Imperial College London
Department of Earth Science & Engineering

Advanced numerical and statistical techniques to assess erosion and flood risk in coastal zones

Mariana Cristina de Andrade Clare

Supervised by
Prof. Matthew D. Piggott
Prof. Colin J. Cotter
Dr Catherine Villaret

Submitted in part fulfilment of the requirements for the degree of
Doctor of Philosophy in Computational Geoscience

January 2022

Statement of Originality

I hereby declare that the contents of this thesis are my own work and that any external resources that I have used have been clearly referenced in the text.

Mariana Cristina de Andrade Clare

Imperial College London, 28th January 2022

Copyright Declaration

Abstract

Throughout history, coastal zones have been vulnerable to the dual risks of erosion and flooding. With climate change likely to exacerbate these risks in the coming decades, coasts are becoming an ever more critical location on which to apply hydro-morphodynamic models blended with advanced numerical and statistical techniques, to assess risk.

We implement a novel depth-averaged hydro-morphodynamic model using a discontinuous Galerkin based finite element discretisation within the coastal ocean model *Thetis*. Our model is the first with this discretisation to simulate both bedload and suspended sediment transport, and is validated for test cases in fully wet and wet-dry domains. These test cases show our model is more accurate, efficient and robust than industry-standard models. Additionally, we use our model to implement the first fully flexible and freely available adjoint hydro-morphodynamic model framework which we then successfully use for sensitivity analysis, inversion and calibration of uncertain parameters. Furthermore, we implement the first moving mesh framework with a depth-averaged hydro-morphodynamic model, and show that mesh movement can help resolve the multi-scale issues often present in hydro-morphodynamic problems, improving their accuracy and efficiency.

We present the first application of the multilevel Monte Carlo method (MLMC) and multilevel multi-fidelity Monte Carlo method (MLMF) to industry-standard hydro-morphodynamic models as a tool to quantify uncertainty in erosion and flood risk. We use these methods to estimate expected values and cumulative distributions of variables which are of interest to decision makers. MLMC, and more notably MLMF, significantly reduce computational cost compared to the standard Monte Carlo method whilst retaining the same level of accuracy, enabling in-depth statistical analysis of complex test cases that was previously unfeasible.

The comprehensive toolkit of techniques we develop provides a crucial foundation for researchers and stakeholders seeking to assess and mitigate coastal risks in an accurate and efficient manner.

Acknowledgements

I would like to thank my academic supervisors, Matthew Piggott and Colin Cotter, for their support and guidance during my PhD. They have provided invaluable contributions to the work in this thesis and I always left our meetings with new ideas and motivation. I am also grateful to my industrial supervisor, Catherine Villaret, for her insight into the *Sisyphe* model, without which many parts of this PhD would have been much more difficult. Great thanks are also due to Stephan Kramer, who was a tireless source of advice and technical assistance throughout my PhD, especially with regards to the *Thetis* model.

I would also like to express my gratitude to current and former members of the Mathematics of Planet Earth CDT and Imperial College's Computational Geoscience and Engineering Group for the many enjoyable lunch breaks and fruitful academic discussions. Particular thanks are due to Joe Wallwork and James Percival for our collaborations.

My PhD may have been based at Imperial, but I have had the pleasure of collaborating with researchers at many other institutions. I would like to thank especially Anastasia Fragkou, Athanasios Angeloudis, Tim Leijnse and Hilary Weller. I am also grateful to the members of the *Firedrake* and *Thetis* development teams, both at Imperial and other institutions, who have always responded helpfully and quickly to my many code queries.

I also gratefully acknowledge the Mathematics of Planet Earth CDT and the EPSRC [grant number EP/R512540/1] for the funding they provided to enable me to pursue this PhD.

Last but not least, I would like to thank my family and friends for their support and patience over the last few years, especially during the pandemic over the last eighteen months.

Contents

List of Tables

List of Figures

Notation

The following are common notation used throughout this thesis.

$\bar{\cdot}$	Depth-averaged variable
$\bar{\cdot}^t$	Time-averaged variable
$\hat{\cdot}$	Statistical estimator
∇_h	2D divergence operator $(\partial/\partial x, \partial/\partial y)$
$\lceil \cdot \rceil$	Ceiling operator (returns smallest integer $\geq$ to the quantity inside it)
c	Sediment concentration
c_{b*}	Equilibrium near-bed sediment concentration
C_h	Quadratic drag coefficient (2.39)
d_{50}	Median sediment diameter
δ	Wetting-and-drying parameter
δ^*	Height of bedload zone
D_b	Deposition flux
E_b	Erosion flux
ϵ_s	Diffusivity coefficient
η	Water elevation
$\mathbf{F}$	External forces acting on the fluid
$F(u, \mathbf{m})$	Forward model in adjoint method
F_{corr}	Correction factor (2.22)
$\mathbf{g}$	Gravity vector [if not bold then scalar]
Γ	Union of edges of mesh
h	Total water depth $(h = \eta - z_b)$
$\tilde{H}$	Modified wetting-and-drying water depth (2.59)
J	Output functional
κ	Von Kármán constant $(\kappa = 0.4)$
k_s	Nikuradse friction height
k_s'	Grain roughness height
l_μ	Minimum level considered in MLMC and MLMF estimator
$\mathbf{m}$	Set of uncertain parameters in adjoint method
m_f	Morphological scale factor
n_m	Manning friction coefficient
$\mathbf{n}$	Outward pointing unit normal to the surface
ν	Kinematic eddy viscosity (used in hydrodynamics)
ν_m	Kinematic molecular viscosity (used in morphodynamics)
Ω	Domain on which mesh is defined
$d\Omega$	Domain boundary
p	Pressure
p'	Bed sediment porosity
ϕ	Sediment slide angle of repose
ψ	Finite element method test function
Ψ	Skin friction coefficient (2.38)
$\mathbf{Q_b}$	Bedload transport
R	Rouse number
ρ_s	Sediment density
ρ_w	Water density
t	Time
τ	Stress

τ_b	Bed shear stress (2.37)	w_s	Sediment settling velocity (2.30)
τ_c	Critical shear stress (2.35)	z_b	Bed/Bedlevel
θ_{cr}	Critical Shields parameter (2.36)	ζ	Norm smoothing parameter
$\mathbf{u}$	Velocity	$\mathring{\zeta}$	Sediment slide length scale
$\mathbf{x}$	Position ($= (x, y, z)$)		

CHAPTER 1

Introduction

Throughout history, coastal zones have been attractive regions for human settlement and leisure due to their abundant resources and the possibilities they offer for commerce and transport. A recent report estimates that 680 million people live in low-lying coastal zones (less than 10 m above sea-level), and forecasts that this number will increase to more than 1 billion by 2050 (Pörtner et al., 2019). Nevertheless, living in coastal zones has always come with the dual risks of coastal erosion and flooding. These risks will likely be exacerbated in the coming decades due to climate change, leading to hydrodynamic changes in coastal zones including sea level rise and changes to the strength and frequency of storms. Vousdoukas et al. (2020) estimate that half of the world's sandy beaches could disappear by 2100 due to future sea level rise, storm changes and ambient coastal erosion (for reference, currently one third of the global coastline consists of sandy beaches).

