltmarsh-PINN trained on all 500 samples in the training set. The PINN is evaluated on 100 samples corresponding to the first stage of landscape coarsening.



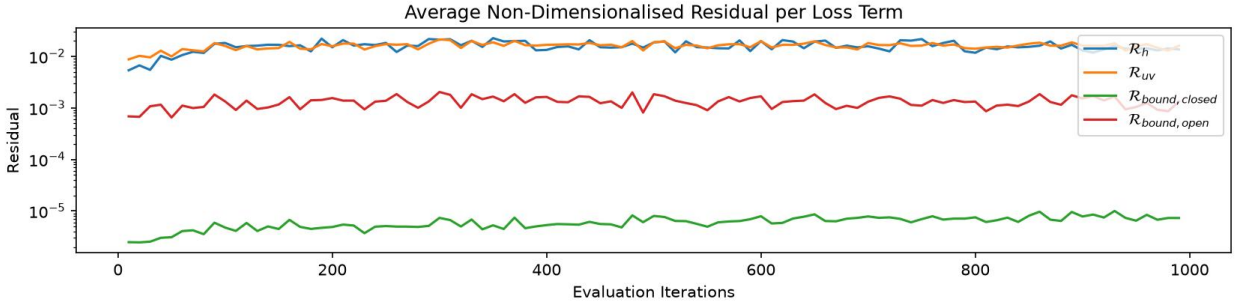
(a) Landscape structuring (Stage 2)



(b) Domain layout

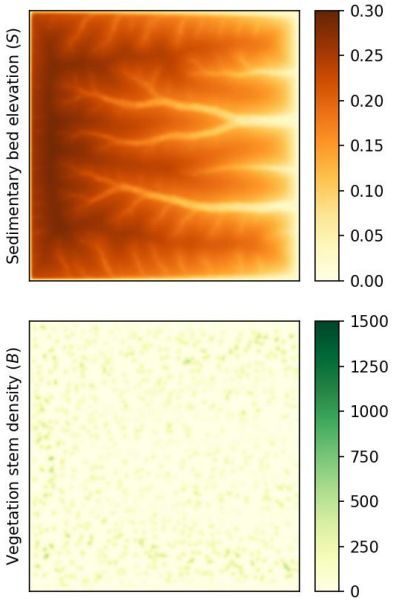


(c) Hydrodynamic solution predicted by the PINN after 200, 400, 600, 800 and 1000 iterations.

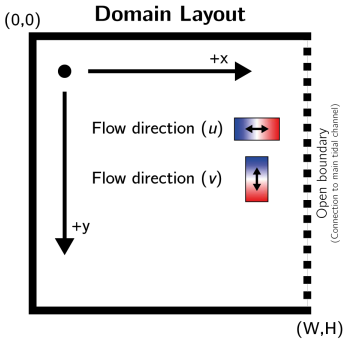


(d) Average Non-Dimensionalised Residual calculated for each term separately.

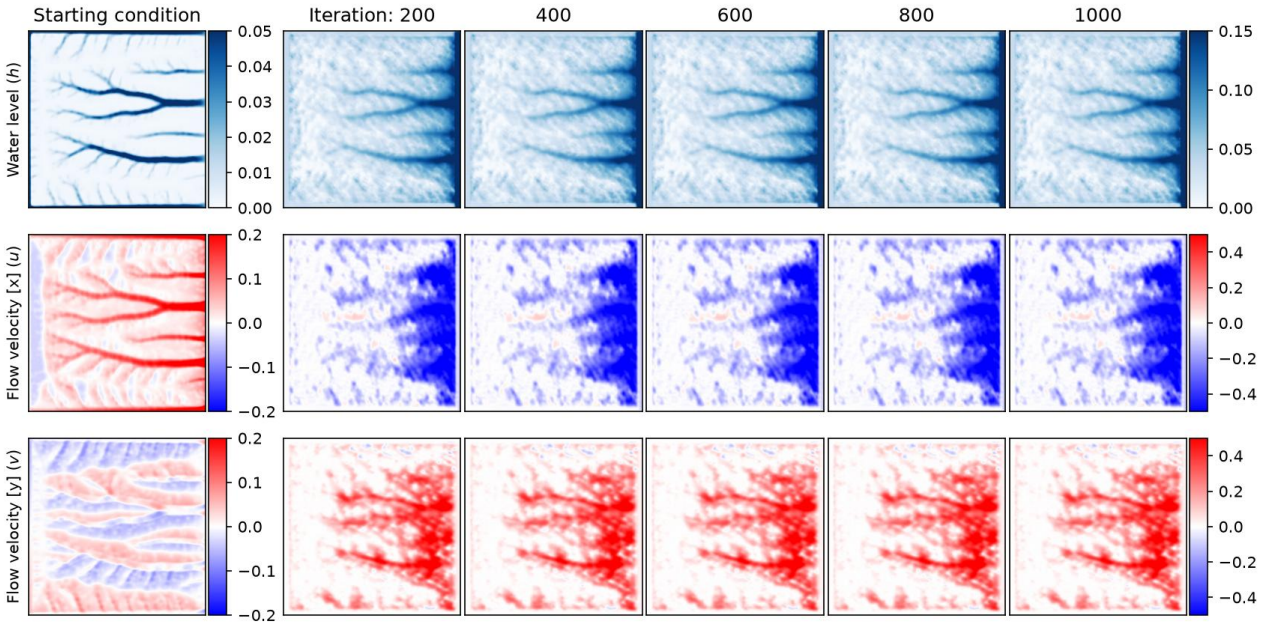
Figure A.15: Hydrodynamic component of the Saltmarsh-PINN trained on all 500 samples in the training set. The PINN is evaluated on 100 samples corresponding to the second stage of landscape coarsening.



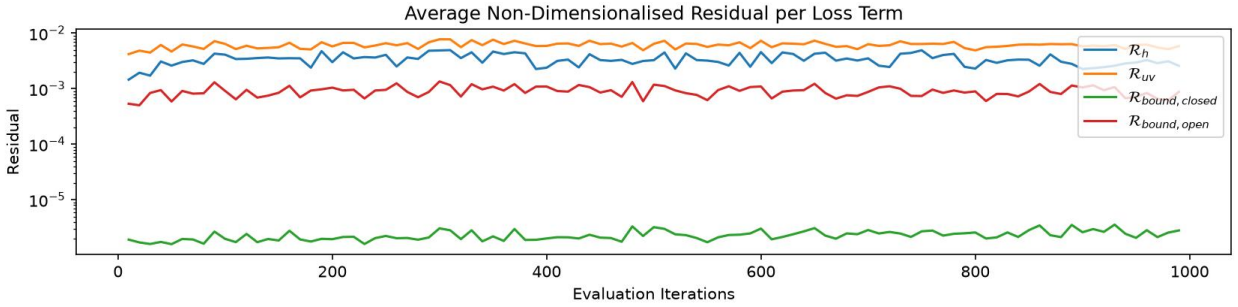
(a) Landscape structuring (Stage 3)



(b) Domain layout

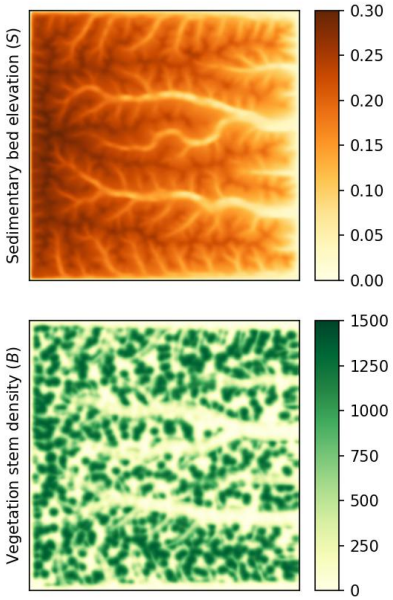


(c) Hydrodynamic solution predicted by the PINN after 200, 400, 600, 800 and 1000 iterations.

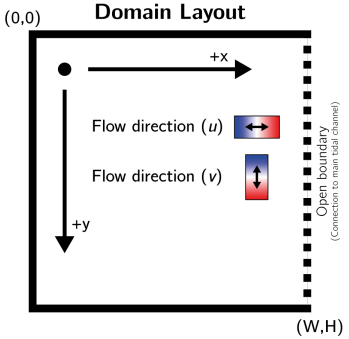


(d) Average Non-Dimensionalised Residual calculated for each term separately.

