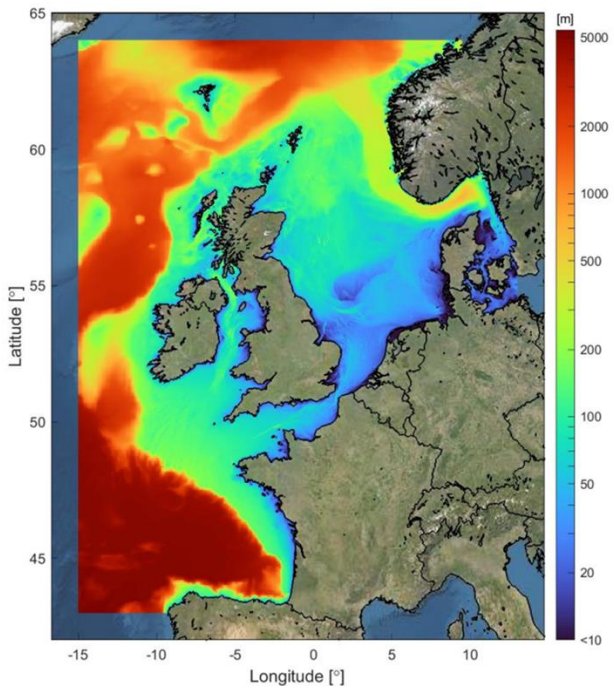
AI-Based Correction of Wind Forcing Bias in Surrogate Storm Surge Models

Sara Vélez Fuente (6182399)

November 30, 2025



Deltares (HAF Department) & Delft University of Technology

Supervised by: Prof. Dr. Martin Verlaan, Dr. Jing Zhao, Dr. Willem Tromp

Abstract

Storm-surge forecasting in the Netherlands relies on computationally expensive numerical models that hinder rapid ensemble predictions critical for operational decision-making. To address this, machine-learning surrogates offer significant speed-ups but inherit systematic biases from their training data. This work investigates whether small neural networks can learn to correct wind stress underestimation, a major bias in atmospheric forcing, by training on observations and surrogate predictions. The approach is validated on two test models: the Burgers' equation with controlled 20% input bias, and a one-dimensional shallow-water equations with realistic 21% wind-stress bias. Results demonstrate that per-station corrections can reduce errors by 50 - 78 % in multi-step rollouts while maintaining stability over 6 to 12 hour horizons. However, correction effectiveness depends critically on surrogate quality, and purely global corrections fail due to spatial heterogeneity in wind-stress sensitivity. The findings suggest that learned corrections are a viable preprocessing step for operational systems but require careful surrogate design and domain expertise for robust deployment.

Chapter 1

Introduction

Coastal Flood Risk and Storm-Surge Forecasting in the Netherlands

The Netherlands is constantly exposed to coastal flooding from the North Sea, a risk that has shaped its history and infrastructure for centuries. Current climate forecasts indicate that this threat is intensifying, as rising sea levels elevate water heights and possible variations in storm intensity increase surge magnitudes during extreme weather events. Within this context, rapid and accurate storm-surge forecasting is essential. Detailed predictions provide decision-makers, such as Rijkswaterstaat (the Dutch directorate responsible for water management), emergency managers, and local authorities, with essential lead time to implement protective measures, issue evacuation orders, and minimise the loss of life and economic damage.

The significance of this work extends beyond coastal defence, addressing broader issues of equity and resilience. Vulnerable populations living in low-lying coastal areas, often overlapping with economically disadvantaged communities, bear disproportionate risk from coastal flooding. Robust and accurate forecasting is essential for protecting communities, while failures or false alarms can deteriorate public trust in emergency systems and cause unnecessary disruption and anxiety. The stakes of this research, therefore, are not merely technical but fundamentally human.

Numerical Modelling of Storm Surges: The DCSM-FM and Computational Constraints

Deltares has developed the Dutch Continental Shelf Model–Flexible Mesh (DCSM-FM), a state-of-the-art hydrodynamic simulation tool built on Delft3D, which solves the shallow-water equations to predict water levels, flow velocities, and other hydrodynamic variables under different meteorological forcing [2, 4]. The model has been developed to high fidelity and provides predictions that closely align with the observations across most scenarios. However, operational storm-surge modelling increasingly relies on ensemble forecasting, a practice in which many simulations are run with perturbed initial conditions and forcing fields to quantify forecast uncertainty and improve reliability. An ensemble consisting of $50 - 100$ members, each requiring several hours of computation on modern hardware, becomes impractical for operational timelines that demand predictions within a few hours [2, 4]. Although the specific numbers may vary based on configuration and available computational resources, the core issue remains: traditional models cannot process enough ensemble members quickly enough for real-time decision-making.

This computational bottleneck motivates a search for faster alternatives. Thus, a promising approach is to develop an AI surrogate, a machine learning model trained to approximate the behaviour of the expensive numerical solver at a fraction of the computational cost. Building on the previous results of these models on weather forecasting and computational fluid dynamics, Deltares is developing a graph neural network (GNN)-based surrogate of the DCSM-FM capable of simulating its predictions rapidly enough to facilitate ensemble forecasting. Preliminary results show significant speedups over the original solver; nevertheless, several important questions remain regarding accuracy, long-term stability, and the proper treatment of systematic model errors.

The Problem of Systematic Input Bias in Operational Models

While AI surrogates offer substantial computational speed-ups, they also inherit the behaviour of the models on which they are trained, including any systematic biases. In operational storm-surge forecasting, a prominent example is the underestimation of wind stress. Because wind stress is derived from probabilistic atmospheric forecasts, which are themselves subject to systematic error, it directly influences the predicted surge magnitude. If the meteorological forcing is systematically underestimated, the resulting surge prediction will be systematically too weak, regardless of how well the surrogate mimics the underlying hydrodynamic model. In other words, surrogates reproduce the biased inputs they see. By contrast, traditional models can be corrected through data assimilation, but their computational cost often makes them impractical for time-critical applications.

Traditional calibration approaches, such as four-dimensional variational assimilation or ensemble Kalman filtering, can correct for such biases. Still, they require running the expensive forward model multiple times and rely on adjoint computations. An alternative strategy is to learn a correction module, a small neural network that learns the missing correction to the forcing based on model observations. If such a network is lightweight, differentiable, and fast to evaluate, it can be integrated into the workflow as a preprocessing step or used alongside a surrogate. This approach can enhance both speed and accuracy, all while avoiding the computational overhead of traditional data assimilation procedures.

1.1 Research Objectives and Scope

This report documents research carried out during a three-month MSc internship at Deltares, in collaboration with Delft University of Technology. The internship contributed to an ongoing effort to develop fast, reliable tools for storm surge forecasting in the North Sea by combining physics-based models with machine learning surrogates. In particular, the project investigates the feasibility and potential of AI-learned input corrections for systematic biases in coastal surge models. The key research question is

Can a small neural network learn to correct the systematic input biases, specifically wind stress underestimation, by training on observations and predictions from a surrogate model?

To address this question, the study uses two test cases:,

- Validation on a simple test case, the Burgers' equation. First, the correction approach is demonstrated on a well-studied nonlinear PDE with controlled 20% input bias, which allows for isolated testing of the methodology without the additional complexities of shallow-water physics.
- Application to a realistic physical system, the 1D shallow-water with wind forcing. Second, the approach is applied to a simplified but physically accurate model of storm-surge dynamics with a 21% wind stress bias, introducing realistic complications, such as spatially varying bathymetry, quadratic friction, wave propagation, and numerical stability constraints.

The project scope is limited to two one-dimensional synthetic models, to establish methodological proof-of-concept and to identify practical challenges, such as surrogate extrapolation limitations and multi-step error accumulation, that must be addressed before operational deployment. All numerical and machine-learning work is implemented in Julia, using DifferentialEquations.jl for PDE solving and Flux.jl for neural-network training.

The application of AI-based forecasting in coastal flood protection raises important ethical considerations. In particular, forecasting systems have to remain transparent and trustworthy, since their outputs influence important decisions made by public authorities. Additionally, the introduction of machine-learning surrogates introduces potential failure modes distinct from those of traditional solvers, for example, a learned correction might behave unexpectedly under extreme forcing or unusual atmospheric conditions, potentially worsening forecasts at critical moments instead of improving them. In an operational setting, these AI methods would require a human expert's oversight, explicit communication of forecast uncertainty, and conservative fallback

options to more traditional numerical models. These precautions are fundamental for maintaining public confidence and ensuring responsible handling. In this project, these considerations primarily help the motivation: the methods investigated are intended as research contributions towards more efficient and accurate forecasting tools, not as production-ready replacements.

Report Structure

The remainder of this report is structured as follows.

Chapter 2 provides background and methodology. It reviews key ideas about machine-learning surrogates for PDEs, introduces the two-stage correction framework adopted in this work, and outlines the evaluation metrics and implementation context. Chapter 3 details the technical background of the two test models, including the governing equations, the systematic biases introduced, and the main challenges encountered in their numerical simulation.

Chapter 4 describes the experimental design and implementation, covering solver settings, dataset construction, network architectures, and training procedures for both surrogate and correction models. Following this, the chapter presents the results for the two models, with emphasis on the effectiveness of the corrections and their behaviour under multi-step unrolling.

Chapter 5 discusses the findings in relation to the central research question, highlights limitations of the current approach, and suggests directions for future work. The Appendix material provides extended figures, additional diagnostics, and implementation details that support, but are not essential to, the main narrative.

Chapter 2

Methodology Overview

2.1 Machine Learning Surrogates for PDE Models

In this project, neural networks are used as surrogate models, data-driven approximations of numerical solvers for time-dependent partial differential equations. The idea is to replace the expensive time integration of the governing equations by a cheap evaluation of a trained model that maps a given state to its next time step. Furthermore, for the one-dimensional models considered here, the state is represented as a vector of values at discrete spatial locations, and the surrogate network learns a map of the form

$$x_{k+1} \approx f_{\text{surrogate}}(x_k; \theta)$$

where x_k denotes the state at discrete time index k. During the network training, the parameters, θ, are optimised to minimise a loss function

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_k \left\| f_{\text{surrogate}}(x_k; \theta) - x_{k+1} \right\|^2 \tag{2.1}$$

In practice, $f_{\text{surrogate}}$ is implemented as a small stack of standard neural-network layers. The exact architecture is not central here, only that the surrogate is fast to evaluate and differentiable. In addition, the training targets x_{k+1} are obtained from high-resolution simulations of the underlying PDEs using standard ODE solvers.

This perspective fits within the broader landscape of machine-learning approximations for PDEs, including neural operators and physics-informed neural networks. Those approaches typically focus on general function-to-function mappings or incorporate the PDE directly into the loss function. In contrast, the present work uses a deliberately simpler viewpoint: the surrogate is trained purely from input–output data generated by a trusted solver, and its main role is to provide a fast approximation of that solver on a fixed discretisation. Crucially, the surrogate must remain differentiable so that a downstream correction network can be trained through it via backpropagation.

A key difference in the current project in comparison to many surrogate studies is that the input data is intentionally biased. For both test models, the surrogate is trained on simulations that are driven by systematically biased (underestimated) inputs, a scaled velocity in the Burgers' model and scaled wind stress in the Shallow-Water model. This structure is important to the following correction step; the surrogate is not meant to approximate the 'true' physical system directly, but rather the behaviour of an imperfect model exposed to biased forcing. Furthermore, a key requirement for the correction approach to work is that the surrogate must learn physically consistent response relationships: increasing the input amplitude should produce proportionally larger responses in the surrogate, mirroring the numerical solver's behaviour.

2.2 The Two-Stage Correction Framework

The main methodological contribution of this work is a two-stage learning framework with the goal of correcting systematic input biases. The two stages are

- Stage 1. Surrogate training on biased data, where a neural-network surrogate $f_{\text{surrogate}}$ is trained to emulate the biased numerical solver. For each model, a long reference trajectory is generated by integrating the PDE with biased input, and one-step-ahead training samples (x_k, x_{k+1}) are created. The surrogate minimises the mean squared error as described above in Equation 2.1.

- Stage 2. Correction learns using the true trajectories, where a second smaller network $g_{\text{correction}}$ is trained while keeping the previous $f_{\text{surrogate}}$ fixed, meaning its parameters remain frozen during this stage. The role of this NN is to learn a correction that compensates for the systematic bias; for this, new trajectories are generated from the numerical solver with unbiased input. The correction network receives a combination of the surrogate input or the exact variables, from which it either produces

  - an additive correction in the state space, or
  - a correction factor for the surrogate input, e.g. a scale for the wind forcing amplitude.

  The training objective of the network is to minimise the differences between the corrected surrogate output and the true state, as

$$\mathcal{L}_{\text{correction}} = \frac{1}{N} \sum_{k} \left\| \hat{x}_{k+1}^{\text{corrected}} - x_{k+1}^{\text{true}} \right\|^2$$

The main idea is that the complex dynamics remain encoded in the surrogate, while the correction network only has to learn a relatively low-dimensional mapping that encodes the bias. This separation keeps the correction lightweight and interpretable. In future workflow, such a correction could be updated as new observations become available, without retraining the entire surrogate.

Moreover, notice how the alternative of training the correction directly against the numerical solver without an intermediate surrogate is impractical in this setting. The numerical solvers (via ODE integrators in `DifferentialEquations.jl`) are not differentiable with respect to their internal parameters, preventing gradient-based training of a correction applied upstream. The surrogate, by contrast, is fully differentiable, making it the natural intermediate for learning corrections.

2.2.1 Bias-Correction Strategies

Within the two-stage framework, several concrete architectures for $g_{\text{correction}}$ are explored. The choice depends on whether the correction acts in the state space or in the inputs.

Additive state correction

This version acts directly on the surrogate's one-step prediction,

$$x_{k+1}^{\text{final}} = f_{\text{surrogate}}\left(x_k^{\text{biased}}\right) + g_{\text{correction}}\left(x_k^{\text{biased}},\, f_{\text{surrogate}}\left(x_k^{\text{biased}}\right)\right)$$

Here, $g_{\text{correction}}$ is a small neural network that operates on both the current biased state and the surrogate output. The primary advantage of this design is that it directly learns the difference between the biased surrogate output and the actual state. If the surrogate is already reasonably accurate, this residual is small and structured, making it ideal for residual learning.