The economic cost of these coastal hazards is also likely to increase dramatically. In Europe, for example, annual damage from coastal flooding already costs €1.3 billion and, by 2100, this cost is estimated to increase by two to three orders of magnitude (Vousdoukas et al., 2018). Government agencies forecast that large financial resources will need to be spent to mitigate against these hazards. For example, in the UK, the Environment Agency estimates that over £1 billion must be invested in coastal and river defences every year for the next 50 years merely to maintain the current risk level from flooding and erosion (Environment Agency, 2020). However, estimates of coastal erosion and flooding risk are often uncertain (see Athanasiou et al., 2020) and, therefore, the coastal zone is becoming an ever more critical location for the application of advanced modelling techniques. This is especially challenging because models must solve both hydrodynamic and sediment transport processes together with their two-way coupled interactions.

For several decades, complex hydro-morphodynamic models have been developed to simulate hydrodynamics, sediment transport and bed morphology in both fluvial and coastal environments. These models can be either one-dimensional (1D), two-dimensional (2D) or three-dimensional (3D), and are solved using numerical methods such as finite difference, finite volume and finite element methods (Amoudry, 2008; Papanicolaou et al., 2008; Amoudry and Souza, 2011). 1D hydro-morphodynamic models are the computationally cheapest, as they use the shallow water approximation and assume the flow is uni-directional. In general, these models solve the 1D Saint-Venant equations (which determine the elevation and velocity of the hydrodynamics) coupled with the Exner equation (which determines the change in bedlevel) (Papanicolaou et al., 2008). They cannot capture velocities in the cross-stream direction and are hence inappropriate for modelling the complex multi-dimensional cases considered in this thesis. 2D

hydro-morphodynamic models are more complex and either use the shallow water approximation (2DH) or assume uni-directional flow (2DV). Of the two 2D model types, 2DH (hereafter referred to as depth-averaged) is the most appropriate for this thesis because it captures velocity in both the streamwise and cross-stream directions, as well as planview geometries in the horizontal plane. In general, 2DH models solve the 2D shallow water equations, the sediment concentration equation and the Exner equation (Wu, 2007; Amoudry, 2008). Finally, 3D models are the most complex type of hydro-morphodynamic models, as they solve the 3D hydrodynamic problem coupled with the same morphodynamic equations as the depth-averaged model. This means 3D models have the potential to be more accurate, but are also more computationally expensive and may require more input data and calibration, as well as having higher verification costs.

In Table 1.1, we present a summary of the key details of commonly-used hydro-morphodynamic models, and refer the reader to Amoudry (2008), Papanicolaou et al. (2008) and Amoudry and Souza (2011) for more details on these and other similar models. Throughout this thesis, we choose to use 2D depth-averaged models because, in the problems we consider, the flow variation in the cross-stream direction is important and close to columnar.

Model name	**Dimension**	**Numerical Method**	**Mesh type**
Mike Hydro River (DHI, 2017)	1D	Finite Difference	Unstructured
Mike 21 (Warren and Bach, 1992)	2D	Finite Volume	Unstructured
XBeach (Roelvink et al., 2015)	2D	Finite Volume	Curvilinear staggered grid
Delft3D (Deltares, 2014)	Both 2D and 3D	Finite Difference and Finite Volume	Orthogonal curvilinear coordinate system with σ mesh in vertical
Telemac-Mascaret (Hervouet, 1999)	Both 2D and 3D	Finite Element	Unstructured
Fast3D (Landsberg et al., 1998)	3D	Finite Volume	Non-staggered curvilinear
ROMS (Warner et al., 2008)	3D	Finite Difference	Orthogonal curvilinear coordinate system with σ mesh in vertical

TABLE 1.1. Key properties of commonly-used hydro-morphodynamic models.

Despite this multitude of model approaches, Syvitski et al. (2010) argue that more accurate and faster hydro-morphodynamic models are needed. Hence, one of the main objectives of this thesis is to work towards an accurate and fast hydro-morphodynamic model using a variety of advanced numerical and statistical techniques. In Chapter 2, we present a novel flexible 2D depth-averaged coupled hydro-morphodynamic model developed within *Thetis*, a finite element coastal ocean modelling system (see Kärnä et al., 2018) built using the *Firedrake* code-generation framework (see Rathgeber et al., 2017). Our model improves upon existing state-of-the-art models by using a discontinuous Galerkin based finite element discretisation (DG), available in *Thetis*, which is second-order accurate in space (Kärnä et al., 2018). Note that we use P1-DG elements (*i.e.* piecewise linear discontinous finite elements). To the best of our knowledge, our model is the first full DG hydro-morphodynamic model to simulate both bedload and suspended sediment transport. Although a similar discretisation approach is presented in

both Kubatko et al. (2006) and Tassi et al. (2008), neither include suspended sediment transport. DG has several advantages, which will be discussed in Section 2.1. In particular, it works especially well for advection-dominated problems, like sediment transport (Dawson et al., 2004); has a stable formulation using upwind fluxes (Shu, 2009); and is locally mass conservative, meaning sediment is conserved on an element-by-element level, a desirable property when coupling hydrodynamics and sediment transport models (Dawson et al., 2004).