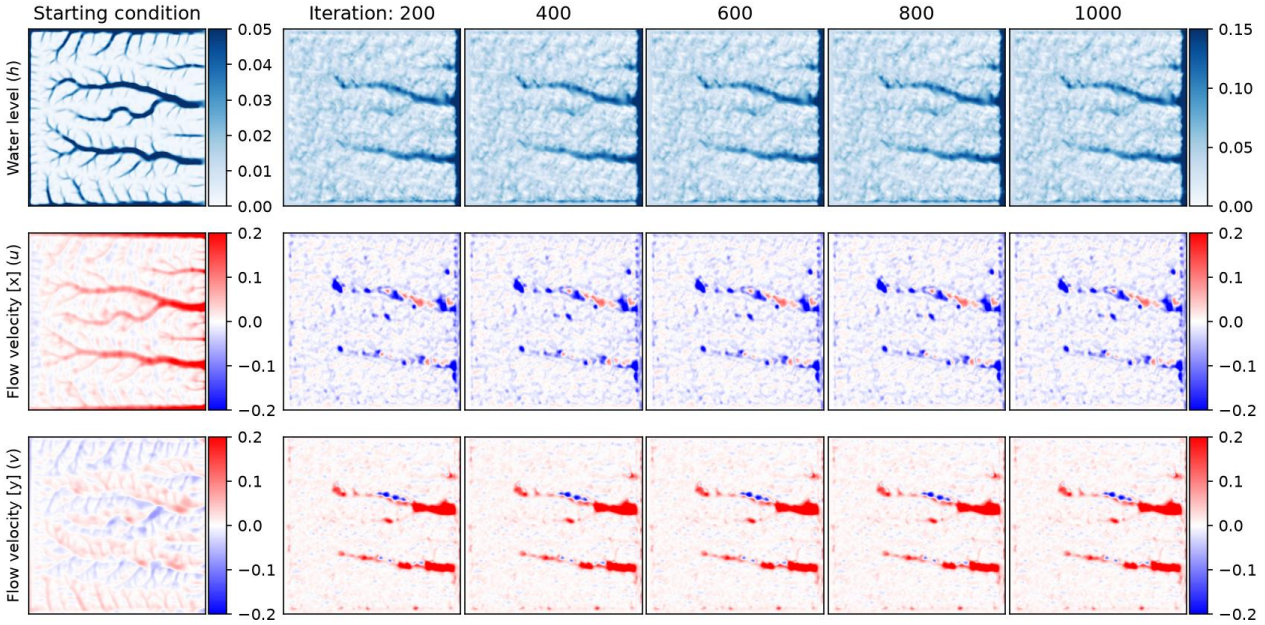
Figure A.16: Hydrodynamic component of the Saltmarsh-PINN trained on all 500 samples in the training set. The PINN is evaluated on 100 samples corresponding to the third stage of landscape coarsening.



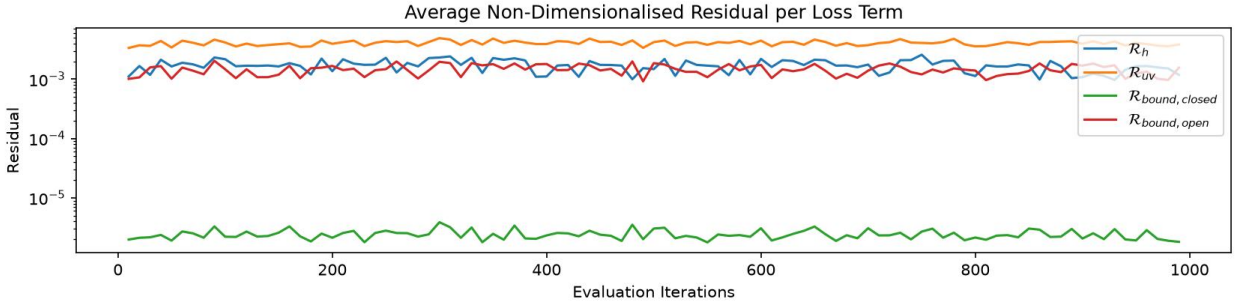
(a) Landscape structuring (Stage 4)



(b) Domain layout

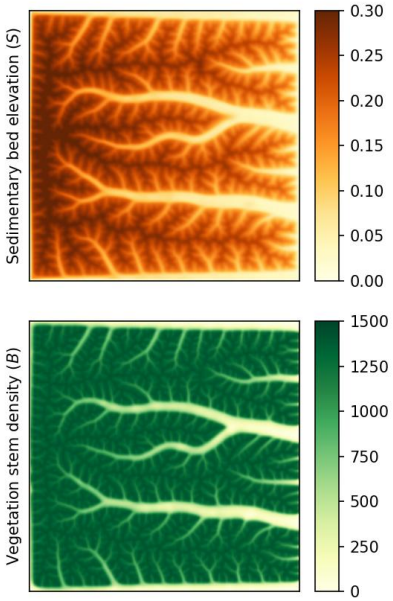


(c) Hydrodynamic solution predicted by the PINN after 200, 400, 600, 800 and 1000 iterations.

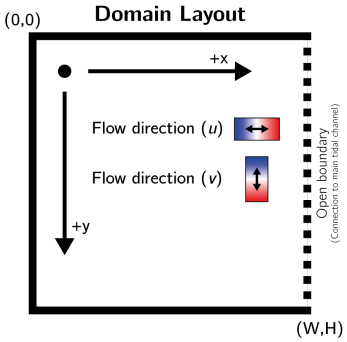


(d) Average Non-Dimensionalised Residual calculated for each term separately.

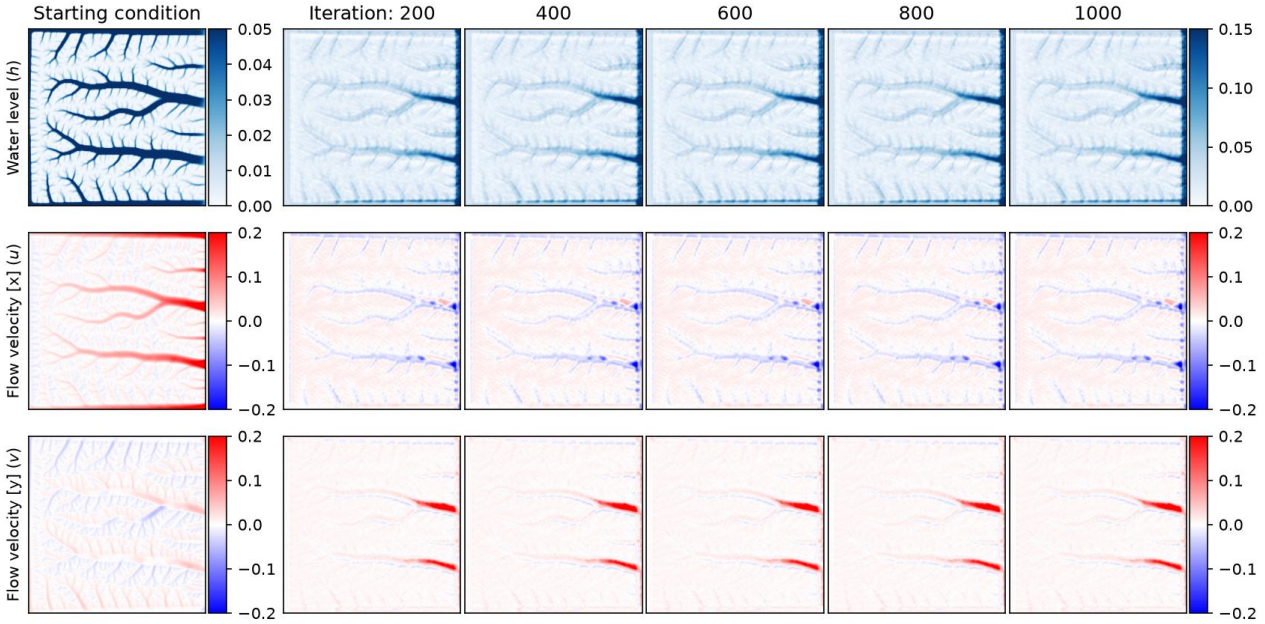
Figure A.17: Hydrodynamic component of the Saltmarsh-PINN trained on all 500 samples in the training set. The PINN is evaluated on 100 samples corresponding to the fourth stage of landscape coarsening.