Input correction (forcing calibration)

In this approach, the correction network acts on the forcing rather than the state, such that

$$\tau_k^{\text{corrected}} = g_{\text{correction}}\left(x_k^{\text{biased}},\, \tau_k^{\text{base}}\right), \qquad x_{k+1}^{\text{final}} = f_{\text{surrogate}}\left(x_k^{\text{biased}},\, \tau_k^{\text{corrected}}\right)$$

where τ^{base} represents the original biased forcing. For the first model, this can be interpreted as correcting a scaled initial condition, while for the second model, it directly represents learning the multiplicative factor on the base wind stress. This architecture is conceptually closer to the physical parameter calibration, where, instead of modifying the state after the prediction, we modify the inputs that drive the dynamics. In practice, the correction network outputs either a single global scale factor α or a location-dependent field α_i that multiplies the base forcing. The surrogate is then run forward using this corrected forcing. Because the correction influences the entire future trajectory, multi-step training becomes particularly important in these experiments.

Combined chain implementation

For implementation, and to explore the viability of the idea, the surrogate and correction are wrapped into a single combined model,

$$\text{CombinedModel}(x) = \begin{cases} x \mapsto f_{\text{surrogate}}(x) + g_{\text{correction}}(\cdot) & \text{additive case} \\ x \mapsto f_{\text{surrogate}}(x;\, g_{\text{correction}}(\cdot)) & \text{input case} \end{cases}$$

During training, gradients are backpropagated through the full combined mapping, but only the parameters of the correction network are updated; the surrogate weights remain frozen. Conceptually, the whole chain behaves as one neural network with a fixed first part (the surrogate) and a trainable second part (the correction). In preliminary tests, this combined implementation produced numerically identical results and very similar training times compared to treating surrogate and correction as separate models, so the final experiments use the simpler independent formulation.

2.2.2 Multi-Step Unrolling and Stability

Training a one-step model on pairs (x_k, x_{k+1}) does not directly guarantee good behaviour when the model is iterated over many time steps. Errors may accumulate and amplify, especially for these nonlinear systems. To address this, the correction model uses multi-step unrolling during the training. Starting from an initial state x_0, the correction and surrogate chain is iterated for K steps to produce $\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_K$. At each step, the model's output becomes the input to the next time step, so small inaccuracies can compound over time. Furthermore, the loss is computed over the steps, such as

$$\mathcal{L}_{\text{correction, unroll}} = \frac{1}{K} \sum_{j=1}^{K} \left\| \hat{x}_j - x_j^{\text{true}} \right\|^2$$

In this setting, gradients are backpropagated through all K steps with the surrogate parameters fixed, rather than only considering a single one-step error.

This ensures that the correction accounts not only for instantaneous mismatches but also for how its decisions affect the future states. In particular, for the second model, where the correction adjusts the wind stress that mainly drives the nonlinear wave system, this is a crucial step. An over-aggressive correction may reduce the short-term error but destabilise the long rollout. Hence, including several steps in the loss helps the network learn conservative corrections that remain stable over operationally relevant horizons. Additionally, sufficiently long unrolling horizons can risk the correction learning to compensate for prediction errors rather than the true systematic bias; thus, the number of steps in the experiments is typically chosen as a practical balance between the two.

Evaluation Strategy and Error Metrics

To quantitatively assess surrogate and correction performance, several complementary metrics are used

- The complete global RMSE

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}(y_i - \hat{y_i})^2}$$

  where y_i and $\hat{y_i}$ denote the true and the predicted values across all spatial points and time steps in the evaluation window.

- Normalised RMSE, computed as NRMSE = RMSE/σ_y, where σ_y is the standard deviation of the observations, enabling scale-independent comparison across models and variables.

- Spatial RMSE, the mean error across all locations at each time step, highlighting where predictions fail.

- Temporal RMSE, the mean error at each location over all time steps, identifying drift.

- The coefficient of determination, R^2, measuring the fraction of variance in y explained by $\hat{y}$,

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

- The pointwise tolerance accuracy, as the percentage of points in space and time, for which the absolute error is below a given tolerance. Useful for assessing how often the corrected surrogate is 'good enough' at all locations, which is relevant in flood-risk contexts where the local errors can be critical

Unless explicitly stated otherwise, all RMSE-based metrics in this report are computed between a model prediction (surrogate or corrected) and the corresponding numerical solver trajectory at the same forcing configuration, evaluated over all space–time points in the window of interest. Likewise, for multi-step roll-outs, these metrics are evaluated over an early-time window to determine the short-term accuracy, and a late-time window to investigate the error growth and stability.

Chapter 3

Technical Background and Mathematical Framework

3.1 Burgers' Equation as Validation Test Case (Model 1)

The first model is based on the Burgers' equation, a standard nonlinear PDE that combines convection and diffusion [1]. While simpler than shallow-water equations, it retains sufficient complexity to test the correction methodology on a well-understood system. The governing equation is

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \nu\frac{\partial^2 u}{\partial x^2}$$

where $u(x,\, t)$ represents the flow velocity field as a function of space x and time t, and where ν is the viscosity coefficient controlling the strength of dissipation. The spatial domain $x \in [0, L]$ is periodic, meaning that the field wraps around at the boundaries, which eliminates boundary conditions complications during this initial validation phase.

The first term of the equation ($\partial_t u$) represents the temporal evolution, the second term ($u\,\partial_x u$) represents nonlinear self-advection, where the flow transports its own velocity field, and the term on the right ($\nu\,\partial_{xx} u$) represents the dissipation through viscous spreading. In particular, this version of the Burgers' equation represents a simplified version of the full Navier-Stokes equations and is well-studied in the literature as a model problem for testing numerical methods and machine learning approaches.

Systematic Bias in the Model Input

The core experiment introduces a controlled systematic error in the input to the reference model, mimicking real-world situations where forcing data contains parameterisation biases. Specifically, all input data are scaled by a constant factor $\beta = 0.8$, so that the model receives only 80% of the true signal. Under this scaling, a surrogate trained on such trajectories learns the dynamics of a weakened system: since the training data are always 20% underestimated, the surrogate cannot distinguish between a genuinely weaker system and the numerical solver's true behaviour. Mathematically,

$$u_{\text{input}}(x,\, t) = 0.8 \cdot u_{\text{true}}(x,\, t)$$

The surrogate model trained on this biased input learns to replicate the dynamics of the physical solver when driven with this weakened signal. The main question then becomes: can a small correction network learn the missing 20% factor and restore accuracy?

This scaling factor is intentionally chosen and kept constant across the domain and time in order to make the problem more easily interpretable. It represents a uniform systematic bias that a properly designed correction model should be able to detect and compensate for. Thus, in the correction, any improvement in

accuracy can be attributed to the learned prediction rather than to accidental cancellation of more complex errors. With this in mind, we can frame the correction task as a two-stage process.

Phase 1. Surrogate Model Training

First, a convolutional neural network surrogate is trained on unrolled trajectories generated by the Burgers numerical solver. The surrogate learns the one-step-ahead prediction map:

$$u_{k+1} = f_{\text{surrogate}}(u_k)$$

where u_k denotes the velocity field at discrete time index k. This surrogate is trained on data generated with the biased (80%) input, so it inherently learns the dynamics of the weakened system.

Phase 2. Correction Model Training

In the second phase, a smaller correction network is trained with the surrogate frozen. The correction model learns to augment or transform the surrogate predictions to match observations of the true (unbiased) system. Three different correction architectures are tested :

1. Additive Correction, where $Y_{\text{final}} = f_{\text{surrogate}}(X_{\text{biased}}) + g_{\text{correction}}(X_{\text{biased}},\, f_{\text{surrogate}}(X_{\text{biased}}))$.
2. Input Correction, where $Y_{\text{final}} = f_{\text{surrogate}}(g_{\text{correction}}(X_{\text{biased}}, f_{\text{surrogate}}(X_{\text{biased}})))$ or $Y_{\text{final}} = f_{\text{surrogate}}(g_{\text{correction}}(X_{\text{biased}}))$.
3. Combined Chain Implementation, where both architectures are implemented as a unified differentiable model while keeping the surrogate parameters frozen.

In this setting, the surrogate represents the imperfect baseline model, and the correction acts as a learnable calibration that adapts this baseline without requiring access to the underlying physical equations. This is a direct specialisation of the general two-stage framework previously introduced.

3.2 1D Shallow-Water Equations with Wind Stress (Model 2)

The second model transitions from the simplified Burgers' equation to a physically grounded representation of coastal storm-surge dynamics [3]. It is based on the one-dimensional shallow-water equations, a standard system for depth-averaged coastal flows. The governing system consists of two coupled partial differential equations which describe the mass conservation, continuity, and momentum balance in a one-dimensional channel with spatially varying bathymetry and time-varying wind forcing.

The continuity equation describes the conservation of water mass, with

$$\frac{\partial h}{\partial t} + \frac{\partial (Hu)}{\partial x} = 0$$

where $h(x,\, t)$ represents the water surface elevation above a reference level (the surge height), $u(x, t)$ the horizontal flow velocity, and $H(x,\, t) = D(x) + h(x,\, t)$ denotes the total water depth. Additionally, $D(x)$ is the bathymetry, depth below the reference level, which varies spatially to represent the transition from shallow coastal waters to deeper offshore regions.

The momentum equation balances pressure gradients, wind forcing, and frictional dissipation, as

$$\frac{\partial u}{\partial t} + g\frac{\partial h}{\partial x} = \frac{\tau(t)}{\rho\, H} - \frac{\rho\, g\, u|u|}{H\, C^2}$$

Similarly, each term has a clear physical interpretation

- $g\, \partial_x h$, represents the pressure gradient force due to water surface slope, the gravity drives flow from high to low elevation.

- $\tau(t)/(\rho H)$, represents the wind stress forcing driving the surge, where $\tau(t)$ is the applied wind stress, and ρ is the water density.

- $\rho g u|u|/(H C^2)$, represents the quadratic bottom friction, where C is the Chezy coefficient that quantifies the roughness.

The friction term has the additional absolute value $|u|$ to ensure that the dissipation opposes the flow direction regardless of the sign. This nonlinearity is critical for the models' accuracy in representing the energy loss in the shallow coastal water.

Boundary Conditions and Domain Configuration

The spatial domain is defined as a closed channel with specified inflows and outflows at the boundaries, $x \in [0,\, L]$, such that,

$$u(x = 0,\, t) = \frac{q_{\text{left}}(t)}{H\,W}, \qquad u(x = L,\, t) = \frac{q_{\text{right}}(t)}{H\,W}$$

where W is the channel width and $q_{\text{left}}(t)$, $q_{\text{right}}(t)$ are the predefined flows. For the experiments presented here, both boundaries are set to zero-flux conditions, $q_{\text{left}}(t) = 0$ and $q_{\text{right}}(t) = 0$. This is chosen to represent a closed domain where dynamics are driven purely by wind forcing and initial perturbations.

The bathymetry profile $D(x)$ is constructed with two smooth hyperbolic tangent transitions to mimic a realistic coastal cross-section, such that

$$D(x) = D_{\min} + \frac{D_{\max} - D_{\min}}{2}\left(1 + \tanh\left(\frac{x - x_1}{w_1}\right)\right) + \frac{D_{\max} - D_{\min}}{2}\left(1 + \tanh\left(\frac{x - x_2}{w_2}\right)\right)$$

where x_1, x_2 are transition locations and w_1, w_2 control the steepness. This creates a shallow region with depth $D_{\min}$ at the boundaries and a smooth transition into deeper water with depth $D_{\max}$ in the interior, see Figure 3.1. This precisely represents a simplified estuary or coastal embayment geometry.

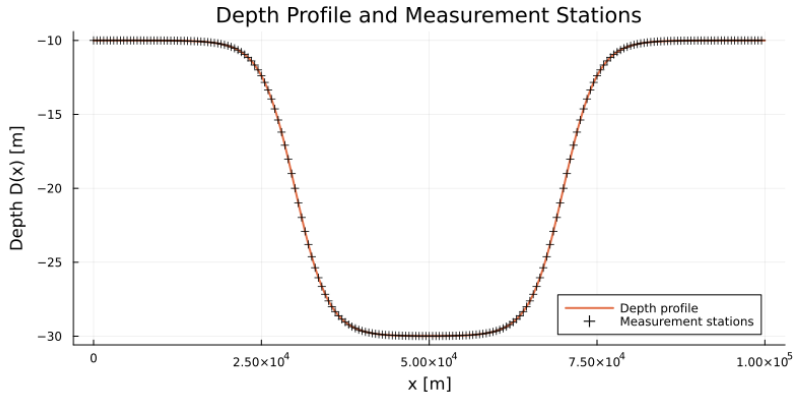


Figure 3.1: Bathymetry profile used in the 1D shallow-water model.

Wind Stress Parameterisation and Systematic Bias

Wind stress $\tau(t)$ drives the storm surge in the simulations. The primary forcing function used is a periodic sinusoidal wind,

$$\tau(t) = \left(A \sin\left(\frac{2\pi t}{T}\right)\right)^2$$

where A represents the amplitude and T is the period, typically 8 hours to represent diurnal or semidiurnal atmospheric patterns. This squared form is commonly used in wind stress parameterisation because wind stress is proportional to the square of wind speed. This formulation ensures non-negativity and naturally

introduces higher harmonics.

The surrogate model of the experiments is trained on data generated with an original wind forcing $\tau_{\text{base}}(t)$, while the 'true' observations are generated using

$$\tau_{\text{true}}(t) = \left(1.1 \cdot A \sin\left(\frac{2\pi t}{T}\right)\right)^2 = 1.21 \cdot \tau_{\text{base}}(t)$$

This choice introduces a controlled systematic bias that mimics a persistent underestimation of wind stress, as often observed in operational forecasts. Because wind stress enters quadratically through the sine-square formula, a 10% underestimation in wind amplitude A translates to a 21% deficit in wind stress, $1.1^2 = 1.21$. The correction model must learn to detect and compensate for this factor of 1.21 using only different combinations of parameters at sparse spatial locations.

Numerical Stability and Multi-Scale Dynamics

Unlike the Burgers equation, the shallow-water system exhibits significant numerical sensitivity. Several factors contribute to instability during the surrogate development.

Initially, an explicit Runge–Kutta solver was considered, but for the final experiments, the KenCarp4 integrator from `DifferentialEquations.jl` is used. This scheme is subject to the Courant–Friedrichs–Lewy (CFL) condition

$$\Delta t \leq CFL \cdot \frac{\Delta x}{\sqrt{g D_{\max}}}$$

where Δx is the spatial grid spacing. In particular, for the parameters used ($\Delta x = 500$m, $D_{\max} \approx 30$m, $g = 10\text{m/s}^2$), the maximum stable time step is approximately 30 seconds. Moreover, the numerical solver automatically maintains stability through adaptive time-stepping, ensuring Δt remains below the CFL limit. Training data are sampled at 60-second intervals, which is sufficiently coarse for stable integration.