In addition to these benefits, we implement novel functionalities in our model which are unavailable in existing industry-standard frameworks. For example, in Chapter 3, we build the first freely available adjoint coupled hydro-morphodynamic model framework. The adjoint method computes gradients of model outputs with respect to input parameters which are immensely useful for sensitivity analysis and for the calibration and inversion of parameter values. The main advantage of this adjoint approach over other methods is that only one adjoint evaluation is ever required to compute the sensitivity of an output quantity, allowing any number of multi-dimensional parameters to be considered simultaneously with almost no effect on the computational cost. Although the adjoint method has been applied to a limited extent to the morphodynamic component of the Telemac-Mascaret model described in Section 2.3.1 (Kopmann et al., 2012; Merkel et al., 2013, 2016), to the best of our knowledge, the adjoint method has never been applied to any fully coupled hydro-morphodynamic model. In our work, we make use of the inbuilt library *pyadjoint* in *Firedrake* to automatically derive and solve the adjoint equations. This library makes *Firedrake*-based models like ours ideally suited to the application of the adjoint method, and the flexibility it provides means any parameter can be assessed with respect to any output functional without further code implementation. Thus, our adjoint framework can be invaluable to a variety of users and stakeholders in understanding and mitigating the impacts of coastal hazards.

We further improve the capabilities of our model relative to existing models in Chapter 4, where we present the first application of mesh movement methods to a depth-averaged hydro-morphodynamic model, taking advantage of the *Firedrake* framework's ability to solve the complex non-linear PDEs that implementations of these methods often require (McManus et al., 2017). Hydro-morphodynamic models must often solve problems on complex and fundamentally multi-scale domains, and mesh movement methods can provide a solution to the multi-scale issues this causes. This is because mesh movement methods typically operate by fixing the mesh topology and then dynamically moving the (fixed number of) mesh nodes, thus allowing different regions of the domain to vary between low and high resolution as flow structures pass through them (Budd et al., 2009). Previous works have applied the alternative mesh adaptation method of h-refinement to hydro-morphodynamic models (for example Mayne et al. (2002); Delandmeter (2017); Benkhaldoun et al. (2013)). With h-refinement, the mesh topology is substantially altered with cells locally created or destroyed and mesh connectivity altered (Piggott et al., 2005, 2009). The advantages of mesh movement over h-refinement are discussed in detail in Section 4.1, where we also discuss the advantages our hydro-morphodynamic model has over the hydro-morphodynamic models on which h-refinement has been applied.

Whilst hydro-morphodynamic models may produce accurate results for a given idealised scenario, there is a large degree of uncertainty associated with them when they are applied to the real-world (see, for example, Unguendoli (2018) and Villaret et al. (2016)). This can either be due to natural variability (aleatory uncertainty) or incomplete knowledge (epistemic uncertainty) of various physical, empirical and numerical closure related parameters in both the hydrodynamic and morphodynamic equations (see, for example, Apel et al. (2004)). The most straightforward approach to assess uncertainty is to apply the standard form of the Monte Carlo method. Using this method, the estimator, $\hat{P}$, for the expectation of a variable $P(\omega)$ with respect to an uncertain input parameter ω is given by

$$\hat{P} = \frac{1}{N} \sum_{n=1}^{N} P\left(\omega^{(n)}\right) \tag{1.1}$$

where $\omega^{(n)}$ are N independent samples taken from the distribution of the uncertain input parameter. The variance of this estimator is $\mathrm{Var}(P)/N$ and hence its root mean square error is $\mathcal{O}(1/\sqrt{N})$, meaning that achieving an accuracy ϵ requires $\mathcal{O}(1/\epsilon^2)$ samples. Thus, very large numbers of samples are typically needed for accurate Monte Carlo estimates and each sample requires a separate model run. Although these runs can be performed in an 'embarrassingly' parallel sense, this approach requires a substantial computational resource commitment, especially given the high computational cost of running hydro-morphodynamic simulations. The Monte Carlo method has already been applied to a range of hydro-morphodynamic models, including the coastal ocean model XBeach described in Section 2.3.2 (Pender and Karunarathna, 2013; Dissanayake et al., 2014; Simmons et al., 2017; Harris et al., 2018). However, the high computational cost has limited the scenarios for Monte Carlo based studies, and researchers have in general been forced to use simplified models for more complex and long-term test cases. A stark example of this is Callaghan et al. (2013) whose use of the simpler 1D semi-empirical model SBeach for their Monte Carlo simulation took 40 days on a single processor, and they estimate that even using the 1D version of XBeach on the same processor would take 4.5 millennia.

Research on alternative methods to assess uncertainty is both highly pertinent and ongoing. This includes numerical approaches, such as the adjoint method discussed above, and statistical approaches, which have the advantage that they are rarely model specific. In this thesis, we focus on Monte Carlo type methods, like the *multilevel* Monte Carlo method (MLMC) first introduced in Giles (2008) and the *multilevel multifidelity* Monte Carlo method (MLMF) first introduced in Geraci et al. (2015). This is the first time either MLMC or MLMF have been applied to hydro-morphodynamic models, and is one of the very first applications of MLMF in any field. Like the standard Monte Carlo method, MLMC and MLMF quantify uncertainty and risk by computing the expected values of key output variables with respect to uncertain parameters, as well as the cumulative distribution functions of the output variables. However, they do so in a much more computationally efficient manner than the standard Monte Carlo method, whilst maintaining the same level of accuracy. MLMC reduces computational cost by accelerating the Monte Carlo method through the use of a hierarchy of level resolutions. In Chapter 5, we apply MLMC to the hydro-morphodynamic coastal model XBeach (described in Section 2.3.2). MLMF reduces cost further by combining MLMC with a multifidelity multi-model approach that takes advantage of both the

accuracy of high fidelity models and the computational efficiency of low fidelity models. In Chapter 6, we apply MLMF to coastal zone cases using XBeach as our high fidelity model and a reduced physics coastal model, SFINCS (described in Section 2.3.3), as our low fidelity model. The computational efficiency improvements of both MLMC and MLMF mean that in-depth statistical analysis of previously impractical complex test cases is now feasible.

Following this introduction on the background and motivation for this thesis, we summarise its objectives below:

Thesis objectives

- Develop a hydro-morphodynamic model capable of simulating sediment transport through both suspended sediment and bedload processes in both fully wet domains and wet-and-dry domains, while taking into account gravitational and helical flow effects;
- Improve model accuracy and computational efficiency by implementing mesh movement methods with our model;
- Assess uncertainty and risk in our model through using the adjoint method for sensitivity analysis, and inversion and calibration of uncertain input parameters;
- Apply novel Monte Carlo type methods to existing industry-standard models to assess uncertainty and risk in a much more computationally efficient manner than existing practices.