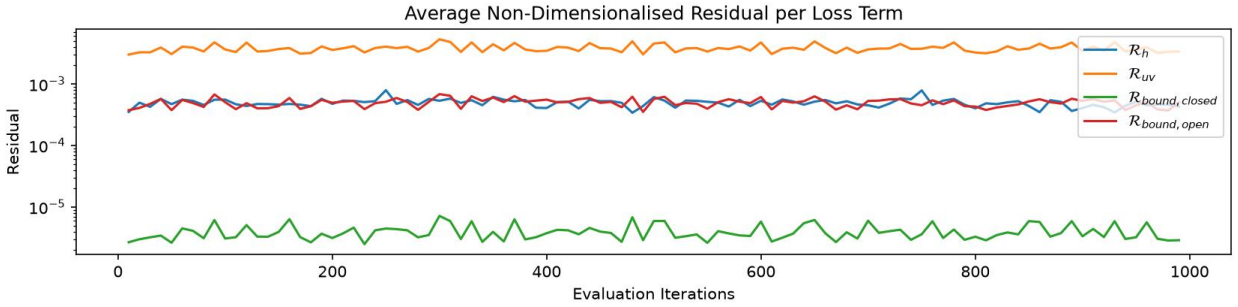
(a) Landscape structuring (Stage 5)



(b) Domain layout



(c) Hydrodynamic solution predicted by the PINN after 200, 400, 600, 800 and 1000 iterations.



(d) Average Non-Dimensionalised Residual calculated for each term separately.

Figure A.18: Hydrodynamic component of the Saltmarsh-PINN trained on all 500 samples in the training set. The PINN is evaluated on 100 samples corresponding to the fifth stage of landscape coarsening.

A.5. Hybrid Morphodynamics

In Section 6.4.1, the Hydro-PINN is trained in hybrid with a numerical morphological solver and several figures are presented to illustrate the type of morphodynamics that result from that simulation. This appendix section provides additional figures for those who want a complete picture of the temporal evolution of the landscapes. For that, Figure A.19 corresponds to the results presented in Figure 6.30, Figure A.20 corresponds to Figure 6.31 and Figure A.21 corresponds to Figure 6.32.

A.6. Saltmarsh-PINN Morphodynamics

In Section 6.4.2, the Saltmarsh-PINN is evaluated on two landscape conditions. First, we presented an evaluation of the PINN on a stage-1 landscape in Figure 6.34. Second, we presented an evaluation of the PINN on a stage-5 landscape in Figure 6.35. This appendix includes more a more detailed overview of those results, in Figures A.23 and A.27, respectively. Additionally, we present figures for the intermediate landscape stages as well, in Figures A.24, A.25 and A.26. Figure A.22 presents the predictions made by the Saltmarsh-PINN on the starting conditions used by SFERE [66].

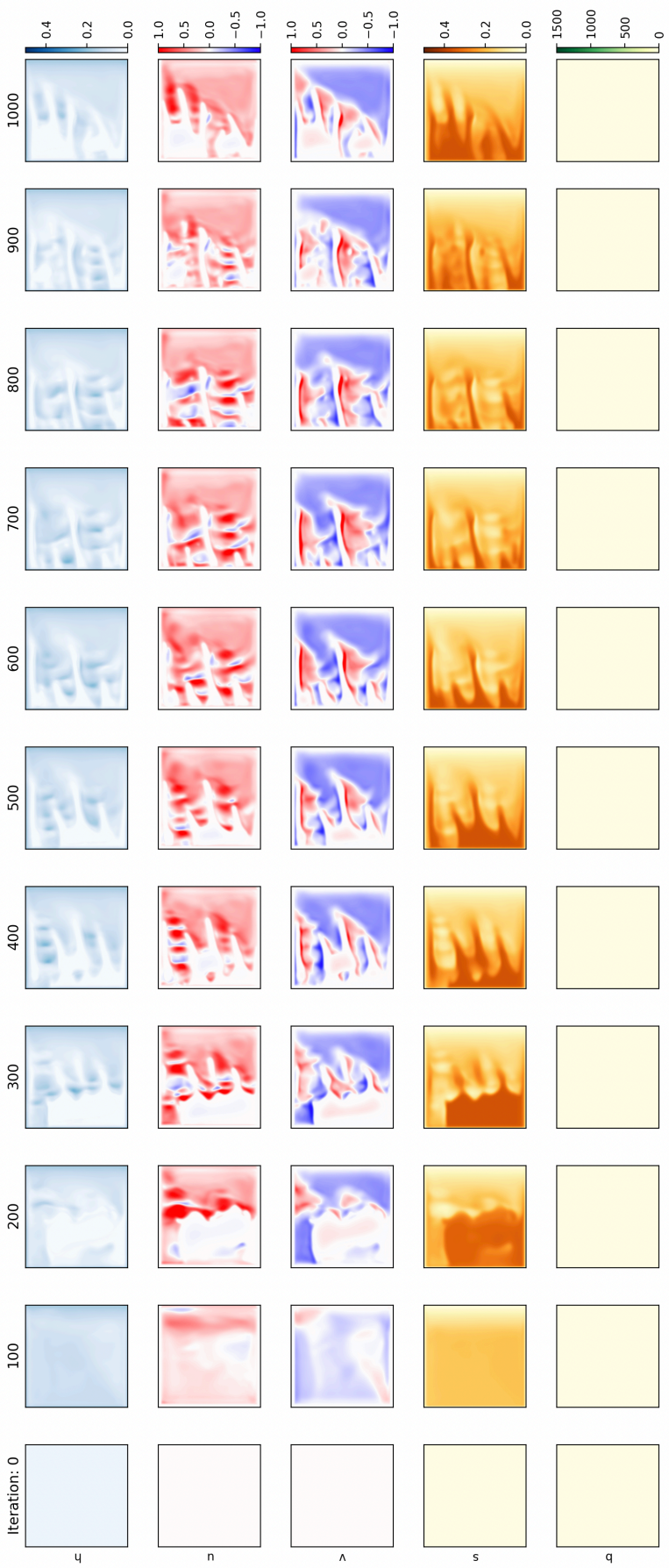


Figure A.19: The hybrid Hydro-PINN is evaluated on a flat landscape with no vegetation, for 1000 iterations.

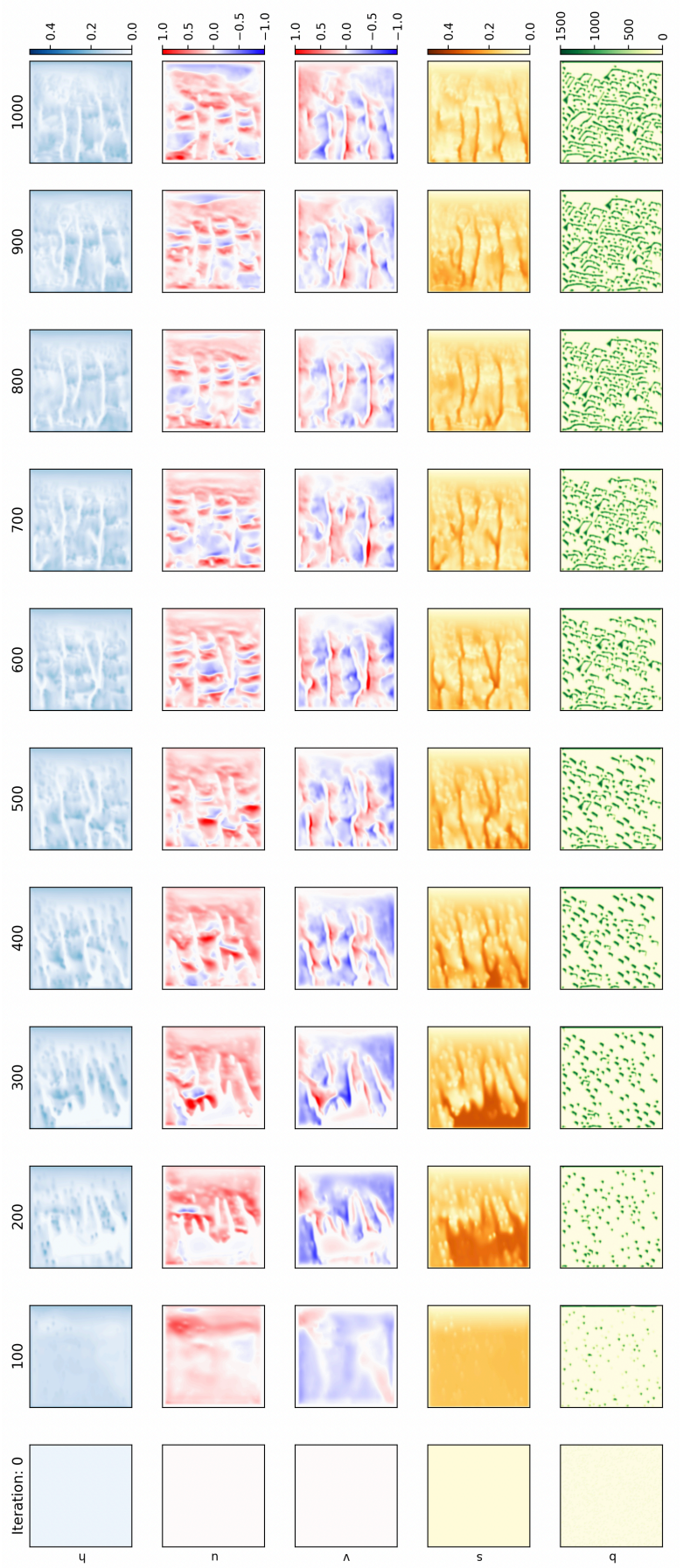


Figure A.20: The hybrid Hydro-PINN is evaluated on a flat landscape with randomly placed vegetation tussocks, for 1000 iterations.