Furthermore, the spatially varying depth $D(x)$ introduces diverse wave speeds. The shallow regions support slower, higher-amplitude waves, while deeper regions allow for faster propagation. Meaning that the surrogate must learn this spatial dependence without explicit knowledge of the underlying wave equation structure. Additionally, the quadratic friction term introduces velocity-dependent dissipation that varies by location, since it scales as $u|u|/H^2$, regions with large velocity or shallow depth experience stronger damping. This asymmetry means surge growth (wind-up) is damped differently than surge decay (wind-down), creating nonlinear dynamics that a surrogate must learn.

Chapter 4

Experimental Design and Implementation

4.1 Model 1: Burgers' Equations

The reference numerical solver for Burgers' equation uses finite-difference spatial discretization on a uniform grid with circular (periodic) boundary conditions. This setup provides a controlled environment where the effect of a known 20% input bias can be isolated and systematically corrected. Key parameters are specified in table 4.1.

Parameter	**Value**	**Description**
Spatial points (n_x)	100	Number of grid nodes in x-direction
Domain length (L)	10.0	Length of periodic domain [0, L]
Grid spacing (Δx)	0.1	Uniform spacing, L/n_x
Viscosity (ν)	0.1	Diffusivity coefficient
Initial condition	$u(x,\,0) = 1.0 + 0.5\cos(2\pi x/L)$	Smooth sinusoidal perturbation
Integration time	0 to 100 time units	Long enough to observe dissipation effects
Integration method	Tsit5 (explicit RK scheme)	Adaptive time-stepping ODE solver

Table 4.1: Burgers' Equation model parameters used for surrogate training.

Furthermore, the data were generated in two parts: a clean trajectory by solving the unbiased Burgers equation, and a training dataset by solving the same equation with all velocity states scaled to 80% at each time step, simulating the biased input. Both datasets were saved at a fixed CFL-based interval to ensure consistent temporal sampling. Consequently, the one-step training samples were extracted at identical time spacing from both the biased and unbiased trajectories to avoid temporal mismatches between datasets. The training dataset consists of 600 one-step-ahead samples, split into 500 training and 100 validation prediction pairs. For each sample index k,

$$(X_k, Y_k) = \big(u_{\text{biased}}(x, t_k),\ u_{\text{biased}}(x, t_{k+1})\big),$$

i.e. the input is the biased velocity field at time t_k and the target is the biased field at the next saved time t_{k+1}.

The evaluation uses a fully independent reference trajectory generated with the unbiased solver, unrolled over the full 100 time units. This trajectory is never used during training or validation. For a multi-step assessment, we recursively feed model predictions into the surrogate/corrected chain and compare the

resulting unrolled trajectory against the reference; this procedure measures both short-term accuracy and long-term stability without any overlap with the training data.

Surrogate Network Architecture

The surrogate employs a convolutional residual block specifically designed for spatially structured data, where we input the velocity field and output the predicted velocity at the next time step, with four 1D convolutional layers embedded in a skip connection. Concretely,

- Hidden channels, 8
- Convolutional filter width, 3 (kernel size)
- Circular padding strategy, periodic boundary handling
- Batch size, 32
- Training epochs, 1000
- Activation function, Swish
- Learning rate, 0.001 (Adam optimizer)

Furthermore, the residual update is defined as $u_{\text{out}} = u_{\text{in}} + \text{ConvNet}(u_{\text{in}})$, allowing the network to learn small incremental corrections to the input, facilitating convergence and stability.

Correction Network Architectures

The additive correction network operates in two concatenated channels, the surrogate's prediction and the biased input. It learns to output a spatial field representing the per-location adjustment needed. The configuration of the model is also a convolutional neural network with 8 hidden channels, the swish activation function and circular padding to maintain the periodic structure. Although the corrective chain has a similar structure to the surrogate, reusing the same convolutional building blocks keeps the architecture simple and ensures that the correction can exploit the same spatial patterns as the surrogate. In addition, we use the mean-squared error between the corrected and the true (unbiased) next state as the loss function. The final output is computed element-wise : $Y_{\text{corrected}} = Y_{\text{surrogate}} + \Delta Y_{\text{correction}}$.

In comparison, the input correction modifies the input before passing it to the surrogate, implementing a preprocessing transformation. The model has 2 input channels (the pretrained surrogate output and the biased input features) following the same concatenation, but a much simpler architecture with only 2 dense layers with 16 hidden channels. This lightweight design makes it computationally more efficient, though at the cost of losing some spatial locality that convolutions preserve. The network returns a per-location multiplicative correction factor that rescales the biased input before it is passed through the frozen surrogate.

For the additive correction, we train for 1000 epochs with batch size 40 and learning rate 10^{-3}(Adam), which is sufficient for convergence without overfitting given the modest dataset size. For the input correction, a longer training of 3000 epochs with the same batch size and learning rate is used, reflecting the slightly harder task of learning a stable multiplicative factor.

4.2 Results Model 1: Burgers' Equations

After training, the corrected models achieved significant improvements over the raw surrogate as seen in table 4.2. The additive correction achieved the lowest RMSE (0.0170) and highest R^2 (0.977), reducing error by 50.4% compared to the uncorrected surrogate. The input correction was slightly less effective (RMSE 0.0216, R^2 0.963) but still achieved 37.2% error reduction. Both methods exceeded the 2.5% tolerance threshold for >97% of spatial-temporal points, compared to 84% for the surrogate alone, indicating substantial localised improvements. The superior performance of additive correction suggests that learning explicit residuals in state space is more effective than learning a global input scaling for this simple periodic problem.

Model	Global RMSE	% within 5% tolerance	R^2 Score
Surrogate alone	0.0344	84.3%	0.905
Additive correction	0.0170	99.3%	0.977
Input correction	0.0216	97.1%	0.963

Table 4.2: Performance metrics for Burgers' equation correction models.

Overall, both correction strategies substantially reduce prediction error, with the additive correction performing best. This is expected because additive correction operates directly in state space and can learn spatially varying residual structures. In contrast, input correction is restricted to applying a global transformation to the input. In particular, for Burgers' equation, which is smooth, periodic, and dominated by spatial patterns, the residual formulation naturally aligns with the error's structure. Here, the 'percentage within tolerance' refers to the share of all space–time points where the relative error $|\hat{u}-u|/(|u|+\varepsilon)$ is below 5%. In contrast, the 'accuracy' reported later on is based on an absolute error threshold $|\hat{u}-u| < 0.025$, which emulates a fixed operational tolerance in physical units.

Multi-Step Rollout Stability and Degradation

While single-step metrics are encouraging, the true test is unrolling the models over many time steps. Errors compound because each prediction serves as input to the next step. To assess this, unrolled trajectories were compared over two time windows, see Tables 4.3 and 4.4. Here, accuracy refers to the percentage of spatial–temporal points for which the absolute error $|\hat{u}-u|$ is below a fixed tolerance of 0.025. Improvement is defined as the difference between this accuracy and the accuracy of the uncorrected surrogate over the same time window.

Model	Accuracy	Improvement
Surrogate	91.7%	-
Additive correction	93.5%	+1.85%
Input correction	73.52%	-18.18%

Table 4.3: Early-time accuracy and improvement for Burgers' equation correction models, measured from time steps 50–150 ($\sim$ 1 – 2.5 hours).

Model	Accuracy	Improvement
Surrogate	63.3%	-
Additive correction	89.4%	+26.11%
Input correction	86.31%	+22.99%

Table 4.4: Late-time accuracy and improvement from Burgers' equation correction models over time steps 250 to simulation end ($\sim$ 4 – 6 hours).

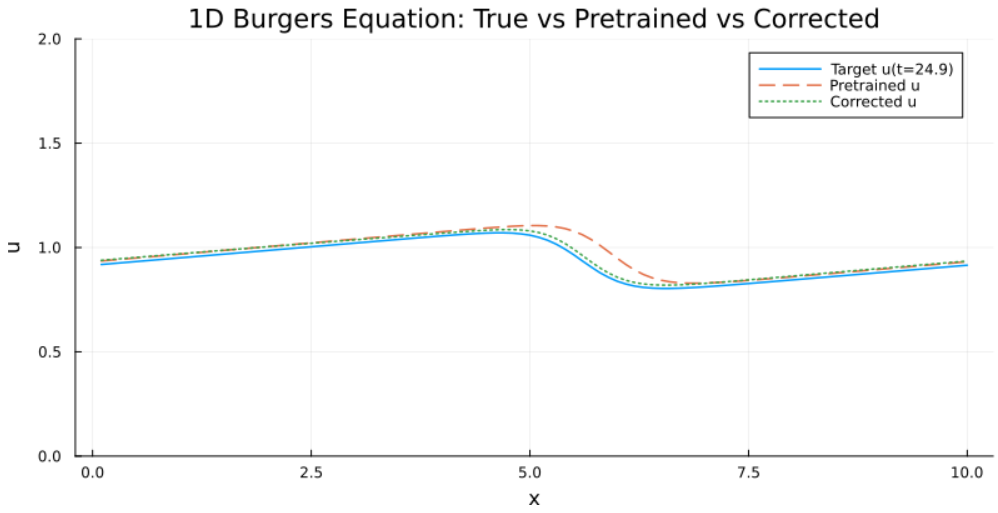
In the early-time window, the surrogate already achieves about 92% accuracy. The additive correction gives only a modest gain (to about 94%), while the input-correction model actually performs worse at this stage, remaining below 75%. In contrast, in the late-time window, where errors have had time to accumulate, both correction models provide large improvements: the surrogate drops to about 63% accuracy, whereas additive and input correction recover it to roughly 89% and 86%, respectively.

The difference in accuracy during both of the time windows is expected, all simulations start from nearly identical initial conditions, so the biased and unbiased dynamics remain close during the first hour. Only after sufficient time does the compounding effect of the 20% input bias become visible, leading to large deviations in the uncorrected surrogate. Overall, we can observe how the main benefit of correction is not a dramatic boost in short-term accuracy, but a substantial reduction in long-term drift, with the input correction showing the slowest degradation over very long horizons.

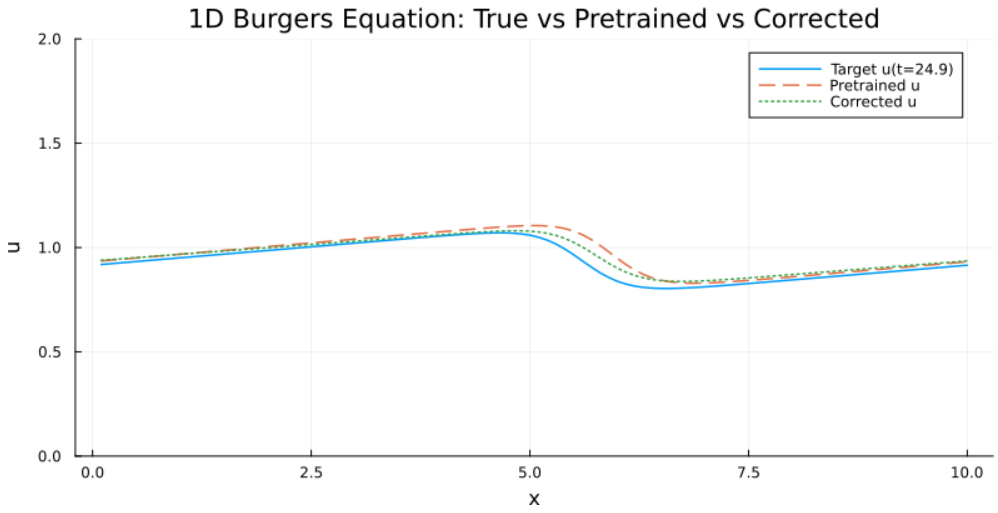
Spatial Snapshots and Field Evolution

Figure 4.1 shows spatial snapshots (u over the full domain) at selected time steps. The blue line represents the unbiased reference solution, while the orange line represents the biased surrogate prediction, and the green

the corrected model prediction. Similarly, Figure 4.2 shows the corresponding time series at a representative spatial location, illustrating how errors accumulate over time and how the correction mitigates this behaviour.



(a) Additive Correction

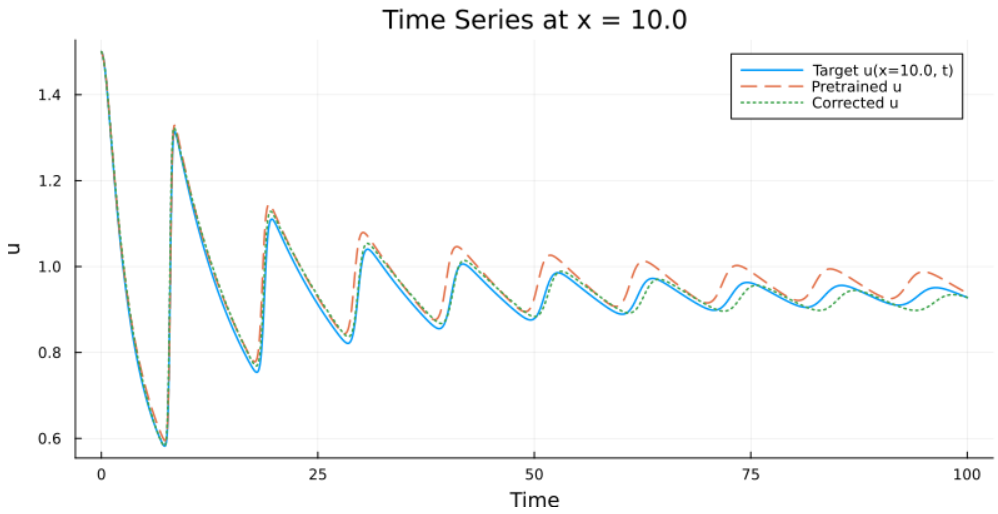


(b) Input Correction

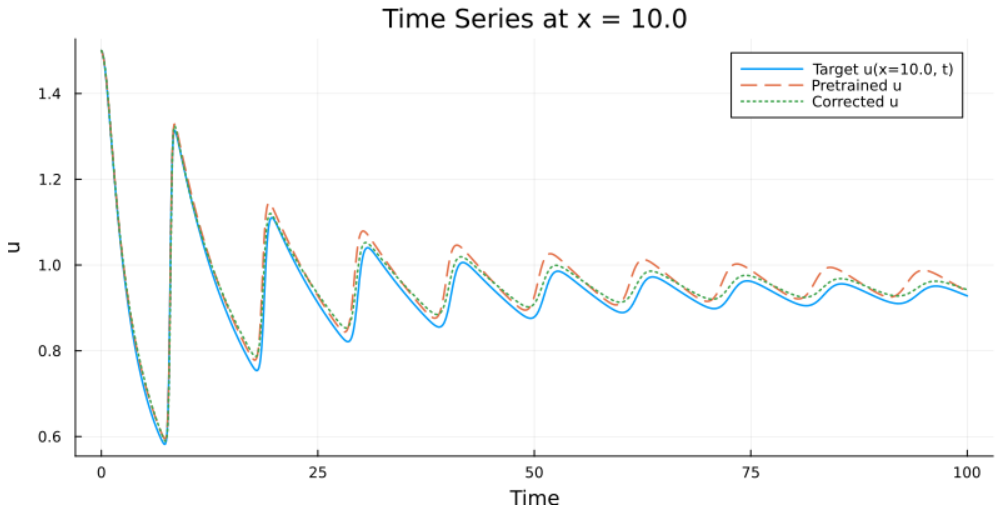
Figure 4.1: Model 1 spatial snapshots at selected time steps.