The remainder of this thesis is structured as follows: in Chapter 2, we detail and validate our hydro-morphodynamic model developed within *Thetis* and briefly summarise other models used in this thesis; in Chapter 3, we apply the adjoint method to our model to assess uncertainty; in Chapter 4, we apply mesh movement methods to our model to improve its accuracy and efficiency; in Chapter 5, we apply the *multilevel* Monte Carlo method (MLMC) to industry-standard hydro-morphodynamic models to assess uncertainty in an accurate and computationally efficient manner; in Chapter 6, we introduce multifidelity approaches and assess uncertainty in industry-standard models using the *multilevel multifidelity* Monte Carlo method (MLMF) to increase efficiency further; and, finally, in Chapter 7, we conclude this thesis and suggest avenues for further work.

Publications as a result of work in this PhD Thesis

Here we present an overview of the publications that have either been published or are under review as a result of work in this thesis. Where appropriate, please refer to the 'Publications and Disclaimer' section at the beginning of each chapter for more details on the overlap between the publication and the work in that chapter.

Published. A significant proportion of the work in Chapter 2 of this thesis has been published in the following paper:

> Clare, M. C., Percival, J. R., Angeloudis, A., Cotter, C. J., & Piggott, M. D. (2021). 'Hydro-morphodynamics 2D modelling using a discontinuous Galerkin discretisation'. Computers & Geosciences, 146, 104658. https://doi.org/10.1016/j.cageo.2020.104658

Under review. A significant proportion of the work in Chapter 3 has been submitted for publication to *Computers & Geosciences*, where it is currently under review. The preprint is:

> Clare, M. C., Kramer, S. C., Cotter, C. J., & Piggott, M. D. (2021). 'Calibration, inversion and sensitivity analysis for hydro-morphodynamic models through the application of adjoint methods.' *Under review at Computers & Geosciences* https://eartharxiv.org/repository/view/2599/

Published. A significant proportion of the work in Chapter 4 has been submitted for publication to *International Journal on Geomathematics*, where it has been accepted pending minor corrections. The preprint is:

> Clare, M. C., Wallwork, J. G., Kramer, S. C., Weller, H., Cotter, C. J., & Piggott, M. D. (2022). 'Multi-scale hydro-morphodynamic modelling using mesh movement methods'. GEM-International Journal on Geomathematics, 13(1), 1-39. https://doi.org/10.1007/s13137-021-00191-1

Under review. A significant proportion of the work in Chapter 5 has been submitted for publication to *Coastal Engineering*, where it is currently under review. The preprint is:

> Clare, M. C., Piggott, M., & Cotter, C. J. (2021). 'Assessing erosion and flood risk in the coastal zone through the application of the multilevel Monte Carlo method.' *Under review at Coastal Engineering* https://eartharxiv.org/repository/view/1956/.

CHAPTER 2

Hydro-morphodynamic models

Highlights

- Implementation of a new hydro-morphodynamic model within a code-generation framework;
- Novel use of discontinuous Galerkin based finite element methods to solve this system;
- Simulation of suspended and bedload transport with gravity and helical flow effects;
- Successful validation of test cases in both the fully wet domain and wet-dry domain.

Summary

The development of hydro-morphodynamic models to simulate sediment transport accurately is a challenging problem because they are highly complex due to the non-linear and coupled nature of sediment transport processes. Here, we implement our new hydro-morphodynamic model using a discontinuous Galerkin based finite element discretisation within the coastal ocean model *Thetis*, built using the code-generation framework *Firedrake*. This model simulates both bedload and suspended sediment transport, taking into account helical flow and gravitational effects. We consider the model for the fully wet domain and for the wet-dry domain separately, as a wet-dry interface adds an extra level of complexity. For validation purposes and to demonstrate model capability, we compare our model results against experimental data and results from the existing models Telemac-Mascaret and XBeach, which are briefly discussed. We consider two test cases in the fully wet domain – a migrating trench and a meandering channel – and show that our model is more accurate, as well as more efficient and robust, than the commonly-used Telemac-Mascaret model. We then consider the wet-dry domain test case of a tsunami-like wave hitting a sloping beach, and show that our model results agree well with experimental data and are at least as accurate as results obtained using the XBeach model.

Publications and Disclaimer

An early version of the fully wet version of our model in *Thetis* with the migrating trench test case (without benchmarking) was submitted to Imperial College London for examination as part of the author's MRes thesis:

Clare, M. (2018). 'Modelling sediment transport in and around offshore wind farms'. MRes thesis, Imperial College London, London, UK.

We draw heavily on this text in Section 2.2 and state clearly any parts which are new to this PhD thesis.

The work in this chapter has also been published/submitted in several publications during this PhD. The complete fully wet version of this model, together with the migrating trench and meandering channel test cases, was published in *Computers & Geosciences*:

Clare, M. C., Percival, J. R., Angeloudis, A., Cotter, C. J., & Piggott, M. D. (2021). 'Hydro-morphodynamics 2D modelling using a discontinuous Galerkin discretisation'. Computers & Geosciences, 146, 104658.

Note the study of meander test case is completely new work conducted in this PhD and was not submitted as part of the MRes.

Furthermore, the development of the wet-dry version of our model is completely new work conducted in this PhD. It is detailed in a paper submitted for publication to *International Journal on Geomathematics*, where it has been accepted pending minor corrections. The preprint of this paper is available here:

Clare, M. C., Wallwork, J. G., Kramer, S. C., Weller, H., Cotter, C. J., & Piggott, M. D. (2022). 'Multi-scale hydro-morphodynamic modelling using mesh movement methods'. GEM-International Journal on Geomathematics, 13(1), 1-39. https://doi.org/10.1007/s13137-021-00191-1

Finally, the tsunami-like wave test case in this chapter is a completely new work conducted in this PhD and has been submitted for publication to Computers & Geosciences, where it is currently under review. The preprint of this paper is available here:

Clare, M. C., Kramer, S. C., Cotter, C. J., & Piggott, M. D. (2021). 'Calibration, inversion and sensitivity analysis for hydro-morphodynamic models through the application of adjoint methods.' https://eartharxiv.org/repository/view/2599/

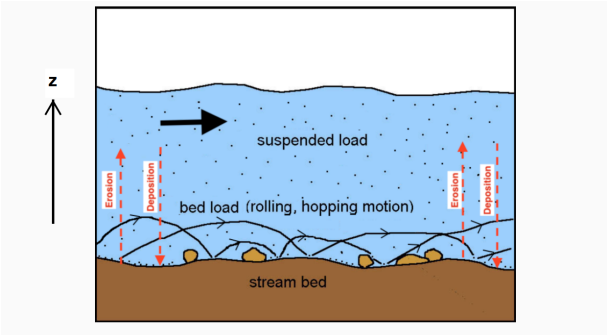


FIGURE 2.1. Diagram of hydrodynamic and sediment transport processes simulated by a full hydro-morphodynamic model, adapted from Geology Cafe (2015).