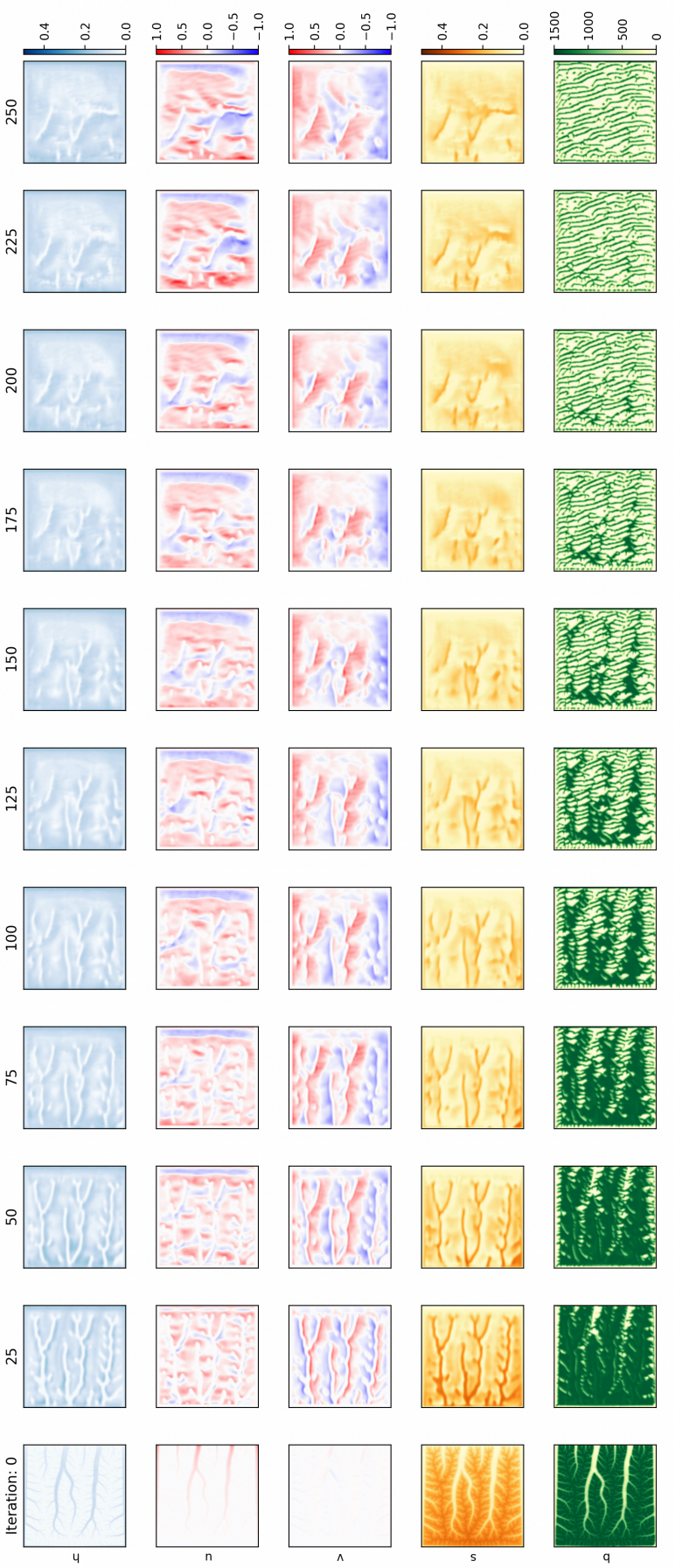


Figure A.21: The hybrid Hydro-PINN is evaluated on a fully developed salt marsh with channelisation at multiple scales, for 250 iterations.

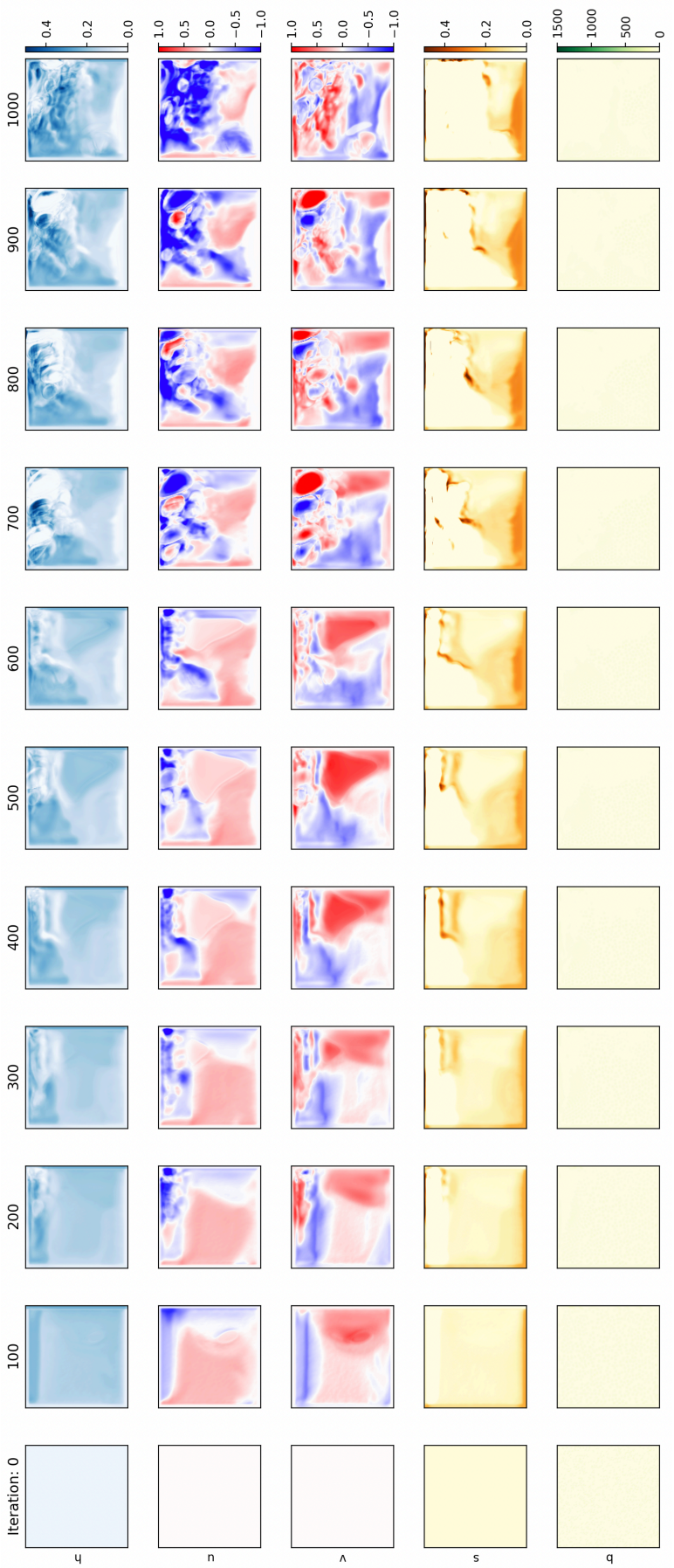


Figure A.22: Saltmarsh-PINN evaluated on a flat bottom topography with no vegetation in the landscape. The left-most column indicates the starting condition, followed by the solution for all five quantities in steps of 100 iterations until iteration 1000.