From the spatial snapshots, we observe that the additive correction tracks the reference very closely during the first part of the simulation, almost eliminating the small phase and amplitude errors of the surrogate. The input correction is slightly less accurate at these early times, but its errors remain more uniform and do not grow as significantly. By the later snapshots, both corrections are clearly closer to the reference than the surrogate, with the input correction showing the smoothest long-term behaviour.

The time-series plots at individual locations confirm this pattern: additive correction minimises the early-time error but exhibits a somewhat faster growth over long rollouts, with a visible phase shift emerging in the later hours. The input correction sacrifices a small amount of short-term accuracy but maintains a more consistent trajectory and smaller long-term drift. If we extend the late-time window further (e.g. beyond step 350), the input correction ultimately yields the lowest sustained error, indicating that it is preferable when very long prediction horizons are of interest.



(a) Additive Correction



(b) Input Correction

Figure 4.2: Time series at a representative spatial location for Model 1.

4.3 Model 2: Shallow Water Equations with Wind Forcing

The reference numerical solver for the shallow-water system uses finite-difference spatial discretisation on a staggered grid to maintain numerical stability and conserve the physical properties. The grid configuration places water heights h at cell centres while velocities u are at the cell edges, such that

- Height grid, $x_h = [\Delta x/2,\, 3\,\Delta x/2,\, 5\,\Delta x/2, \ldots, L - \Delta x/2]$.
- Velocity grid, $x_u = [0,\, \Delta x,\, 2\Delta x,\, \ldots,\, L]$.

This staggered structure ensures that fluxes Hu can be computed at the cell interfaces using the locally-averaged depths, preserving mass conservation to machine precision.

The spatial derivatives are approximated by finite differences, as

$$\left.\frac{\partial h}{\partial x}\right|_i \approx \frac{h_i - h_{i-1}}{\Delta x}, \qquad \left.\frac{\partial (Hu)}{\partial x}\right|_i \approx \frac{(Hu)_{i+1} - (Hu)_i}{\Delta x}$$

Moreover, the time integration uses the KenCarp4 explicit Runge-Kutta solver (4th-order with adaptive stepping), with error tolerances reltol = 10^{-8} and abstol = 10^{-10}. This solver automatically adjusts Δt to satisfy the stability constraints while maintaining the predefined error tolerances.

The domain represents a 100km coastal channel with parameters chosen to reflect typical North Sea conditions, usually scaled due to computational efficiency, see Table 4.5. The bathymetry profile creates a realistic depth-varying wave speed, where shallow regions, $\sqrt{gH} \approx 10$ m/s, versus deep regions, $\sqrt{gH} \approx 17$ m/s, introducing the spatial diversity that the surrogate for the numerical solver must simulate.

Parameter	**Value**	**Description**
Domain length (L)	100 km	Coastal bay scale
Channel width (W)	100 m	Idealized cross-shore width
Spatial resolution (Δx)	500 m	200 grid points
Min depth ($D_{\min}$)	10 m	Shallow nearshore
Max depth ($D_{\max}$)	30 m	Deeper offshore
Depth transitions (x_1, x_2)	30 km, 70 km	Coastal slope locations
Transition widths (w_1, w_2)	5 km each	Smoothness of bathymetry
Water density (ρ)	1000 kg/m^3	Standard seawater
Gravity (g)	10 m/s^2	Approximated for simplicity
Chezy friction (C)	60 m$^{1/2}$/s	Moderate bottom roughness
Wind period (T)	8 hours	Semi-diurnal tidal/atmospheric
Simulation duration	$6 - 12$ hours	Short-term forecast horizon

Table 4.5: Shallow-Water Equations model parameters.

Initial Conditions: Three Test Scenarios

To develop and carefully test the surrogate robustness across different dynamical regimes, three distinct initial conditions were designed.

- Flat Initial Condition ($h_0 = 0, u_0 = 0$), to simulate a system that starts idle. Thus, all dynamics are purely driven by wind forcing, with no initial perturbations. This tests the surrogate's ability to capture forced response from the equilibrium.

- Small Bump Initial Condition ($h_0 = 0.25, u_0 = 0$, width $w = 0.05L$, centre $c = 0.3L$), to represent a localized Gaussian perturbation in the water level with a 25cm amplitude, 5km width and located 30km from the left boundary. This produces a free wave propagation superimposed on wind-driven dynamics, useful for testing the model's ability to handle a combination of wind forcing and free oscillations.

- Larger Bump Initial Condition ($h_0 = 1.0, u_0 = 0$ same width and centre), similarly, represents a more extreme initial surge scenario. This high-amplitude case pushes the nonlinear terms into regimes where small errors can compound rapidly.

Each initial condition creates different balances between wave propagations, wind forcing, and frictional dissipation, see Figure 4.3. These variations allow the surrogate and correction models to be tested under regimes dominated by wind forcing (flat), mixed free-wave and forced response (small bump), and stronger nonlinear interactions (large bump).

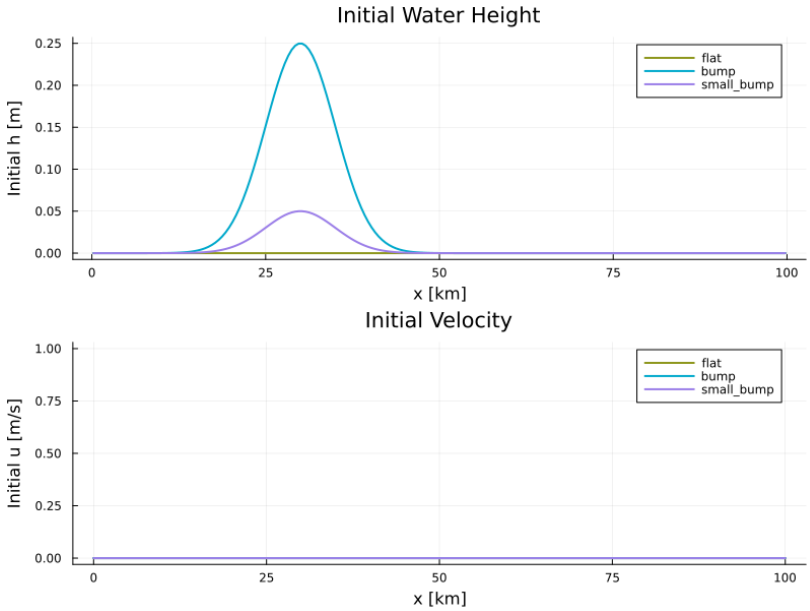


Figure 4.3: Initial conditions in water level and velocity used in the experiments.

Wind Forcing Functions

Although several alternative forcing functions were implemented, the results in this chapter use only the periodic sinusoidal forcing. The additional wind types were explored during development but not used in the final training or evaluation due to stability limitations. These alternative forcings produce more irregular and less predictable wind time series, which made the surrogate training problem substantially harder and often led to unstable long unrollings; in this work they are therefore treated as exploratory, and the main quantitative results focus on the periodic case. In particular, the additional functions are

- Multi-Frequency Forcing, as a superposition of multiple sinusoids,

$$\tau(t) = \sum_{i=1}^{N} w_i \left(A \sin\left(\frac{2\pi t}{T_i} + \phi_i \right) \right)^2$$

- Piecewise Constant Forcing, as step changes with random durations, where the wind stress remains constant for random intervals (averaging 1 hour), then jumps to a new value drawn from $N(\mu, \sigma^2)$.

- Autoregressive Process (AR model), as

$$\tau_t = \sum_{i=1}^{p} \phi_i \tau_{t-i} + \epsilon_t, \qquad \epsilon_t \sim N(0, \sigma^2)$$

Figure 4.4 illustrates the original and biased periodic wind stress time series used for the main experiments. The base curve corresponds to $\tau_{\text{base}}(t)$; while the 'biased' curve represents $\tau_{\text{true}}(t) = 1.21 \cdot \tau_{\text{base}}(t)$. These are exactly the forcings used when generating the surrogate training data and the target trajectories for correction.

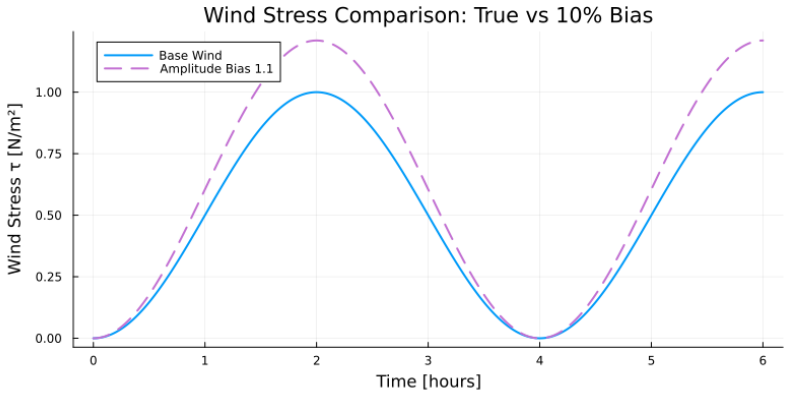


Figure 4.4: Wind forcing time series used for training surrogate and correction networks.

Training Data Generation and Normalisation

The training data is designed as one-step-ahead prediction pairs sampled from long numerical solver trajectories. For each simulation,

1. Use the numerical solver to solve the shallow-water equations from $t = 0$ to $t = 6$ or 12 hours
2. Save the snapshots at predefined intervals, $\Delta t_{out} = 60$s, giving ~ 360 time steps.
3. Form input-output pairs, as $(x_k,\, x_{k+1})$ where for each station $i = 1, 2, \ldots, 100$ at time k we have

$$[h_i,\, u_i,\, D_i,\, \text{mask}_i,\, \tau_k,\, \text{scale}_k]$$

   Although the wind stress τ_k and the amplitude scale 'scale$_k$' are uniform across the domain, they are repeated across all stations in the input. This repetition allows the network to 'see' the local state and the global forcing together at each station, reinforcing the importance of these scalar parameters and matching the structure expected by the correction model, which must later infer the same scaling factor from similar inputs. The physical forcing remains spatially uniform; only its representation in the feature vector is repeated. The station mask $m_i \in \{0, 1\}$ indicates to the model whether the location i is a boundary station.

For simplicity, stations are uniformly spaced every 10^{th} grid points, giving 100 observation locations. Meaning that the full input vector layout is,

$$\begin{aligned}
&\text{Station 1}: \quad [h_1,\, u_1,\, D_1,\, \text{mask}_1,\, \tau,\, \text{scale}] \\
&\text{Station 2}: \quad [h_2,\, u_2,\, D_2,\, \text{mask}_2,\, \tau,\, \text{scale}] \\
&\quad \ldots \\
&\text{Station 100}: \quad [h_{100},\, u_{100},\, D_{100},\, \text{mask}_{100},\, \tau,\, \text{scale}]
\end{aligned}$$

With a total size of $6 \times 100 = 600$ for each trajectory. In mathematical form,

$$x_k = [h_1, \ldots, h_2, \ldots, h_{100},\, u_{100},\, D_{100},\, \text{mask}_{100},\, \tau_k,\, \text{scale}_k]^T \in \mathbb{R}^{600}$$

Furthermore, neural networks train most effectively when the input features have a zero mean to help gradients flow symmetrically, unit variance to avoid large weights destabilising training, and comparable scales so that no feature dominates the updates. Consequently, we implement a unique normalisation scheme.

Initially, we identify the indices of the parameters we should normalise, avoiding the mask, to keep the binary structure, and the amplitude scale, and compute the statistics on the training data. With this, we check

whether we should normalise the wind forcing to prevent division by zero errors and apply a column-wise normalisation of the specific part of the input and keep the rest unchanged, such that

$$\tilde{x}_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j} \qquad \text{or} \qquad \tilde{x}_{ij} = x_{ij}$$

This transformation is also applied to the validation data. Subsequently, after the network makes predictions in the normalised space, we reverse the transformation,

$$\hat{y}_{ij} = \tilde{y}_{ij} \cdot \sigma_j + \mu_j$$

Converting the model predictions back into the physical units.

The development of the surrogate model involved iterative refinement over multiple weeks to achieve a stable, physically accurate surrogate. The first surrogate models used simple dense networks with inputs $[h, u, D, \text{mask}, \tau]$, leading to exploding errors after ~ 3 hours of simulation, insensitivity to wind amplitude changes or overfitting to the specific training scales and failing on test with different wind amplitudes that the model had not been trained on.

The most notable improvement came from explicitly providing the amplitude scale as an additional input feature. Instead of implicitly learning the scale dependence from the wind stress magnitude alone, the surrogate directly received it in the input. This scale-conditioning allowed the network to learn how the wind forcing pattern changes with its overall intensity, notably improving both accuracy and robustness.

Surrogate Network Architecture

The surrogate network is designed with a residual learning strategy, such that it predicts the state increments rather than absolute values, which directly benefits the stability of the training. In particular,

$$\Delta h, \Delta u = f_{\text{surrogate}}(h_k, u_k, D, \tau_k, \text{scale}_k)$$
$$h_{k+1} = h_k + \Delta h, \quad u_{k+1} = u_k + \Delta u,$$

This implementation exploits the fact that changes between consecutive time steps are small, since $\Delta t = 60$ s is much shorter than the dominant system time scales. Predicting these increments is easier and more stable than predicting full states directly, and if the network outputs values close to zero, errors do not accumulate as easily. To further improve robustness and reduce overfitting, small Gaussian noise (with a standard deviation of 10% of the typical increment size) is added to the state variables during training, encouraging the network to learn smooth, noise-tolerant mappings rather than overfitting individual trajectories.