2.1 Introduction

Over the last 30 years, increasingly sophisticated hydro-morphodynamic models have been developed to simulate erosion and deposition of sediment in fluvial and coastal zones. Sediment can be transported by both suspended sediment transport and bedload transport. Figure 2.1 shows a diagrammatic representation of what a complete hydro-morphodynamic model should simulate. The equations that govern this transport can be derived using well-known mathematical and physical theory (Wu, 2007). As discussed in Chapter 1, we use depth-averaged models in this thesis because, in the test cases considered throughout, the flow is close to columnar and hence depth-averaging is a good compromise between cost and accuracy.

In this chapter, we develop a new depth-averaged hydro-morphodynamic model capable of simulating both suspended sediment and bedload transport, taking into account gravitational and helical flow effects. Furthermore, our model is capable of simulating cases in both fully wet domains and wet-dry domains, by using the wetting-and-drying scheme described in Kärnä et al. (2011). Hydro-morphodynamic models cannot be solved analytically and must be solved using numerical schemes, such as finite difference, finite volume and finite element methods (see Table 1.1 in Chapter 1). For our model, we use a discontinuous Galerkin finite element discretisation (DG) with piecewise linear elements implemented within the coastal ocean model *Thetis*, which is built within the code-generation framework *Firedrake* (Kärnä et al., 2018). The DG method has several advantages, including that it is well-suited to advection-dominated problems (Kärnä et al., 2018); is geometrically flexible; allows higher-order local approximations (Bastian et al., 2019); and is locally mass conservative, an advantage for coupling because sediment is conserved on an element-by-element level (Dawson et al., 2004). Other works use a DG approach (Kubatko et al., 2006; Tassi et al., 2008), but their hydro-morphodynamic models do not simulate suspended sediment transport. Thus, the model developed here has the significant novelty of being the first DG based hydro-morphodynamic model to include both bedload and suspended sediment transport.

As our model involves a novel implementation, we must validate it and demonstrate its capabilities by comparing its results with experimental data and results from other already existing and trusted models, including Telemac-Mascaret (Hervouet, 1999) and XBeach (Roelvink et al., 2015). For validation purposes, we consider test cases in both a fully wet domain and a wet-dry domain, where we use a

migrating trench and a meandering channel for the former, and a tsunami-like wave hitting a sloping beach for the latter. These validated test cases form the basis for the cases considered in Chapter 3 and Chapter 4. Along with the work in this chapter, these chapters will show that our model is a significant novel contribution to the field of hydro-morphodynamic modelling, but it cannot yet simulate short wavelength wind-driven waves. Hence, in this chapter, we also discuss two alternative existing models: XBeach, a widely-used depth-averaged hydro-morphodynamic model, and SFINCS, a reduced physics coastal flooding solver. Note that we use these models in Chapters 5 and 6 to simulate test cases with complex wave spectra.

The remainder of this chapter is structured as follows: in Section 2.2, we derive our hydro-morphodynamic model in *Thetis*; in Section 2.3, we describe the other hydro-morphodynamic models we use; in Section 2.4 and Section 2.5, we validate the test cases in a fully wet domain and a wet-dry domain, respectively, using experimental data and other model results; and, in Section 2.6, we conclude this chapter.

2.2 Implementing our hydro-morphodynamic model in *Thetis*

We begin this chapter by deriving the equations for our 2D depth-averaged hydro-morphodynamic model and implementing them in *Thetis*. The model equations for the fully wet domain are slightly different to those for the wet-dry domain because the wet-dry interface adds an extra level of complexity and requires the implementation of a wetting-and-drying scheme. In this section, we thus consider the two scenarios separately.

2.2.1 Model equations for fully wet domain

For the fully wet domain hydro-morphodynamic model, we follow the presentation and notation in Wu (2007), where more details can be found. We start by assuming that the sediment-water mixture is dilute, *i.e.* the sediment concentration is lower than 10%. This is standard practice for the problems we consider (see Wu, 2007) and means that the flow density ρ_w can be considered constant and equal to the standard density of water ($1000\,\mathrm{kgm}^{-3}$). Therefore the hydrodynamic component of the sediment-water mixture is governed by the (3D) Navier-Stokes equations for single phase flow

$$\nabla \cdot \mathbf{u} = 0, \tag{2.1}$$

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u} \otimes \mathbf{u}) = \frac{1}{\rho_w}\mathbf{F} - \frac{1}{\rho_w}\nabla p + \frac{1}{\rho_w}\nabla \cdot \tau, \tag{2.2}$$

where $\mathbf{u}$ is the velocity (and $\otimes$ represents the outer product), p the pressure, τ the stress tensor and $\mathbf{F}$ any external forces acting on the fluid. Here the only external force is the gravity acting on the fluid (because of the assumption above that the sediment particles are negligible relative to the fluid) and therefore $\mathbf{F} = -\rho_w \mathbf{g}$. The first equation represents the conservation of volume principle and the second equation represents the conservation of momentum principle.

Solving the equations in this form requires the resolution of the turbulent fluctuating motions within the flows, which is computationally expensive and often unnecessary (Wu, 2007). For some general variable ϕ, the motion can be split into $\overline{\phi}^t + \phi'$, where $\overline{\phi}^t$ is the mean component in time and ϕ' the fluctuating component. We can time-average the model equations so as to filter out these turbulent fluctuations and find (2.1) becomes

$$\nabla \cdot \overline{\mathbf{u}}^t = 0, \tag{2.3}$$

and (2.2) becomes

$$\frac{\partial \overline{\mathbf{u}}^t}{\partial t} + \nabla \cdot (\overline{\mathbf{u}}^t \otimes \overline{\mathbf{u}}^t) = \frac{1}{\rho_w}\overline{\mathbf{F}}^t - \frac{1}{\rho_w}\nabla \overline{p}^t + \frac{1}{\rho_w}\nabla \cdot \overline{\tau}^t - \nabla \cdot (\overline{\mathbf{u}' \otimes \mathbf{u}'}^t). \tag{2.4}$$