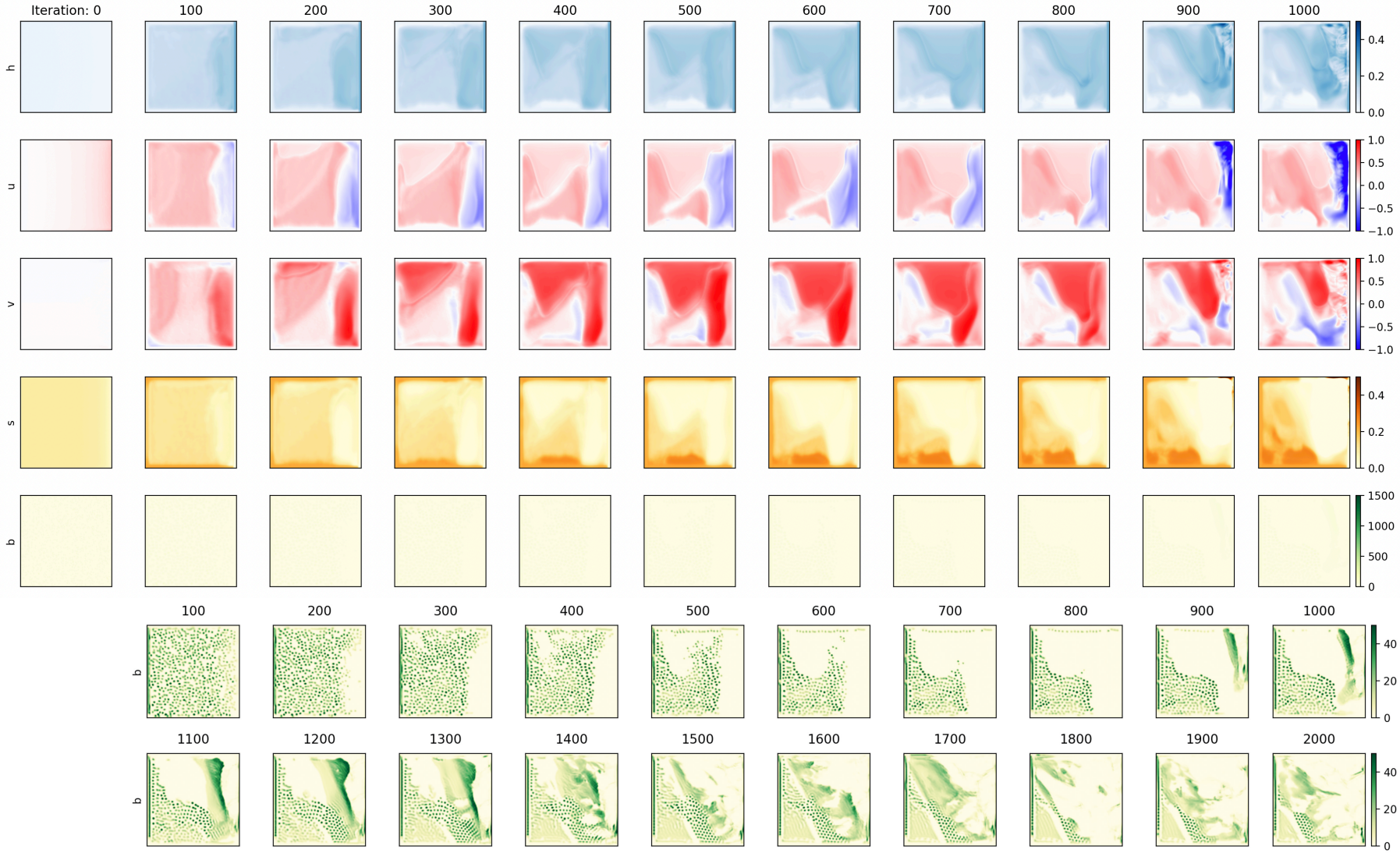


Figure A.23: Saltmarsh-PINN evaluated on the first category of landscape coarsening in Figure 4.5a. The left-most column indicates the starting condition, followed by the solution for all five quantities in steps of 100 iterations until iteration 1000. The bottom two rows contain a duplicate of the vegetation predictions on a narrower scale to reveal small scale patterns, where the bottom row further extends the vegetation simulation from 1000 to 2000 iterations.

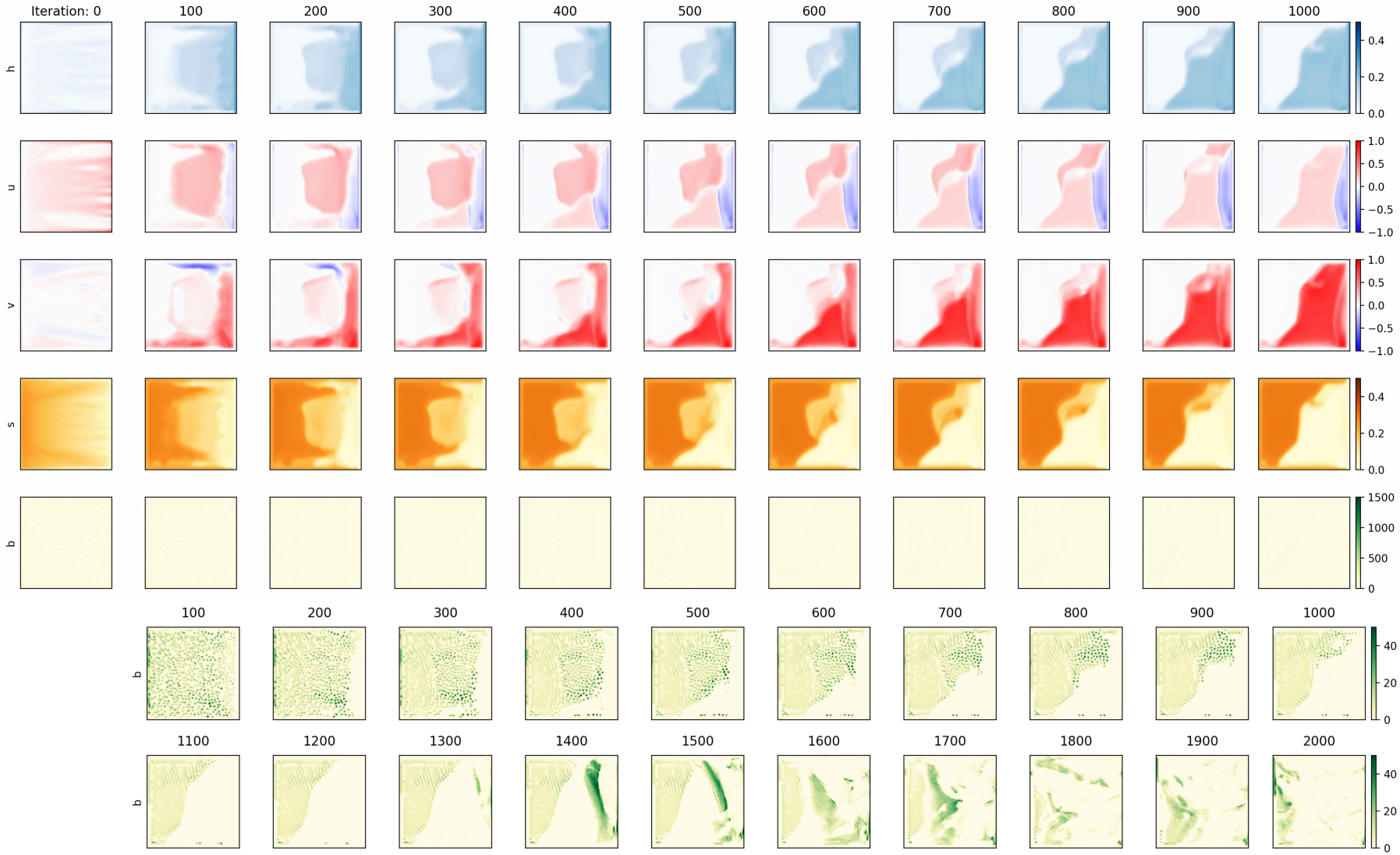


Figure A.24: Saltmarsh-PINN evaluated on the first category of landscape coarsening in Figure 4.5b. The left-most column indicates the starting condition, followed by the solution for all five quantities in steps of 100 iterations until iteration 1000. The bottom two rows contain a duplicate of the vegetation predictions on a narrower scale to reveal small scale patterns, where the bottom row further extends the vegetation simulation from 1000 to 2000 iterations.