The base parameters of the final version of the surrogate network are,

- 128 hidden channels
- 3 fully-connected layers
- 10 % state noise
- Batch size, 32
- Swish activation function
- 10^{-4} learning rate (Adam optimizer)

The large hidden dimension is necessary to capture spatial gradients and nonlinear interactions across all the stations. add

Multi-Scale and Multi-IC Training Strategy

To address these limitations and improve generalisation across wind amplitudes not seen during training, the surrogate is trained on data spanning multiple scaling factors, for example, on a narrow range of three scales $[0.9, 1.0, 1.2]$ or on a broader range $[0.6, \ldots, 1.4]$. Each scale s modifies the wind amplitude as $\tau(t, s) = (s \cdot A)^2 \sin^2(2\pi t/T)$. Therefore, the training data consists of the independent trajectories for each scale, stacked together to form a diverse dataset. This scale is provided in the amplitude scale feature in

the input, allowing the network to learn it and condition its predictions on the forcing intensity with more physical accuracy.

This multi-scale training strategy substantially enhanced both stability and physical reproducibility. Surrogates trained on a narrower range [0.9, 1.0, 1.2] produced consistently stable unrolled trajectories, even when evaluated at unseen scales. By contrast, surrogates trained on a broader range $[0.6 - 1.4]$ maintained stability across more extreme scales, but their accuracy at the target scales was sometimes reduced, reflecting the same robustness-accuracy trade-off seen in the exploratory surrogate variants described above.

Finally, when multiple initial conditions are included, the resulting surrogates can achieve very low errors for some random initialisations but show large variability between training runs.

Correction Model

Once a stable surrogate is trained on the base wind data (τ_{biased}), the correction model learns to adjust the wind input to match the observations that had been generated with the 'true' wind $\tau_{\text{true}} = 1.21 \cdot \tau_{\text{biased}}$. Thus, the correction takes the form of a learned scaling factor α,

$$\tau_{\text{corrected}} = \alpha \cdot \tau_{\text{biased}}$$

The correction model outputs this scale based on the current state observations and wind information. In addition, two variants were tested,

1. Global scale prediction, since α was applied uniformly to all stations.
2. Per-station scale prediction, where α_i is different for each measurement stations i, allowing for spatially-varying corrections.

The correction model receives one of two input configurations: either the full state vector, identical to the surrogate input, or a reduced feature set containing only the most predictive variables. Normalisation follows the same protocol as for the surrogate.

Crucially, the correction is trained while the surrogate parameters remain frozen, and gradients are backpropagated through the full surrogate–correction chain. As a result, the quality of the correction is fundamentally limited by the surrogate: if the surrogate is unstable or systematically biased in a way not representable by a simple scale factor, the correction model cannot fully recover the true dynamics and may even amplify surrogate errors during long rollouts.

4.4 Results Model 2: Shallow Water Equations with Wind Forcing

4.4.1 Surrogate Model Performance and Stability

To evaluate the different correction methods, we first need to assess the stability and accuracy of the underlying surrogate on the biased wind system. This is essential since the correction effectiveness depends entirely on the quality of the baseline surrogate. As previously described, developing a robust surrogate required iterative refinement of the training strategy. Early surrogates trained on a single scale exhibited significant error blow-ups for various wind bias inputs during unrolling, or were insensitive to wind amplitude variations. This overfitting on the training scale rendered them unreliable for the correction task, which required the surrogate to accurately reproduce the true relationship between the final result and the different ranges of forcing conditions.

Five surrogate variants were evaluated across different seeds and training strategies, which are defined in Table 4.6. Overall, the training strategy revealed a clear trade-off between scale coverage, accuracy, and robustness. Surrogates trained on only three scales (e.g. 2.1, 3.1) learned the biased scale scale = 1.1 more easily, but tended to degrade more rapidly when evaluated far outside their training range. In contrast,

the broad-scale Surrogate 1 was more reliable across a wider range of wind amplitudes, yet showed slightly higher global RMSE at the target scales due to the increased diversity of training conditions.

Surrogate Variant	Trajectory	Training scales	Training IC
Surrogate 1	Short duration	$[0.6, \ldots, 1.4]$	Flat
Surrogate 2.1	Short duration	$[0.9, 1.0, 1.2]$	
Surrogate 2.2	Long duration	$[0.9, 1.0, 1.2]$	
Surrogate 3.1	Short duration	$[0.9, 1.0, 1.2]$	Multiple
Surrogate 3.2	Long duration	$[0.9, 1.0, 1.2]$	

Table 4.6: Summary of surrogate model variants tested for the 1D shallow-water equations.

The multiple initial condition variants (Surrogate 3.x) were the most accurate in the best cases, but also displayed strong sensitivity to random initialisation; most seeds performed worse than the single-IC surrogates, while a single favourable seed yielded the best results. This variability made them less attractive as a stable baseline for the correction experiments, despite their potential peak performance.

Preliminary Results: Surrogate Sensitivity and Generalisation

Before exploring the correction models, it is essential to establish that the underlying surrogate has correctly learned the physical response to wind forcing. A surrogate that is insensitive to changes in wind stress would render any downstream correction trivial, as the correction model relies on the surrogate's ability to translate a modified input scale into a modified hydrodynamic state. To verify this, we tested the surrogates on unseen wind amplitudes (e.g., scale = 1.1) to ensure they could extrapolate beyond their training data (e.g., scale = 1.0).

Figure 4.5 illustrates the performance of Surrogate 2.1 under the two conditions: the standard training scale and the 'true' target scale. The results confirm that the surrogate successfully distinguishes between forcing intensities. In the early stages of the rollout ($t < 3$ hours), the surrogate tracks the numerical solution with high precision for both scales. As the simulation progresses, a typical accumulation of error is observed. Despite this expected drift, the surrogate clearly maintains the correct physical offset between the two scenarios, validating its suitability as a differentiable base for the correction network.

Furthermore, table 4.7 shows that all multiple-scale surrogates achieve low global RMSE. In both columns, the RMSE is computed between each surrogate and the corresponding numerical solution at the same forcing scale (e.g. surrogate at 1.0 vs truth at 1.0, and surrogate at 1.1 vs truth at 1.1), not across different scales. In particular, Surrogate 3.1 performs best, followed by Surrogate 2.1, and finally the broad-scale Surrogate 1. Moreover, the increase in the error when moving from the training scale of 1.0 to the unseen scale of 1.1 is not problematic for any variant. This indicates that the networks do not catastrophically fail under the new forcing and maintain reasonable extrapolation ability.

Model Variant	Base Scale (1.0)	Unseen Scale (1.1)
Surrogate 1	0.0126	0.0124
Surrogate 2.1	0.0067	0.0100
Surrogate 3.1	0.0048	0.0071

Table 4.7: Global RMSE comparison of surrogate variants' predictions against corresponding true solutions.

Additionally, Table 4.8 further presents the error of Surrogate 2.1 across space and scales. We observe how the errors are clearly location-dependent: the boundary stations (1 and 100) exhibit RMSE values one order of magnitude larger than the interior (Station 50), reflecting the stronger dynamical activity and wave

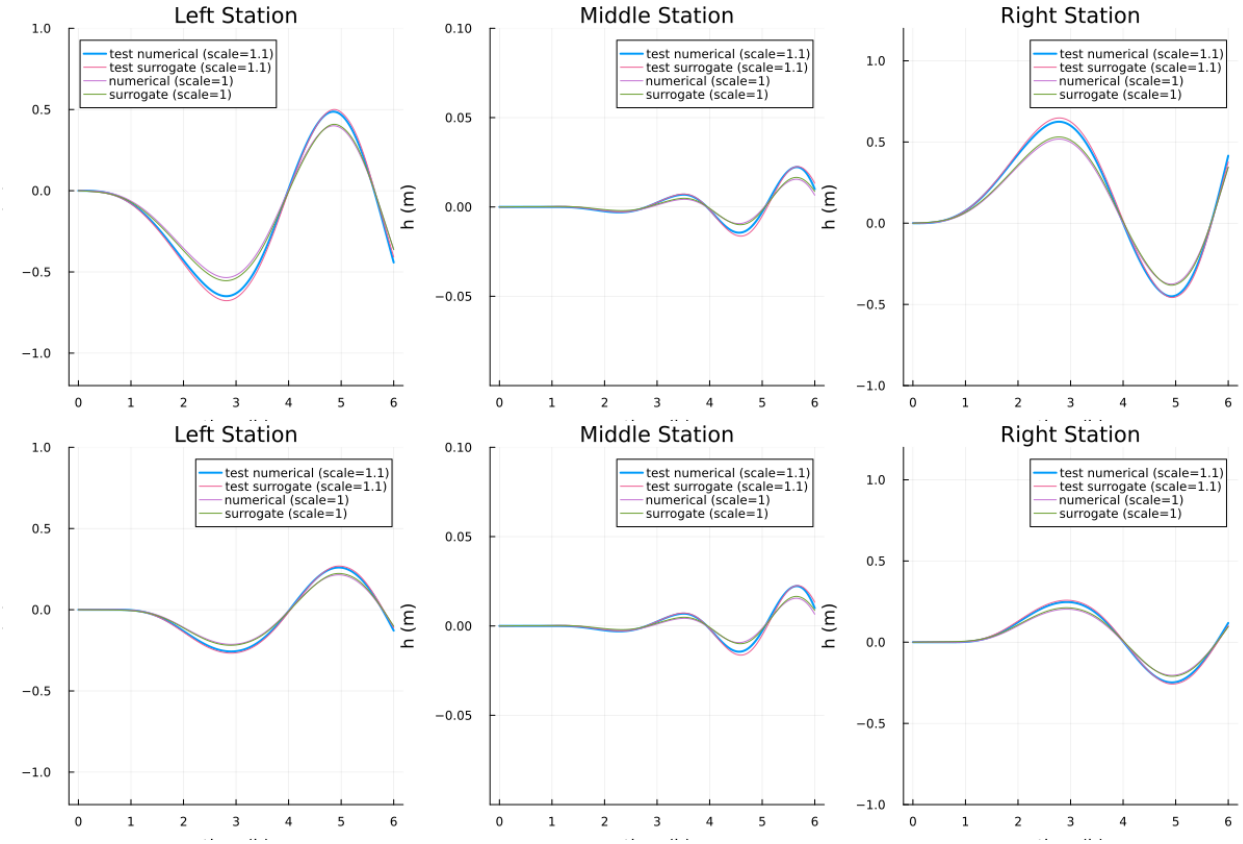


Figure 4.5: Unrolled Surrogate 2.1 output at training and unseen forcing scale

reflections near the boundaries, whereas the mid-domain remains very smooth and thus is predicted with almost negligible absolute error.

Measurement Station	**Base Scale (1.0)**	**Unseen Scale (1.1)**
Station 1 (Left Boundary)	0.0126	0.0174
Station 25	0.0051	0.0074
Station 50	0.0006	0.0009
Station 75	0.0058	0.0081
Station 100 (Right Boundary)	0.0093	0.0151

Table 4.8: RMSE statistics for Surrogate 2.1.

4.4.2 Correction Network Configurations and Architecture Search

Several distinct correction architectures were systematically compared on the different surrogates, holding their parameters fixed and varying the correction design. This configuration search explores how the input features, output parameterisation, and scale prediction strategy affect the learning and generalisation of the network. In particular, the tested configurations are,

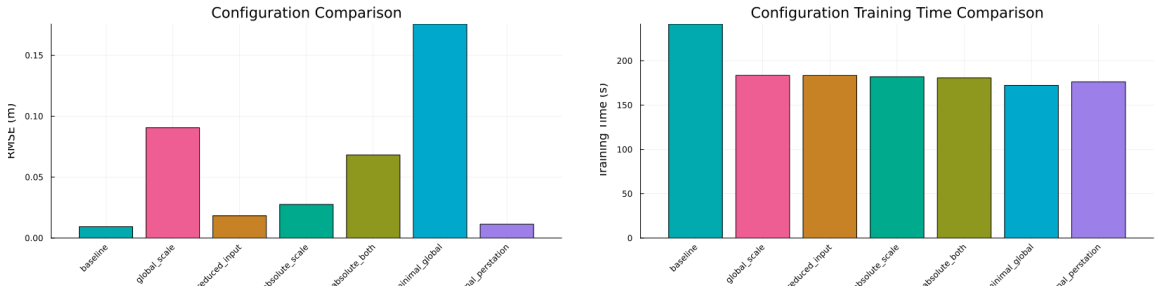
- Baseline: Full surrogate input parameters (h, u, D, mask, τ, scale), a per-station scale correction, and a scale increment prediction.

- Global Scale: Same input, but single global scale α applied to all stations, and also a scale increment prediction.

- Reduced Input: Subset of features (h, u, τ, scale), a per-station scale correction, and a scale increment prediction.
- Absolute Scale: Full input, per-station correction, but predict absolute scale (not increment).
- Global Absolute Both: Full input, but single global scale and predict an absolute value.
- Minimal Global: Reduced input, global scale, absolute scale prediction.
- Minimal Per-Station: Reduced input, per-station scale, absolute scale.

Although Surrogate 2.1 was used to demonstrate surrogate sensitivity to wind amplitude, the main correction experiments shown here were carried out with Surrogate 1. In practice, Surrogate 1 provided a more balanced trade-off between stability and responsiveness: its slightly higher base error is compensated by smoother long-horizon rollouts, and the correction networks trained on it achieved the largest and most consistent error reductions. By contrast, corrections on Surrogate 2.1 improved RMSE only for a subset of configurations and occasionally destabilised the rollout, making it less suitable as a primary demonstrator. The surrogate 1 sensitivity results are included in the Appendix for reference.

Furthermore, the results are found in Figure 4.6 and the Table 4.9, for 5000 epochs of training with a learning rate of 1^{-5}. In these results, we define improvement as the relative change in RMSE compared to the uncorrected surrogate, so positive values indicate error reduction and negative values indicate degradation. For reference, the uncorrected Surrogate 1 has an error of approximately 0.042462 m in comparison to the true solution at scale 1.1, meaning that the 'Improvement' values in Table 4.9 reflect the changes relative to this baseline error.



(a) RMSE performance

(b) Training Time

Figure 4.6: Metrics comparing different correction network configurations applied to Surrogate 1.

Configuration	Corrected RMSE (m)	Improvement	Average α	Training Time
Baseline	0.009293	+78.11%	1.0950	317.27 s
Global Scale	0.090560	-113.27%	0.7114	290.21 s
Reduced Input	0.018382	+56.71%	0.9880	796.83 s
Absolute Scale	0.027574	+35.06%	1.0691	172.06 s
Global Absolute Both	0.068226	-60.68%	0.7114	172.63 s
Minimal Global	0.175651	-313.67%	0.4000	166.03 s
Minimal Per-Station	0.011333	+73.31%	1.1140	178.22 s

Table 4.9: Metrics comparing different correction network configurations applied to Surrogate 1.