Note that the only term containing fluctuations that remains is $\overline{\mathbf{u}' \otimes \mathbf{u}'}^t$, which represents the momentum due to the turbulence in the system. The presence of this term means that (2.3) and (2.4) are not a closed system and hence following standard practice in hydro-morphodynamic models (see Wu, 2007; Tassi and Villaret, 2014), we use the Boussinesq eddy viscosity concept to close the system. This assumes that the turbulent stresses are proportional to the mean velocity gradients meaning

$$-\overline{u_i' u_j'}^t = -\frac{1}{3}\overline{u_i' u_i'}^t \delta_{ij} + \nu_{\text{turb}}\left(\frac{\partial \overline{u_i}^t}{\partial x_j} + \frac{\partial \overline{u_j}^t}{\partial x_i}\right), \tag{2.5}$$

where ν_{turb} is the kinematic eddy viscosity and we have switched to Einstein notation for clarity. The first term is absorbed into the pressure gradient term in (2.4) and the second term is combined with the stress tensor $\frac{1}{\rho_w}\frac{\partial \tau_{ij}}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\nu_m\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)\right)$ to form $\frac{\partial}{\partial x_j}\left((\nu_m + \nu_{\text{turb}})\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)\right)$. Here ν_m is the molecular viscosity, which is much smaller than ν_{turb} and hence can be ignored so $\nu_{\text{turb}} \approx \nu_{\text{turb}} + \nu_m$. There exist a variety of turbulence models to determine the eddy viscosity, ν_{turb}, but following Tassi and Villaret (2014), we assume that ν_{turb} is constant. Finally, dropping the bars for simplicity, we can rewrite (2.3) and (2.4) as

$$\nabla \cdot \mathbf{u} = 0, \tag{2.6}$$

and

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot (\mathbf{u} \otimes \mathbf{u}) = \frac{1}{\rho_w}\mathbf{F} - \frac{1}{\rho_w}\nabla p + \nu_{\text{turb}}\nabla^2 \mathbf{u}, \tag{2.7}$$

where we have used (2.6) to simplify the stress term in (2.7).

An advantage of many coastal models, including *Thetis*, is that they have both a 3D model and a simpler more computationally efficient depth-averaged 2D model. Given that most of the test cases considered in this thesis have long simulation times, we use depth-averaged models throughout. Using a 2D model requires assuming that the only external force acting on the system is gravity and that the flow is columnar, *i.e.* $\partial u_1/\partial z = \partial u_2/\partial z = 0$, where $\mathbf{u} = (u_1, u_2, u_3)$ (for more details, see Segur (2009)).

The 2D model is then derived by depth-averaging the equations from the bed z_b to the water surface η. For example, for some variable ϕ, the depth-averaged variable is defined by

$$\overline{\phi} = \frac{1}{h}\int_{z_b}^{\eta} \phi \, dz, \tag{2.8}$$

where we follow standard convention and denote depth-averaged variables with a bar, $\bar{\cdot}$. To depth-average, we must also define conditions at the fluid boundary: at the water surface, η, we apply the kinematic boundary condition because it is a free moving boundary, while at the bottom boundary, z_b, we apply a no-slip condition, justified here because we assume throughout that the bed evolution is slower than the flow. Applying these boundary conditions and using the fact that the bed evolution is slow, the time-averaged depth-averaged conservation of volume equation simplifies to

$$\frac{\partial \eta}{\partial t} + \nabla_h \cdot (h\overline{\mathbf{u}}) = 0, \tag{2.9}$$

where the 2D divergence operator $\nabla_h = (\partial/\partial x, \partial/\partial y)$, $h = \eta - z_b$ is the total water depth and $\overline{\mathbf{u}}$ is the depth-averaged velocity.

Depth-averaging (2.7) is more complex and results in dispersion momentum transport terms due to the velocity not being vertically uniform (see Wu, 2007). The latter are stress-like terms and, following standard practice (see Wu, 2007), can be combined with the turbulent stress effects $\nu_{\text{turb}}\nabla_h^2\overline{\mathbf{u}}$ to form $\nu\nabla_h^2\overline{\mathbf{u}}$. Note that $\nu \approx \nu_{\text{turb}} + \nu_{\text{disp}}$ and from hereon, for brevity, we refer to ν as the kinematic eddy viscosity. We acknowledge that because ν is now representing turbulent and dispersive effects, it may require some calibration in test cases (see for example Section 2.4.2) and indeed many studies view this ν as more of a calibration parameter (Li and Huang, 2013). Thus, using ν, applying the boundary conditions and assuming hydrostatic pressure, the time-averaged depth-averaged conservation of momentum equation simplifies to

$$\frac{\partial \overline{\mathbf{u}}}{\partial t} + \overline{\mathbf{u}} \cdot \nabla_h \overline{\mathbf{u}} + \mathbf{g}\nabla_h \eta = \nu\nabla_h^2\overline{\mathbf{u}} - \frac{C_h}{h}|\overline{\mathbf{u}}|\overline{\mathbf{u}}, \tag{2.10}$$

where the last term is an approximation of the effect of friction on the flow. Here C_h is the quadratic drag coefficient (defined in 2.39) and $\mathbf{g}$ represents gravity. Note that equations (2.9) and (2.10) are also known as the shallow water equations.

Key aspects of our work, especially in Chapter 3, require our model equations to be differentiable. However, the norm operator in (2.10) means this equation is not differentiable. Following Funke et al. (2017), we resolve this issue by replacing the norm operator in (2.10) with the following smooth approximation

$$|\overline{\mathbf{u}}| \approx \sqrt{|\overline{\mathbf{u}}|^2 + \zeta^2}, \tag{2.11}$$

where ζ is a user-defined norm smoothing parameter which, when equal to 0, returns (2.11) to the standard norm operator. In Funke et al. (2017), ζ is set equal to the wetting-and-drying parameter (defined in Section 2.2.2). However, the issue of non-differentiability exists independently of whether the domain has dry sections or not, and so here we do not use this value. In the test case in Section 4.4, we experiment with values of ζ between 0.1 and 1 and chose a value of 0.4 because we find this value is large

enough for the model not to crash due to non-differentiability issues, but not so large that the friction term spuriously affects the velocity. Note, though, that a good choice for ζ is always partly dependent on the test case. Furthermore, we recommend only using a non-zero value for ζ when the flow is changing rapidly from positive to negative, implying that a zero-velocity may occur. Otherwise, the approximation (2.11) is just adding an unnecessary spurious error, however small it might be.

Equations (2.9) and (2.10) complete the derivation of the hydrodynamic component of the fully wet domain model.