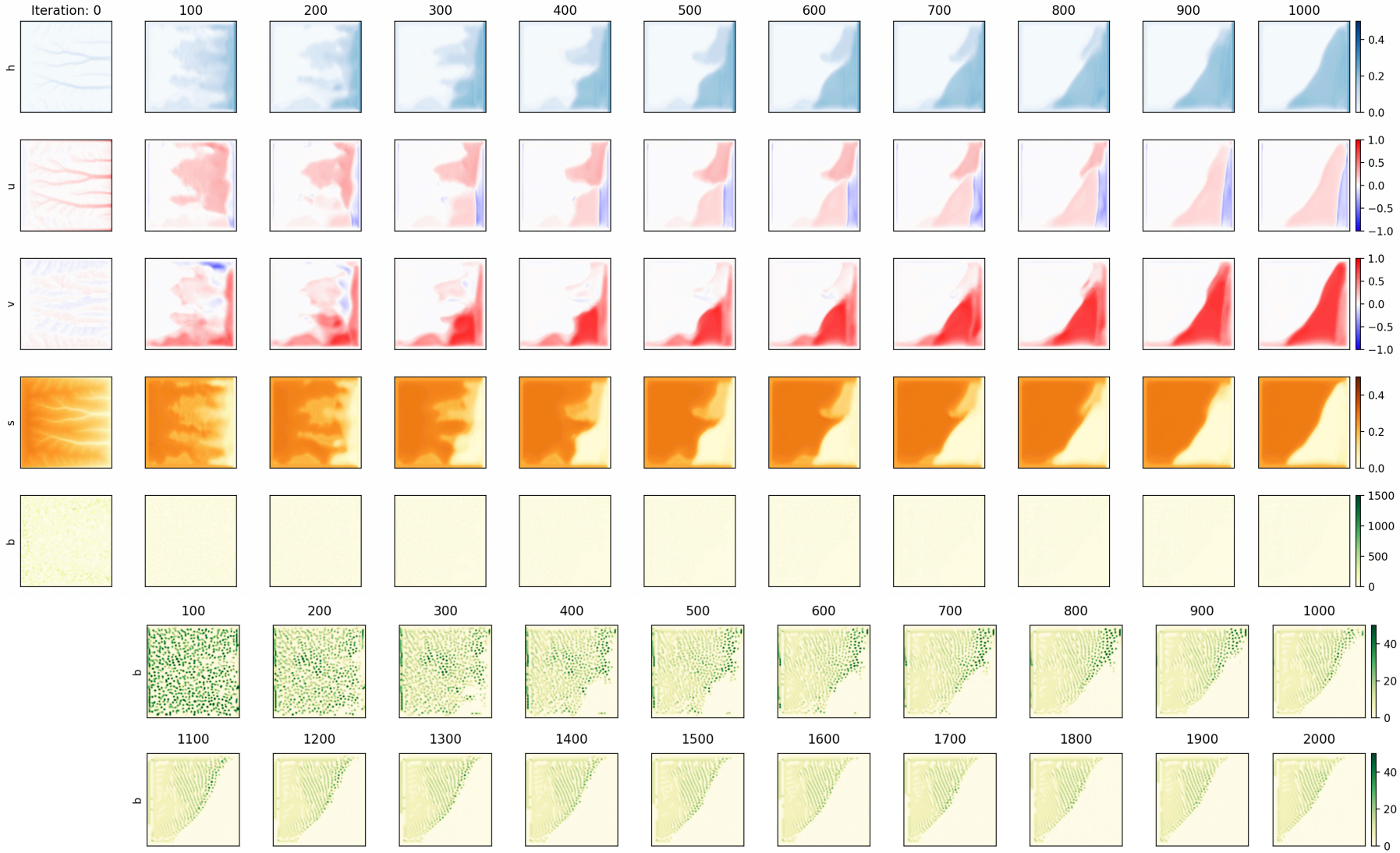


Figure A.25: Saltmarsh-PINN evaluated on the first category of landscape coarsening in Figure 4.5c. The left-most column indicates the starting condition, followed by the solution for all five quantities in steps of 100 iterations until iteration 1000. The bottom two rows contain a duplicate of the vegetation predictions on a narrower scale to reveal small scale patterns, where the bottom row further extends the vegetation simulation from 1000 to 2000 iterations.

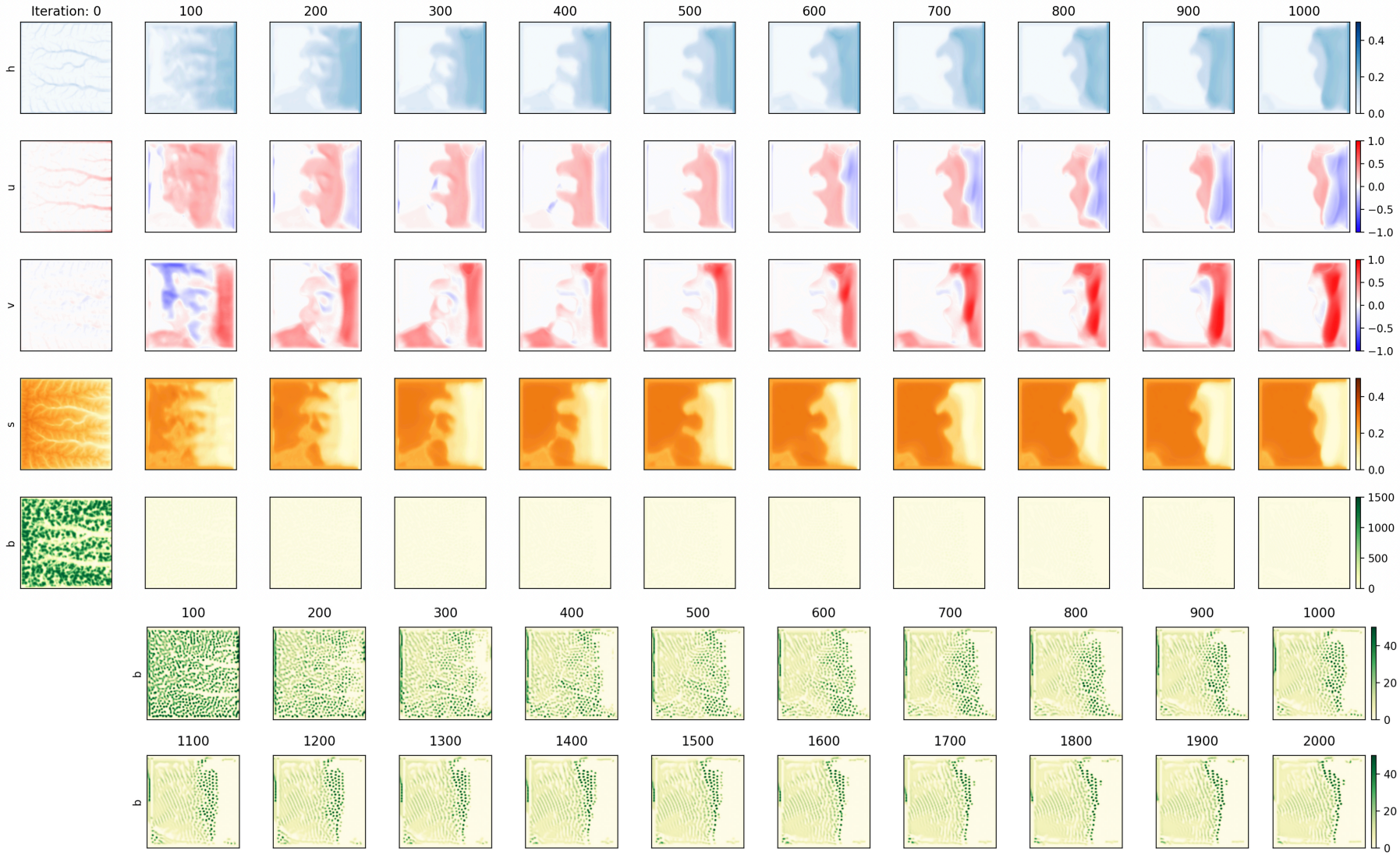


Figure A.26: Saltmarsh-PINN evaluated on the first category of landscape coarsening in Figure 4.5d. The left-most column indicates the starting condition, followed by the solution for all five quantities in steps of 100 iterations until iteration 1000. The bottom two rows contain a duplicate of the vegetation predictions on a narrower scale to reveal small scale patterns, where the bottom row further extends the vegetation simulation from 1000 to 2000 iterations.