From this, we observe that all per-station correction methods (Baseline, Minimal Per-Station, Reduced Input, and Absolute Scale) achieve substantial error reductions, ranging from roughly 35% to over 78%. In contrast, the global correction approaches either fail to improve upon the surrogate or actively degrade performance, with RMSE increasing by more than 100%. This indicates that the impact of wrongly estimating

the wind stress is not uniform across the domain. The local dynamics, such as depth variations and boundary interactions, require spatially varying corrections that a single global scale cannot capture.

In the following results, we will focus on the baseline configuration since it achieves the lowest error and demonstrates the highest stability. Additionally, the average of the final learned α_i values of 1.095 is very close to the true wind scaling bias of 1.1, differing by only 0.5%. We also observe that the Minimal Per-Station approach performs almost as well, achieving a nearly identical error rate and α learning, with the added benefit of slightly faster training. This suggests that most of the information needed to infer the wind scale is contained in a small subset of features and that the per-station structure of the correction is more important than using the full feature set.

4.4.3 Correction Learning Dynamics and Alpha Evolution

After training the correction network with multi-step rollouts (K = 2-4 steps), we observed a stable and consistent decline in the Baseline loss, with no evidence of instability or overfitting. The per-station scale factors, $\alpha_i(t)$, gradually shifted from their initial values and converged toward an average of approximately 1.1 across all stations. This convergence reveals that the network effectively captures the systematic wind underestimation. In other words, the correction network learns to infer the wind bias in a robust and interpretable manner once the optimal configuration is identified.

The per-station scale factors converge to an average that closely matches the true bias. Yet the magnitude of correction is greatly dependent on the spatial position of each point, with slightly stronger adjustments appearing in the final time steps, reflecting the network's effort to compensate for accumulated unrolling error. This non-uniform behaviour, seen in Figure 4.7, highlights that the correction network captures physically meaningful structure. Rather than imposing a uniform fix, it learns that the impact of wind underestimation varies across stations according to local dynamics.

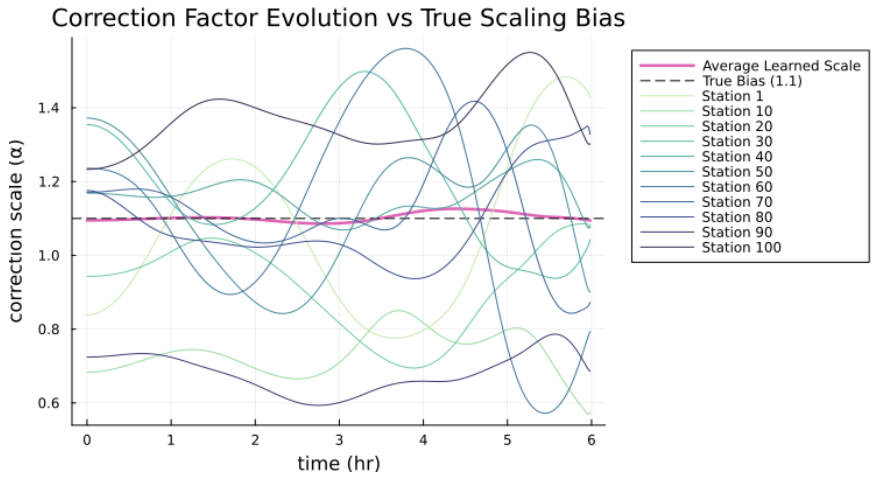


Figure 4.7: Per-station learned scale factors (α_i) for Surrogate 1 baseline correction varying over time.

The learned per-station scales α_i exhibit a weakly oscillatory spatial pattern, with larger corrections near the boundaries, shallower regions, and smaller corrections in the deeper interior. Observe how the most distinct corrections from the true bias are for the stations on the right of the domain, stations at index 90 and 100 (darker shade), while stations in the middle of the domain surround the true bias more closely, see Figure 4.7. This matches the physical expectation that wind stress has a stronger impact where the water column is shallow and where waves reflect from the boundaries. The α_i field also shows a smooth, wave-like structure along the domain; this pattern reflects how the periodic wind forcing projects onto the domain's natural modes: stations that experience larger local surge responses receive slightly higher effective α_i, while quieter regions remain closer to the global mean of about 1.1. Rather than being pure noise, the correction

field therefore encodes how a uniform wind bias translates into non-uniform water-level errors.

Over time, the range of the learned scales also widens slightly: early in the rollout, most α_i values lie roughly between 0.5 and 1.4, whereas at later times they can reach between about 0.6 and 1.6. This gradual broadening reflects the correction network's attempt to compensate not only for the true uniform wind-stress bias but also for accumulated surrogate unrolling error. In other words, part of the non-uniform correction field is actively counteracting surrogate drift, especially near dynamically active regions, rather than representing a change in the underlying physical bias.

It is important to note that, in the numerical experiments, the true wind bias was applied uniformly in space. The fact that the trained network converges to a non-uniform correction field indicates a combination of two effects: the spatially varying sensitivity of shallow-water dynamics to wind forcing, and the residual surrogate modelling errors that are not perfectly homogeneous across the domain. In this sense, a purely uniform correction would be suboptimal, as we can see from the global scale predicting networks' results. The learned spatial structure can be interpreted as a physically informed refinement of the applied bias rather than a contradiction of it.

4.4.4 Multi-Step Unrolling Performance

The major test of correction effectiveness is the multi-step time evolution, where bias and nonlinear error accumulation are most damaging for naive surrogates. Multi-step rollouts allow us to distinguish between two failure modes: either the surrogate itself is too inexpressive to represent the true dynamics, or the correction model fails to identify the correct bias even though the surrogate would, in principle, allow it.

A direct comparison of the surrogate-only and corrected RMSE (Figure 4.9 and 4.8) confirms that the correction model not only lowers the domain-averaged error but also improves the spatial and temporal distribution of errors, particularly in regions most affected by the biased wind forcing. Across all 100 stations, the corrected spatial RMSE stays below about 1.5 cm at most locations, with somewhat larger residuals near the shallower boundary regions where the dynamics are strongest. The temporal RMSE curves show that the corrected error grows slowly over time and remains well below it even in the later hours of the simulation. This indicates that the correction is effective both for the domain mean and at individual locations, and that it primarily acts by damping the late-time drift that would otherwise arise from the systematic wind underestimation.

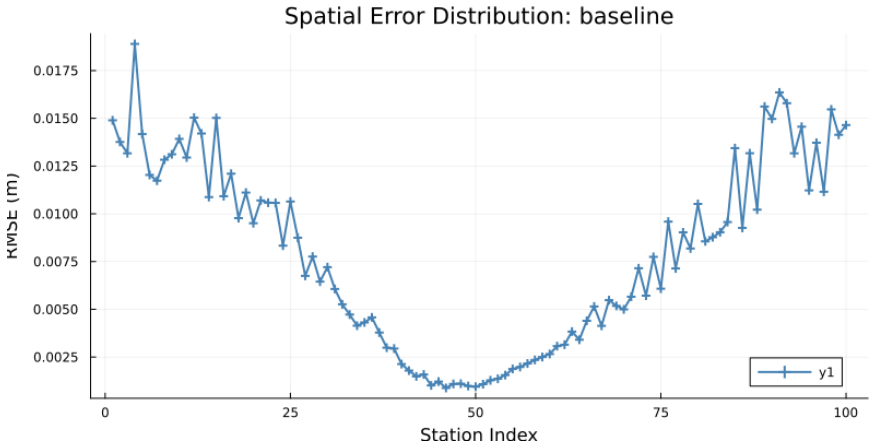


Figure 4.8: Spatial RMSE for Surrogate 1 baseline correction.

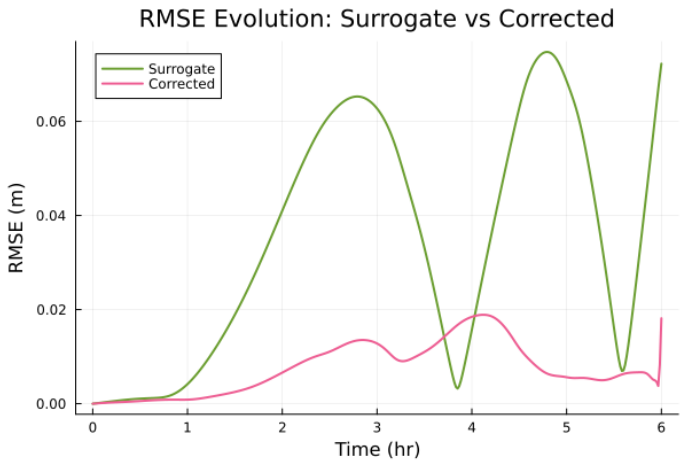


Figure 4.9: Temporal RMSE for Surrogate 1 baseline correction.

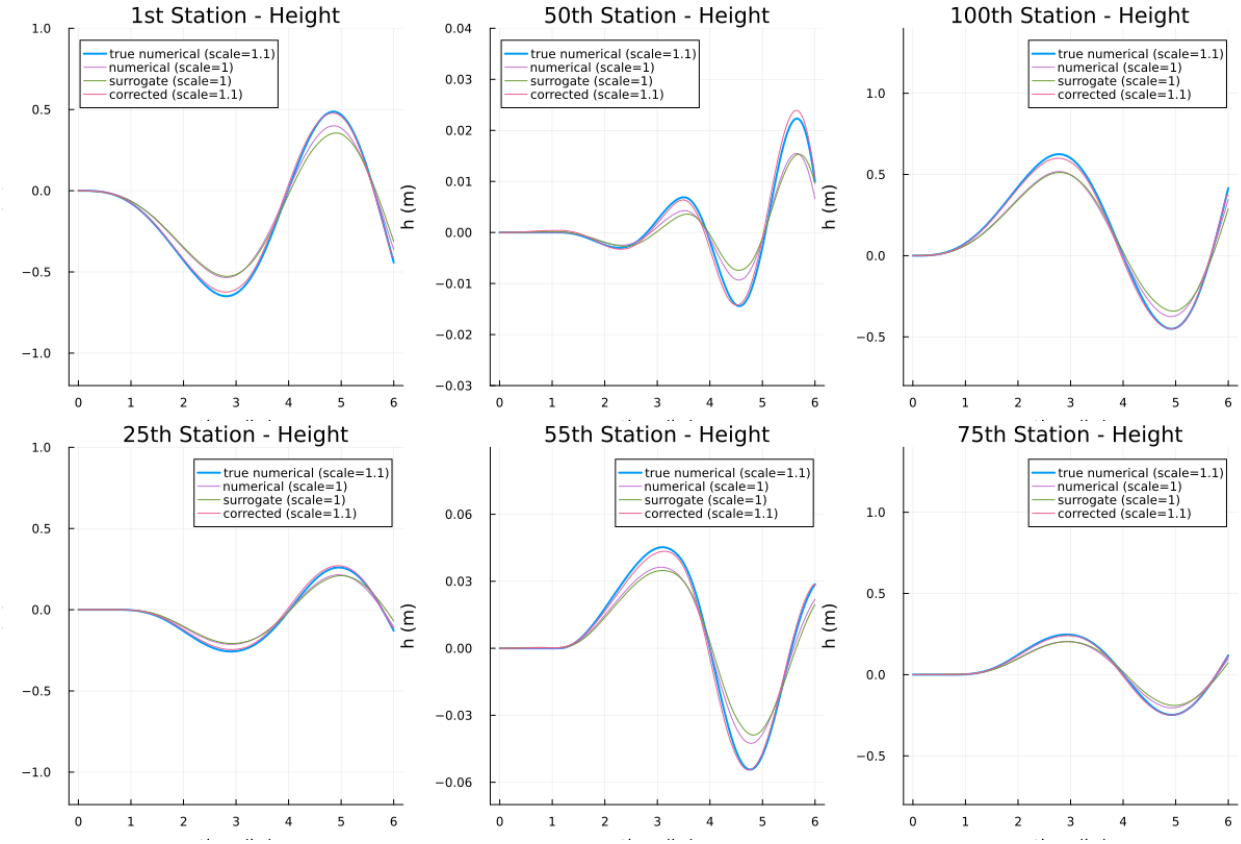


Figure 4.10: Multi-station time series comparison for Surrogate 1

Figure 4.10 illustrates this effect at representative stations along the domain. At the left and right boundaries (stations 1 and 100), the uncorrected surrogate (scale 1.0) shows a clear unrolling error relative to the numerical solution at the same scale: phase drift and amplitude damping become visible after several hours, which is expected in regions where reflections and shallow depths amplify the dynamics. In contrast, for interior stations such as 25 and 75, the surrogate trajectory appears almost perfect to the eye, because the local surge amplitudes are much smaller and the absolute errors remain on the order of a few millimetres.

When the wind scale is corrected to 1.1, the corrected surrogate closely tracks the numerical solution at scale 1.1 at all these locations. The large boundary errors are substantially reduced, and even at the mid-domain station, the differences between corrected and true trajectories are barely noticeable at the plotting scale. The apparent 'worst' behaviour in the middle station in Figure 4.10 is mainly a visual artefact of the smaller axis limits rather than a genuinely larger error. Together with the spatial and temporal RMSE curves, these plots confirm that the correction effectively removes most of the systematic unrolling drift while preserving the underlying wave structure.

4.4.5 Robustness Across Surrogate Configurations

Finally, to assess robustness, we applied the Baseline correction configuration to all surrogate variants. The results, in Table 4.10, show that most surrogates benefit substantially from correction, with error reductions between about 65% and 78% for Surrogates 1, 3.1, and 3.2. However, for the less stable Surrogates 2.1 and 2.2, the same Baseline configuration actually degrades performance, indicating that a sufficiently accurate and well-behaved surrogate is a prerequisite for successful correction.

Surrogate	Corrected RMSE	Improvement
Surrogate 1 (short)	0.009293 m	+78.11%
Surrogate 2.1 (short)	0.038082 m	-21.93%
Surrogate 2.2 (long)	0.095109 m	-20.11%
Surrogate 3.1 (short)	0.011559 m	+65.65%
Surrogate 3.2 (long)	0.019927 m	+72.37%

Table 4.10: Performance of the baseline per-station correction configuration across different surrogate variants.