2.2.1.1 Suspended sediment equation

For the sediment transport equations, we can take either a Lagrangian or an Eulerian approach. The Lagrangian approach models the path of discrete sediment particles, whilst the Eulerian approach makes a macroscopic assumption and models the sediment via a concentration field. Here, we follow standard convention for hydro-morphodynamic models and take the Eulerian approach (Papanicolaou et al., 2008). Thus, we represent the sediment dynamics via an advection-diffusion equation for a sediment concentration field, c. Note that in this work we only consider non-cohesive sediment.

If we assume the sediment diameter is finer than 1 mm and the sediment concentration, c, is lower than 10% of the fluid volume as is standard practice (Wu, 2007), then the only significant relative motion between the flow and the sediment is settling due to gravity. Additionally these conditions mean that the settling velocity of the sediment, w_s, can be approximated by that of a single sediment particle in clear water. Hence the equation governing the suspended sediment concentration is

$$\frac{\partial c}{\partial t} + \nabla \cdot (c\mathbf{u}) = \frac{\partial}{\partial z}(w_s c). \tag{2.12}$$

As with the hydrodynamic equations, we time average (2.12) to filter turbulent fluctuations. In direct analogy with the hydrodynamic case, this results in a turbulent sediment flux term $-\overline{c'\mathbf{u}'}^t$ and we again use the Boussinesq eddy viscosity concept to close the system. This concept assumes that the turbulent flux term is proportional to the gradient of the sediment concentration *i.e.*

$$-\overline{c'\mathbf{u}'}^t = \epsilon_s \nabla \bar{c}^t, \tag{2.13}$$

where ϵ_s is the sediment turbulent diffusivity coefficient. Therefore time-averaging (2.12) gives

$$\frac{\partial c}{\partial t} + \nabla \cdot (c\mathbf{u}) - \frac{\partial (w_s c)}{\partial z} = \nabla \cdot (\epsilon_s \nabla c), \tag{2.14}$$

where we have dropped the bars for simplicity. Note that adding time-averaged turbulence effects has turned (2.12) into an advection-diffusion equation.

As we are deriving a 2D hydro-morphodynamic model, we must also depth-average (2.14). This results in dispersion momentum terms appearing due to the velocity and sediment concentration not being vertically uniform (see Wu, 2007). In analogy to the case of the hydrodynamic equation (2.10), these dispersion terms can be combined with the diffusion term to form $\nabla \cdot (\epsilon_s \nabla c)$, where ϵ_s now represents both

diffusivity and dispersion effects. For simplicity, throughout this thesis, we refer to ϵ_s as the diffusivity coefficient, as this is the dominant process.

To depth-average we must also define the suspended sediment boundary conditions. As suspended sediment transport occurs through the fluid water column and bedload transport occurs along the bed, the model domain can be conceptually divided into bedload and suspended sediment zones with an interface at $z = z_b + \delta^*$, where the height $\delta^* = \max(k_s/2,\ 0.01)$m and k_s is the Nikuradse friction height. However, following standard practice (see Wu, 2007; Tassi and Villaret, 2014), as δ^* is small, we do not explicitly represent the bedload zone in the model domain, although δ^* is still used in sediment formulae such as (2.32). Thus, the sediment gradient boundary condition

$$-\epsilon_s \frac{\partial c}{\partial z}\bigg|_{z=z_b} = w_s c_{b*} = E_b, \tag{2.15}$$

and the concentration boundary condition

$$w_s c|_{z=z_b} = w_s c_b = D_b. \tag{2.16}$$

are both applied at $z = z_b$ rather than $z = z_b + \delta^*$. Here E_b is the near-bed sediment erosion flux, D_b the deposition flux, w_s the settling velocity of the particles defined in (2.30), c_{b*} the equilibrium near-bed sediment concentration defined in (2.32) and c_b the actual near-bed sediment concentration defined in (2.27).

Thus, applying the boundary conditions and combining the diffusion and dispersion effects, the depth-averaged sediment concentration equation simplifies to

$$\frac{\partial \overline{c}}{\partial t} + \nabla_h(\overline{c}\overline{\mathbf{u}}) = \nabla_h \cdot (\epsilon_s \nabla_h \overline{c}) + \frac{E_b - D_b}{h}, \tag{2.17}$$

where we choose to implement the non-conservative form of the sediment concentration equation for the fully wet model. Due to the coupled nature of our model, however, we cannot calculate $\overline{c}\overline{\mathbf{u}}$, but only the product of $\bar{c}$ (from the sediment transport component) and $\bar{\mathbf{u}}$ (from the hydrodynamic component). These two quantities are not equal because the product of two integrated variables is not equal to the integral of their product. Thus, following Huybrechts, Villaret and Hervouet (2010) and using flow incompressibility (2.9) to move the derivative on the advection term, we rewrite (2.17) as an advection-diffusion equation for $\bar{c}$

$$\frac{\partial \overline{c}}{\partial t} + \mathbf{u}_{\text{adv}} \cdot \nabla_h \overline{c} = \nabla_h \cdot (\epsilon_s \nabla_h \overline{c}) + \frac{E_b - D_b}{h}, \tag{2.18}$$

with advection velocity

$$\mathbf{u}_{\text{adv}} = \frac{\overline{c}\overline{\mathbf{u}}}{\overline{c}} = F_{\text{corr}} \overline{\mathbf{u}}, \tag{2.19}$$

where F_{corr} is a correction factor which varies in both space and time. To determine the expression for F_{corr}, we assume $\mathbf{u}$ has a logarithmic profile, *i.e.* near the bed $\mathbf{u}$ is zero due to bed friction, and it increases as the distance from the bed increases, due to the lessening effect of friction. Additionally, we

assume that the concentration has a Rouse profile

$$\frac{c(z)}{c(\delta^*)} = \left(\frac{(h-z)\delta^*}{(h-\delta^*)z}\right)^R \tag{2.20}$$

where z indicates the vertical co-ordinate, δ^* the height of the bedload zone defined, as before, as $\max(k_s/2,\, 0.01)$m and R is the Rouse number defined by

$$R = w_s/\kappa u_*. \tag{2.21}$$

Here κ is the Von Kármán constant equal to 0.4 (Wu, 2007), and $u_* = \sqrt{\tau_b/\rho_w}$ is the shear velocity with τ_b defined in (2.37). Thus, we obtain the following expression

$$F_{\text{corr}} = \frac{I_2 - \log\left(\frac{B}{30}\right) I_1}{I_1 \log\left(\frac{eB}{30}\right)}, \tag{2.22}$$