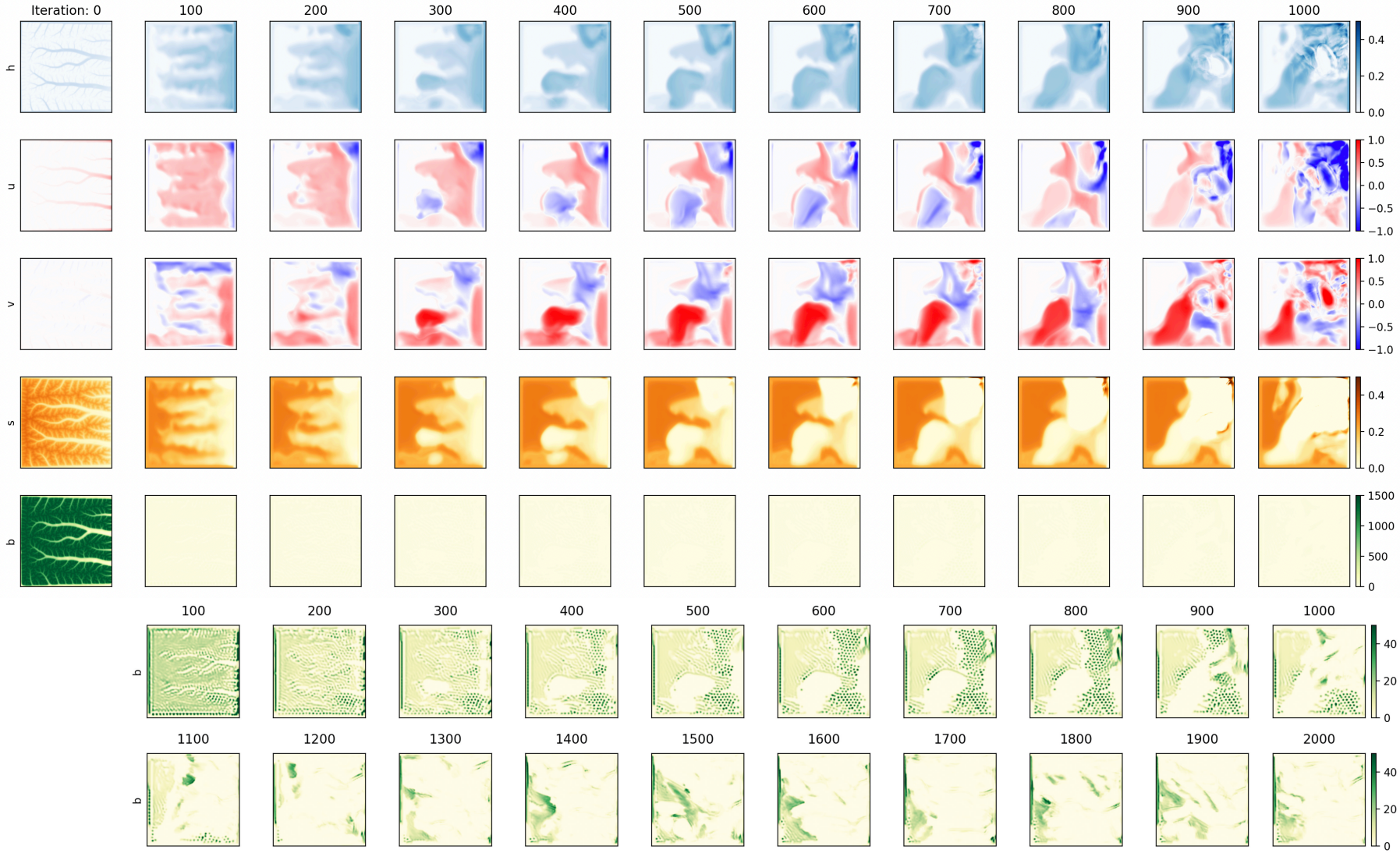


Figure A.27: Saltmarsh-PINN evaluated on the first category of landscape coarsening in Figure 4.5e. The left-most column indicates the starting condition, followed by the solution for all five quantities in steps of 100 iterations until iteration 1000. The bottom two rows contain a duplicate of the vegetation predictions on a narrower scale to reveal small scale patterns, where the bottom row further extends the vegetation simulation from 1000 to 2000 iterations.

A.7. Automatic Random Domain Rotation

To remove as much spatial asymmetry as possible during training, I've implemented automatic random rotation of the domains. This can now be done without re-running the numerical simulation, by simply rotating the spline latent representations directly. Figure A.28 shows an illustration of the four different orientations. Automatic rotation of spline latent space images is quite straightforward and simply requires rotating the spline weights 90, 180 and 270 degrees to obtain the North, West and South orientations from the default East orientation. However, special attention is required when rotating flow velocity components u and v. After rotating, we must swap the u and v channels to obtain correct North and South orientations. In addition, after rotating and swapping, we must multiply u by -1 when orienting towards the West or South and multiply v by -1 when orienting towards the North or West. This can be done by directly multiplying their Spline weights by -1, as the final interpolated value at any given point in the field is a linear combination of these weights multiplied by their respective spline-kernel functions.

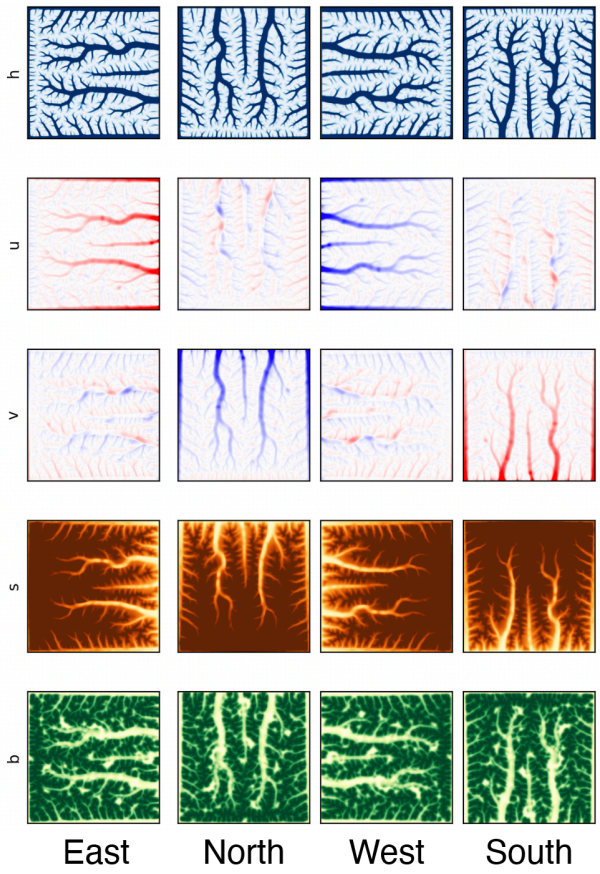


Figure A.28: Tidal channel formation (sampled from Spline latent space) after 650.000 iterations of the numerical simulation, automatically rotated to face East, North, West and South, indicating the position of the connection to the main tidal channel with respect to the intertidal area.

A.8. PINN Learns Principal Component First

Up until this point, I've always merged the loss components $\mathcal{L}_u$ and $\mathcal{L}_v$ into a single learning objective

$$\mathcal{L}_{\text{momentum}} = \mathcal{L}_u + \mathcal{L}_v$$

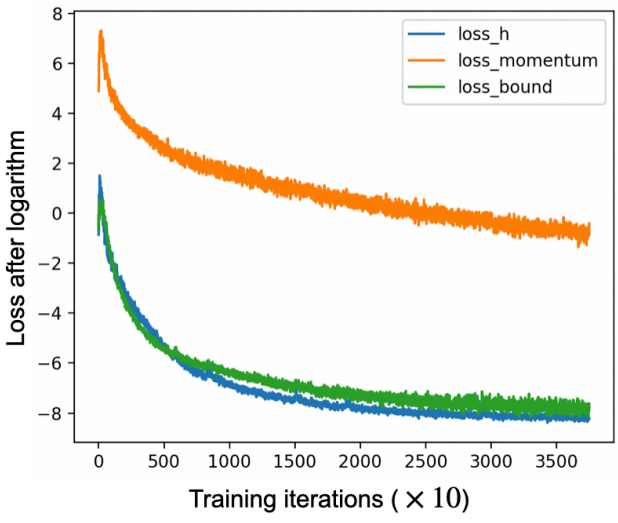
where the loss terms $\mathcal{L}_u$ and $\mathcal{L}_v$ would be separate log-loss terms:

$$\mathcal{L}_u = \log(\beta L_u + \epsilon)$$
$$\mathcal{L}_v = \log(\beta L_v + \epsilon).$$

where β acts as an optional loss weight and ϵ is a small bias to prevent landing in the discontinuity of $\log$. Because we sum *after* taking the logarithm, the momentum loss term would converge to much smaller values than the continuity and boundary loss terms $\mathcal{L}_h$ and $\mathcal{L}_{\text{bound}}$, making it harder to visualise convergence. Therefore, I instead opted for an alternative merged loss-term, where the order of summing and log is swapped:

$$\mathcal{L}_{\text{momentum}} = \log(\beta(L_u + L_v) + \epsilon)$$

so that all three loss terms $\mathcal{L}_{\text{momentum}}$, $\mathcal{L}_h$ and $\mathcal{L}_{\text{bound}}$ would ideally converge to the same zero-loss value. After training for almost 40 epochs, the training loss had not yet converged, and the momentum loss is larger than that of the other terms. This can be seen in Figure A.29a. The resulting shallow water simulation in Figure A.29b pictures waves traveling only along the diagonal of the domain. This could indicate that the principle component $[(+u) \times (+v)]$ of the flow velocity vector has first been learned by the model before $[(+u) \times (-v)]$ or $[(-u) \times (+v)]$.