Surrogate	Corrected RMSE	Improvement
Surrogate 1 (short)	0.018382 m	+56.71%
Surrogate 2.1 (short)	0.013203 m	+57.73%
Surrogate 2.2 (long)	0.059856 m	+24.41%
Surrogate 3.1 (short)	0.048126 m	-43.02%
Surrogate 3.2 (long)	0.033340 m	+53.78%

Table 4.11: Performance of the reduced input correction configuration across different surrogate models.

A similar pattern appears for the Reduced Input configuration (Table 4.11): for some surrogates it still yields some improvements, but for others it amplifies errors. Together, these results highlight that while the correction approach is flexible, it is not completely surrogate-agnostic: configurations that work well for one surrogate may need to be adapted or retuned for another.

These experiments demonstrate that input-bias correction can substantially reduce surge height errors when paired with a sufficiently stable surrogate and an appropriate correction configuration. Per-station scale corrections with well-chosen feature sets are particularly effective, often recovering most of the 21% wind stress underestimation while maintaining stable long-horizon rollouts. At the same time, the mixed results across surrogate variants highlight that correction cannot fully compensate for poor surrogate quality, which motivates careful surrogate design and validation as a prerequisite for applying this methodology to more complex coastal models

Chapter 5

Discussion and Analysis

This work investigated AI-learned input-bias correction in surrogate storm-surge models, focusing on correcting the systematic underestimation of wind stress. The two-stage framework combined fast neural surrogates trained on biased numerical model outputs with lightweight neural correction networks trained on true dynamics. The results show that the correction models perform successfully. For the Burgers' equation test case, the correction networks significantly reduced the error of the surrogate trained on deliberately biased input, achieving stable forecasts in this setting. For the 1D shallow-water model, corrections effectively compensated for a realistic 21% wind stress underestimation, improving the surrogate predictions and maintaining stability across operationally relevant forecast horizons.

However, limitations remain. The quality and stability of the baseline surrogate strongly influence correction performance: a poorly trained or unstable surrogate hinders correction effectiveness and can introduce instability. Furthermore, corrections converged to spatially varying scaling factors, despite the uniform input bias, which reflects local dynamical sensitivities and the spatial structure of surrogate residuals. Such variability indicates that global corrections applied uniformly are insufficient for complex coastal systems with heterogeneous bathymetry and boundary interactions. Moreover, the correction network partially compensates for surrogate unrolling errors, addressing both the input bias and the model inaccuracies.

Applying this methodology to full 2D or 3D coastal models and real observational data will demand advances in noise robustness, multi-station data assimilation under sparse conditions, and thorough validation against operational atmospheric forcing. Despite these challenges, the large computational gains of surrogates combined with learned input corrections present a practical preprocessing tool to enable rapid ensemble storm surge forecasting, supporting timely decision-making by authorities such as Rijkswaterstaat and coastal planners. Ethical considerations are essential for responsible deployment, which would require human oversight, quantification of uncertainty, and fallback to physical solvers to maintain trust and reduce risks.

Reflection on Limitations and Extensions

This study has several notable strengths. A carefully controlled experimental design allowed the key assumptions to be tested in isolation, where complementary evaluation metrics were applied systematically. The analysis was further reinforced by the use of two physically representative model systems. In addition, multi-step rollout assessments offered valuable insight into stability and error accumulation, both of which are critical for operational time scales.

Looking ahead, future research should focus on developing multi-dimensional models that incorporate realistic and complex bathymetry and boundary conditions typical of operational coastal environments. Future work should also incorporate observational noise and address realistic data sparsity through sparse assimilation, thereby improving the robustness and operational relevance of these models. Furthermore, creating more sophisticated correction architectures that can learn spatial-temporal dependencies, beyond simple scalar or per-station scale factors, will allow us to capture complex nonlinear bias patterns more effectively.

Additionally, it is crucial to connect these corrections with real atmospheric forecast datasets and evaluate their transferability across diverse storm events and geographic regions. Finally, exploring uncertainty quantification and robust deployment strategies, possibly through ensemble-based correction uncertainty estimation, will significantly strengthen the operational potential of this framework.

Resource limitations restricted most of the experiments to synthetic 1D settings with moderate model sizes. Due to computational constraints such as longer runtimes on personal computing hardware, the ability to conduct broader hyperparameter searches was restricted, leading to a reliance on relatively small models. All experiments were conducted on a personal workstation, which is a practical constraint that should be acknowledged when interpreting the results.

Conclusion

This project demonstrates the feasibility and practical benefits of using AI to correct input biases in fast surrogate storm-surge models. By systematically addressing the underestimation of wind stress biases present in the surrogate training data, small correction networks can greatly enhance the accuracy and stability of ensemble forecasts over relevant timescales. Most importantly, this method maintains the computational speed required for real-time applications.

While challenges remain in the surrogate design and the multi-dimensional applications, this approach offers a promising enhancement to current physics-based and data assimilation methods. It effectively combines the efficiency of machine learning with the interpretability and robustness of physical modelling. This foundation paves the way for the further development of improved graph neural network surrogates for the Dutch Continental Shelf Model (DCSM-FM), which can be operationally deployed to help mitigate coastal flood risks.

Bibliography

[1] Contributors. Burgers' equation. https://en.wikipedia.org/wiki/Burgers%27_equation. Accessed: 2025-11-30.

[2] Deltares, Firmijn Zijl. 3D Dutch Continental Shelf Model - Flexible Mesh. https://www.deltares.nl/en/expertise/projects/3d-dutch-continental-shelf-model-flexible-mesh, 2024. Accessed: 2025-11-30.

[3] Randall J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge Texts in Applied Mathematics. Cambridge University Press, 2002.

[4] Firmijn Zijl, Tammo Zijlker, Stendert Laan, and Julien Groenenboom. 3D DCSM-FM: a sixth-generation model for the NW European Shelf: 2022 release. Report, Deltares, 2023. https://www.deltares.nl/expertise/publicaties/3d-dcsm-fm-a-sixth-generation-model-for-the-nw-european-shelf-2022-release.