(a) Training loss over time



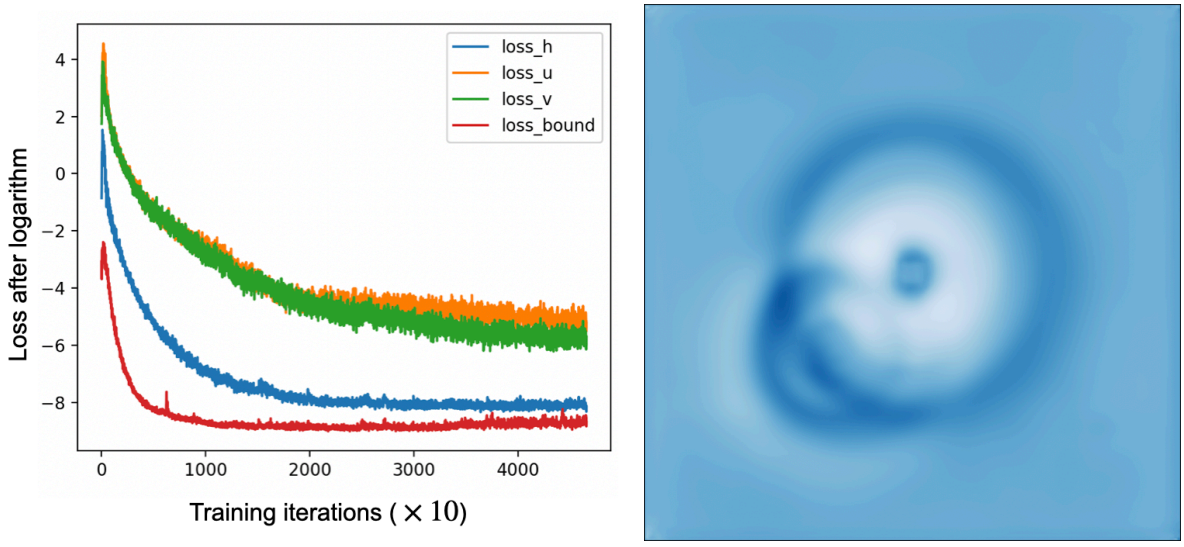
(b) Waves introduced by the oscillator in the centre propagate only along the diagonal.

Figure A.29: Learning shallow water equations in a closed water basin with momentum loss represented by a merged term $\mathcal{L}_{\text{momentum}} = \log(\beta(L_u + L_v) + \epsilon)$.

When we instead treat the momentum components as separate loss terms

$$\mathcal{L}_u = \log(\beta L_u + \epsilon)$$
$$\mathcal{L}_v = \log(\beta L_v + \epsilon).$$

and perform gradient surgery on each of the terms individually, this effect disappears and the wave starts to propagate evenly in all directions again. This can be seen in Figure A.30b. Furthermore, training seems to be more efficient, as Figure A.30a indicates a lower overall loss for the individual momentum terms compared to their grouped loss in Figure A.29a. Training has likely not fully converged yet, however, judging by the secondary wave-artifact that can be seen in Figure A.30b.



(a) Training loss over time

(b) Waves introduced by the oscillator in the centre propagate evenly in all directions.

Figure A.30: Learning shallow water equations in a closed water basin with momentum loss represented by separate momentum loss terms $\mathcal{L}_u = \log(\beta L_u + \epsilon)$ and $\mathcal{L}_v = \log(\beta L_v + \epsilon)$.

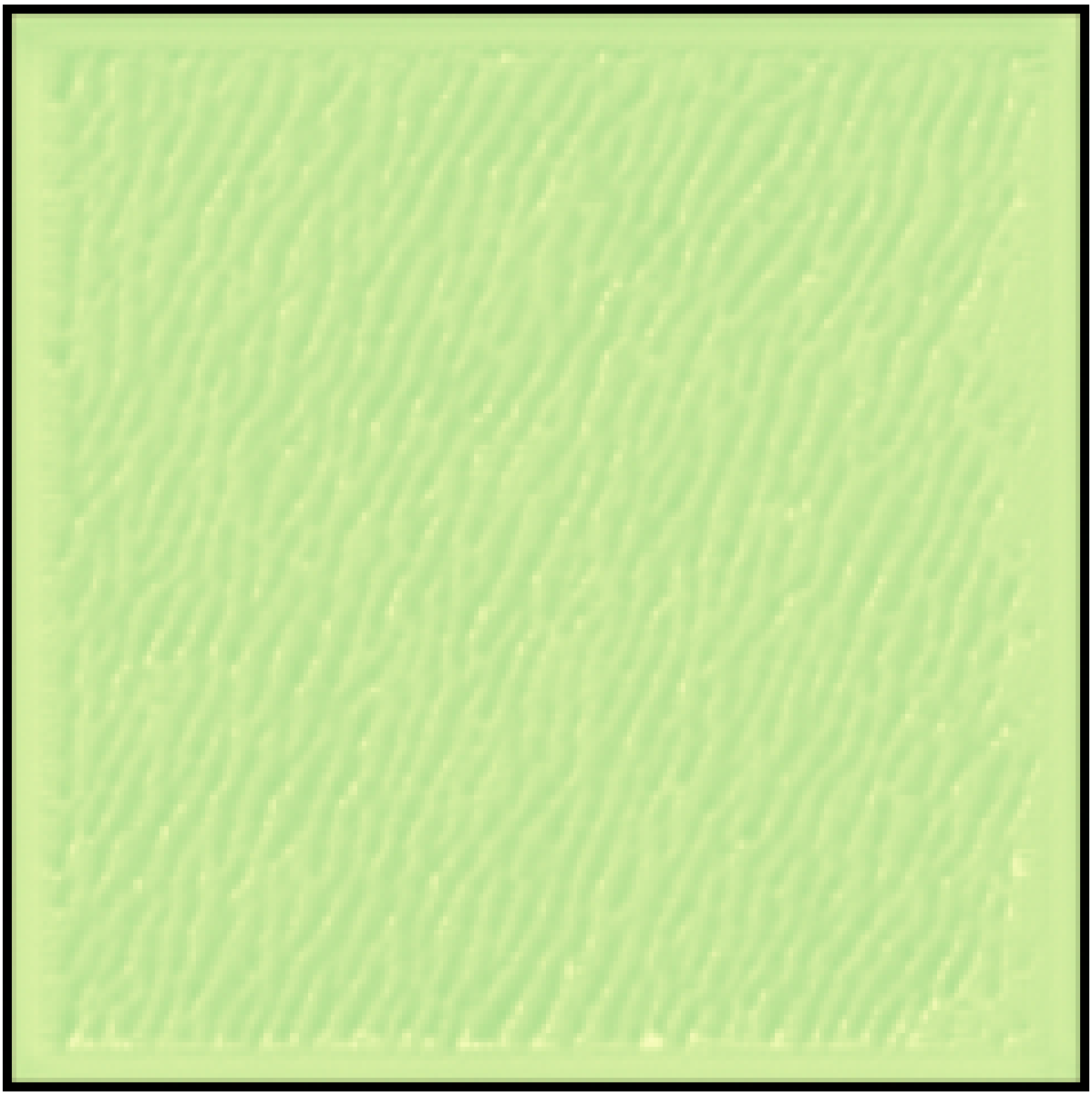


Figure A.31: Single scale Turing patterns appear in the vegetation after starting with a random noise starting condition.