Appendix A. Additional Plots

Additional Results Model 2: Shallow Water Equations with Wind Forcing

Surrogate Models Trained over Short Trajectories

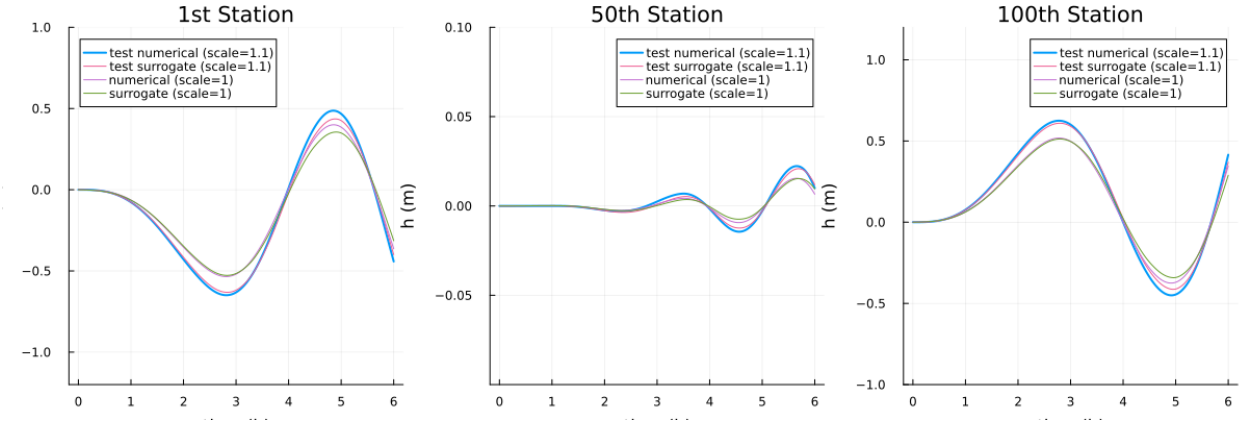


Figure 5.1: Surrogate 1 Unrolling with Different Scales

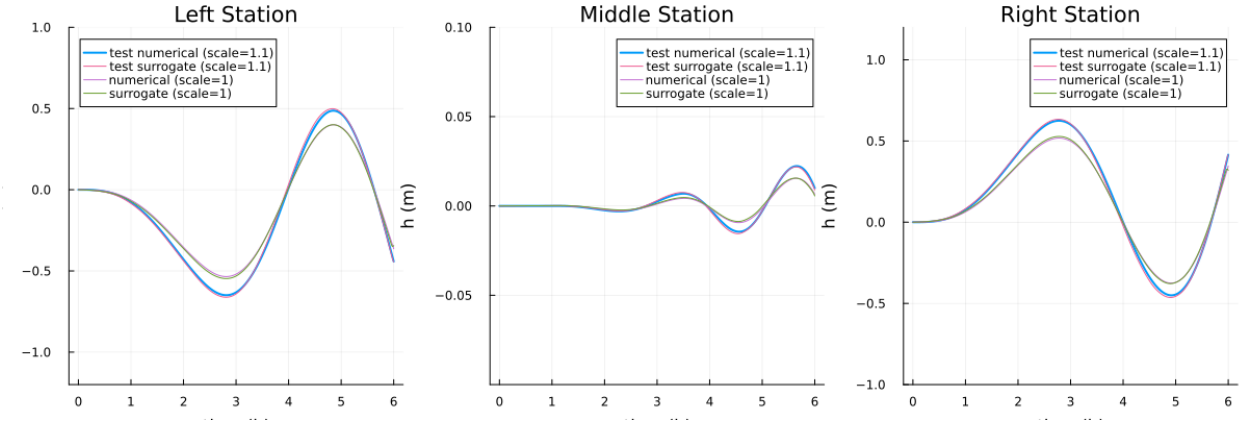


Figure 5.2: Surrogate 3.1 Unrolling with Different Scales

Surrogate Models Trained over Longer Trajectories

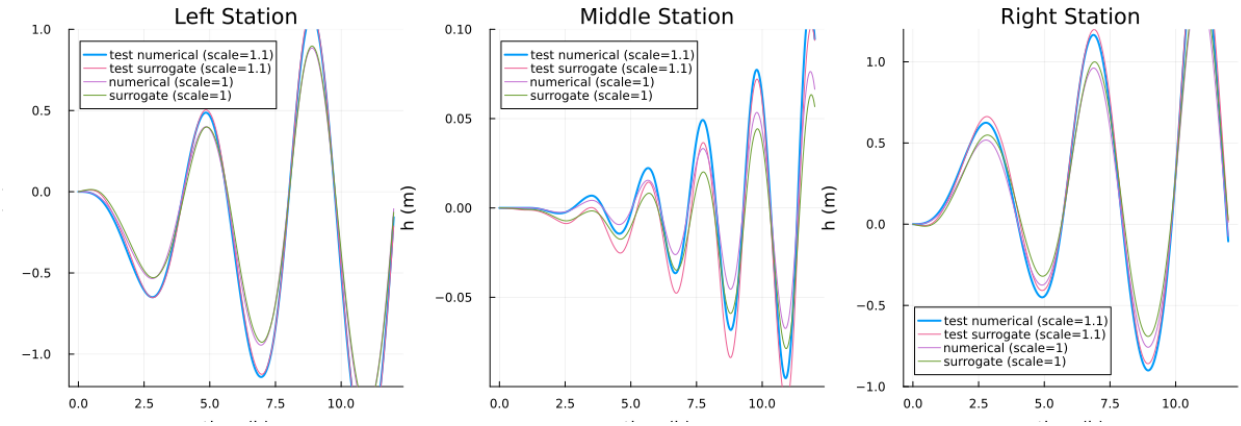


Figure 5.3: Surrogate 2.2 Unrolling with Different Scales

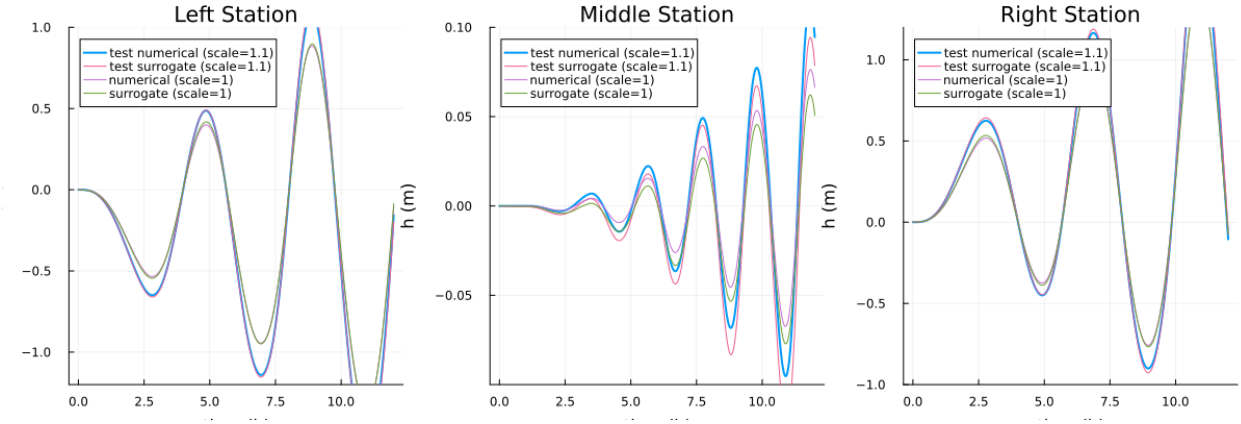


Figure 5.4: Surrogate 3.2 Unrolling with Different Scales

Reduced Input Correction Model for Surrogate 1

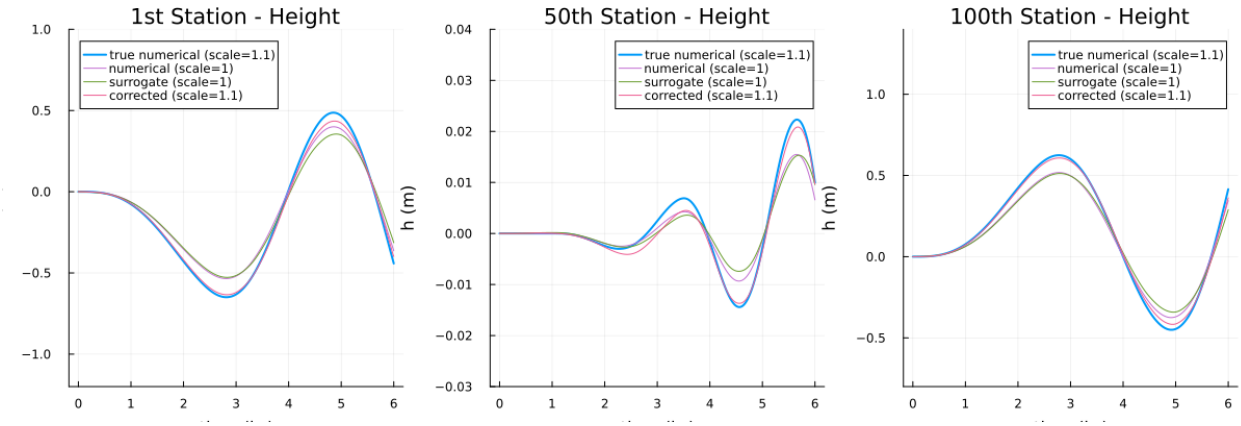


Figure 5.5: Surrogate 1, Reduced Input Correction Unrolling

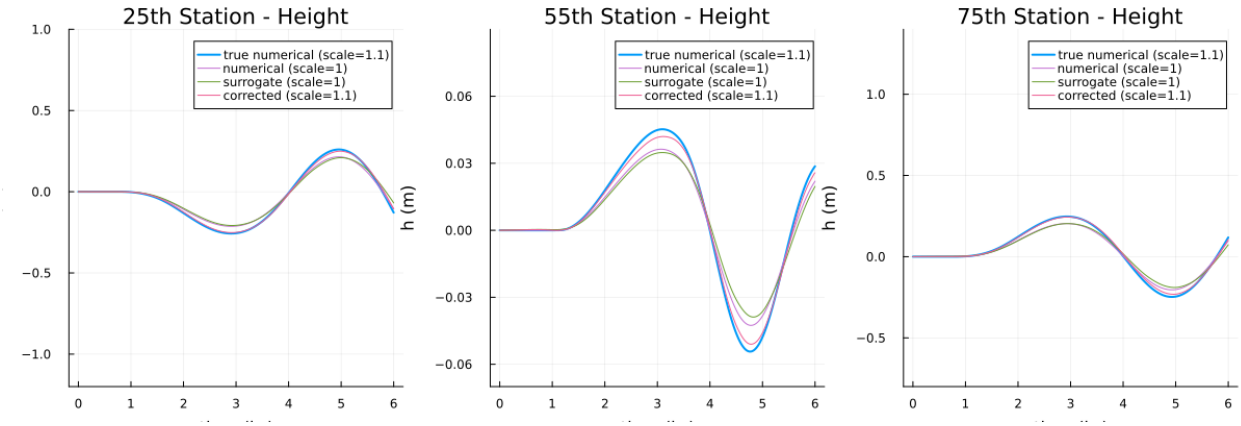


Figure 5.6: Surrogate 1, Reduced Input Correction Unrolling

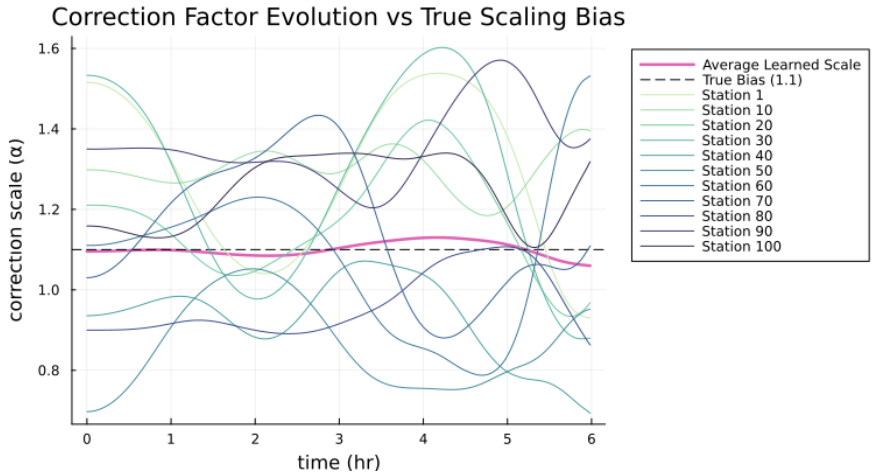


Figure 5.7: Surrogate 1, Reduced Input Correction Scale Time Series

Minimal Per-Station Correction Model for Surrogate 1

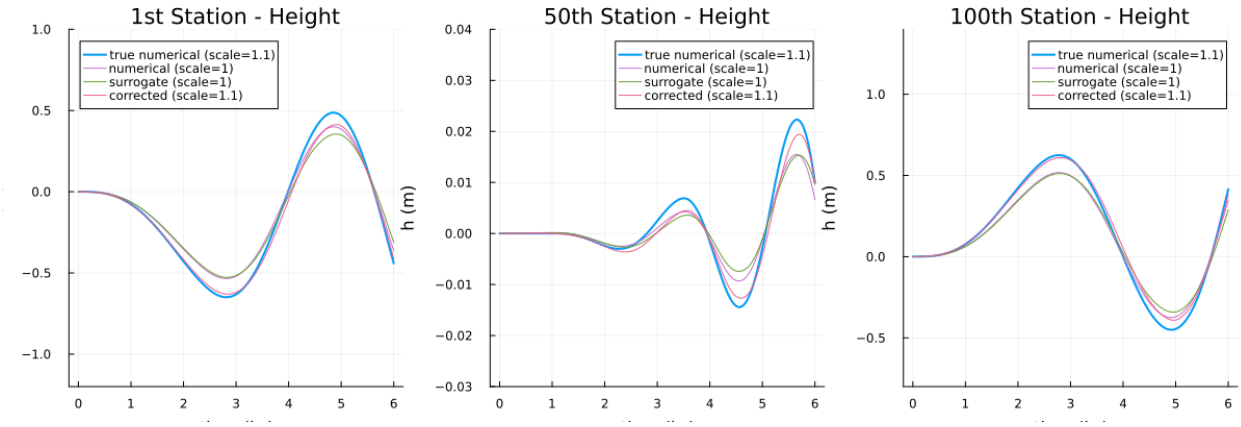


Figure 5.8: Surrogate 1, Minimal Per-Station Correction Unrolling

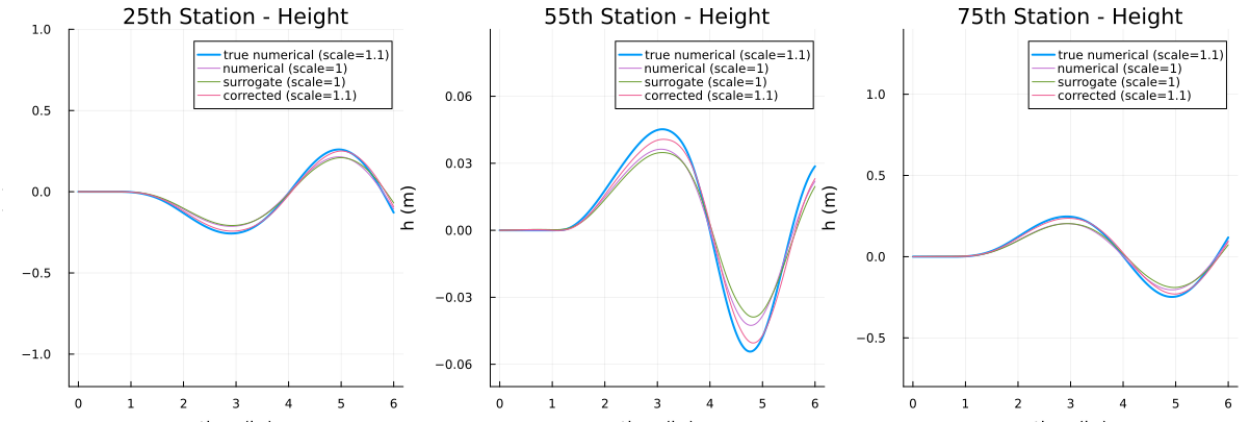


Figure 5.9: Surrogate 1, Minimal Per-StationCorrection Unrolling

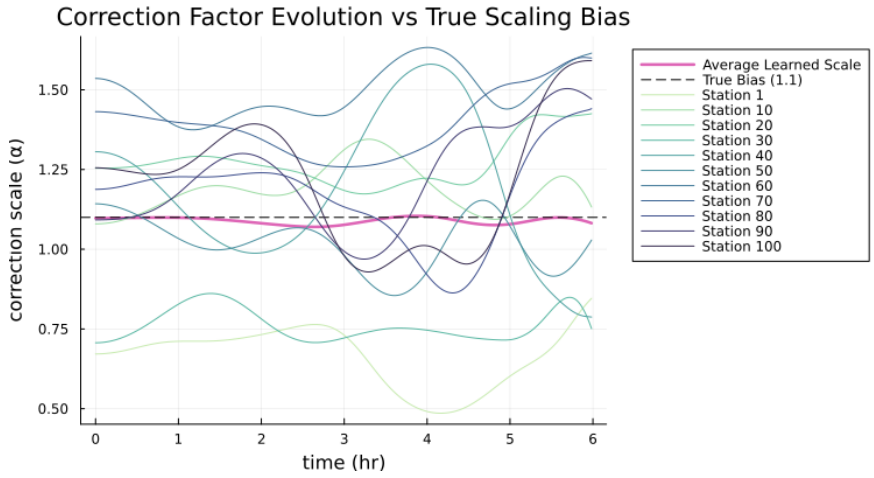


Figure 5.10: Surrogate 1, Minimal Per-Station Correction Scale Time Series

Baseline Correction Model for Surrogate 3.2

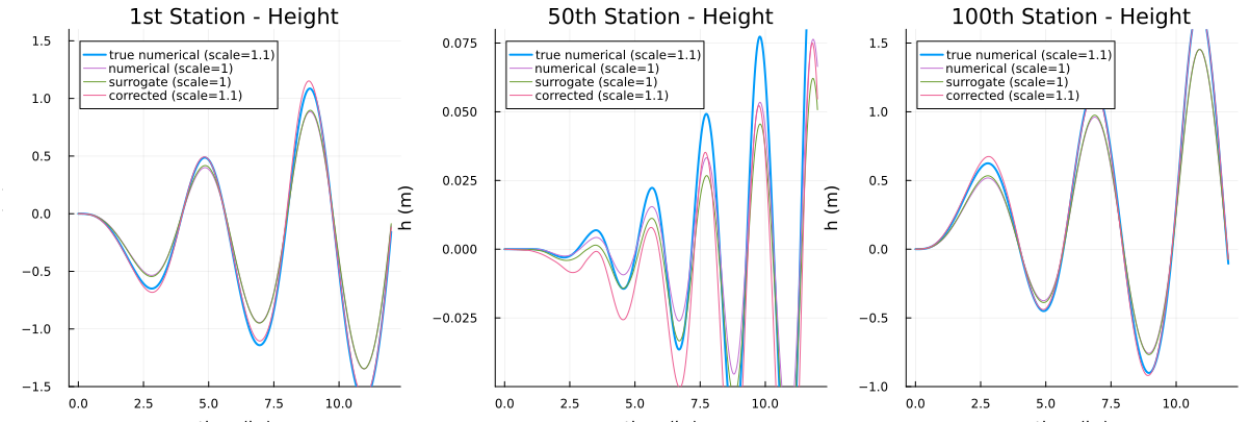


Figure 5.11: Surrogate 3.2, Baseline Correction Unrolling

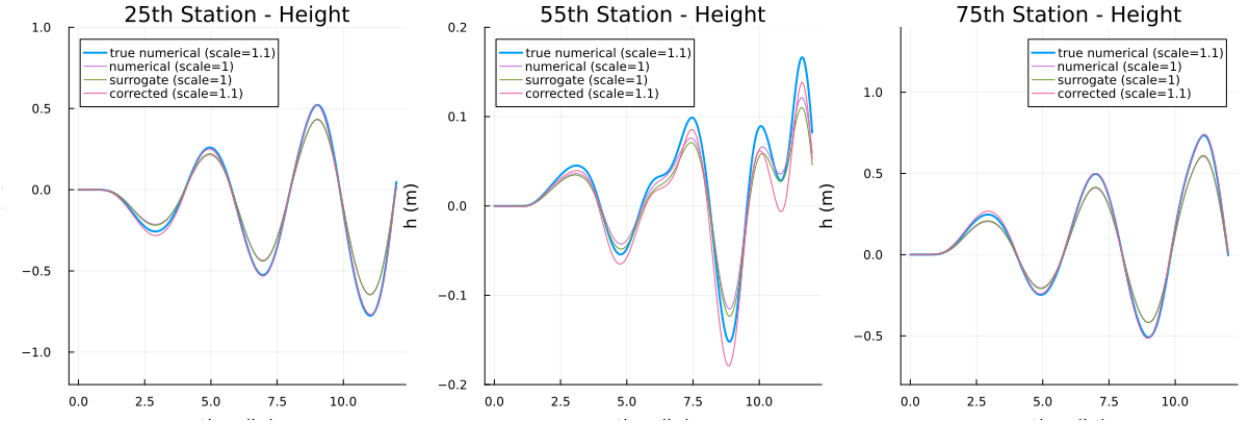


Figure 5.12: Surrogate 3.2, Baseline Correction Unrolling

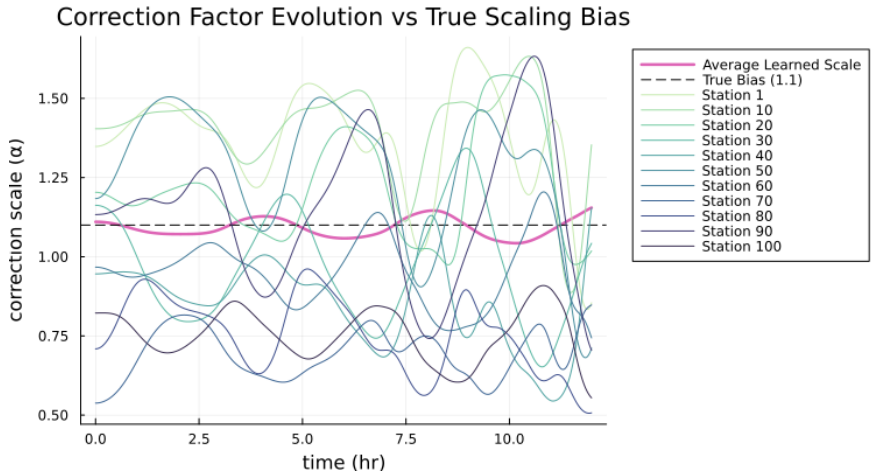


Figure 5.13: Surrogate 3.2, Baseline Correction Scale Time Series

Appendix B. Github Repository

The project's complete experiments and implementation are detailed in a public GitHub repository, which includes Jupyter notebooks demonstrating the described results.

Repository URL: https://gitshare.me/repo/saravlez