where

$$I_1 = \int_{B^{-1}}^1 \left(\frac{(1-a)}{a}\right)^R da, \tag{2.23a}$$

$$I_2 = \int_{B^{-1}}^1 \log a \left(\frac{(1-a)}{a}\right)^R da, \tag{2.23b}$$

with $a = z/h$, $B = k_s'/h$ (where $k_s' = 3d_{50}$ is the grain roughness height) and R is the Rouse number defined by (2.21). Following Tassi and Villaret (2014), to avoid numerical integration, the Rouse concentration profile is simplified such that (2.23) becomes

$$I_1 = \begin{cases} \frac{1}{1-R}(1 - B^{1-R}), & R \neq 1, \\ -\log(B), & R = 1, \end{cases} \tag{2.24a}$$

$$I_2 = \begin{cases} \frac{I_1 + \log(B) B^{1-R}}{R-1}, & R \neq 1, \\ -0.5(\log(B))^2, & R = 1. \end{cases} \tag{2.24b}$$

Thus, the sediment concentration equation in our fully wet model is

$$\frac{\partial \overline{c}}{\partial t} + F_{\text{corr}} \overline{\mathbf{u}} \cdot \nabla_h \overline{c} = \nabla_h \cdot (\epsilon_s \nabla_h \overline{c}) + \frac{E_b - D_b}{h}, \tag{2.25}$$

where F_{corr} depends on the model variables h and $\overline{\mathbf{u}}$.

To fully describe this equation, we must define the sediment source term, $E_b - D_b$, and we recall here from (2.15) and (2.16) that

$$E_b - D_b = w_s c_{b*} - w_s c_b. \tag{2.26}$$

Additionally, following Tassi and Villaret (2014),

$$c_b = \alpha_c \overline{c}, \tag{2.27}$$

where α_c is a coefficient greater than unity which accounts for the near-bed sediment concentration value being higher than $\bar{c}$ due to gravity. This coefficient can be approximated by the following

$$\frac{1}{\alpha_c} = \begin{cases} \left|\frac{A(1-A^r)}{r}\right|, & |R-1| > 10^{-4}, \\ |-A\log(A)|, & |R-1| \leq 10^{-4}, \end{cases} \tag{2.28}$$

where

$$r = \min(R-1, 3) \tag{2.29}$$

$A = \max\left(\frac{\delta^*}{h}, 1\right)$, R the Rouse number and δ^* the height of the bedload zone. We calculate the sediment settling velocity, w_s, in (2.26) using the following formula from Van Rijn (1984),

$$w_s = \begin{cases} \frac{gDd_{50}^2}{18\nu_m}, & d_{50} \leq 10^{-4}, \\ \frac{10\nu_m}{d_{50}}\left(\sqrt{1+0.01\frac{gDd_{50}^3}{\nu_m^2}}-1\right), & 10^{-4} \leq d_{50} \leq 10^{-3}, \\ 1.1\sqrt{gDd_{50}}, & d_{50} > 10^{-3}, \end{cases} \tag{2.30}$$

where d_{50} is the median sediment diameter, ν_m is the water's kinematic molecular viscosity (equal to $1 \times 10^{-6}\,\mathrm{m^2 s^{-1}}$ throughout this thesis) and

$$D = \frac{\rho_s}{\rho_w} - 1, \tag{2.31}$$

where ρ_s is the sediment density, and ρ_w the water density.

As discussed in Garcia and Parker (1991), there are alternative formulae for c_{b*} in (2.26). In our *Thetis* model, we use the following formula from Van Rijn (1984)

$$c_{b*} = 0.015\frac{d_{50}}{\delta^*}\frac{S_0^{3/2}}{d_*^{3/10}}, \tag{2.32}$$

where d_* is the non-dimensional diameter

$$d_* = d_{50}\left(\frac{gD}{\nu_m^2}\right)^{1/3}, \tag{2.33}$$

and S_0 is the non-dimensional excess bed shear stress

$$S_0 = \frac{\Psi\tau_b - \tau_c}{\tau_c}. \tag{2.34}$$

This formula is appropriate because it is suitable for simulating problems with fine sediment, like those considered in this thesis. In (2.34), τ_c is the critical shear stress defined by

$$\tau_c = (\rho_s - \rho_w)\, g d_{50}\, \theta_{cr}, \tag{2.35}$$

where θ_{cr} is the critical Shields parameter given as

$$\theta_{cr} = \begin{cases} 0.24 d_*^{-1}, & 1 < d_* \leq 4, \\ 0.14 d_*^{-0.64}, & 4 < d_* \leq 10, \\ 0.04 d_*^{-0.1}, & 10 < d_* \leq 20, \\ 0.013 d_*^{0.29}, & 20 < d_* \leq 150, \\ 0.055, & 150 < d_*, \end{cases} \tag{2.36}$$

τ_b is the bed shear stress acting against the velocity flow and equal in magnitude in both directions given by

$$\tau_b = \frac{1}{2} \rho_w C_h |\overline{\mathbf{u}}|^2; \tag{2.37}$$

and Ψ is the skin friction correction

$$\Psi = \frac{C_h'}{C_h}. \tag{2.38}$$

Here C_h is the quadratic drag coefficient for the total friction defined by

$$C_h = 2 \frac{\kappa^2}{\log\left(\frac{11.036h}{k_s}\right)^2}, \tag{2.39}$$

where k_s is the Nikuradse friction height. Similarly, C_h' is the quadratic drag coefficient for the skin friction calculated using the grain roughness height, k_s', instead of k_s in (2.39). Therefore, whereas the total friction C_h is related to the height of the bedload zone δ^* (as $\delta^* = k_s/2$), the skin friction is related to the sediment size (as $k_s' = 3d_{50}$). To summarise E_b therefore depends on the model variables h and $\overline{\mathbf{u}}$, and D_b depends on the model variables $\bar{c}$, h and $\overline{\mathbf{u}}$.

This completes the derivation of the non-conservative suspended sediment transport equation in a fully wet domain.

2.2.1.2 Bedload transport

As shown in Figure 2.1, sediment can also be transported via bedload transport. Unlike with suspended sediment transport, bedload transport is simulated using an empirical formula. Following Tassi and Villaret (2014), to model bedload transport, we define the bedload transport flux, $\mathbf{Q_b}$, as

$$\mathbf{Q_b} = \phi_s \sqrt{g \left(\frac{\rho_s}{\rho_w} - 1\right) d_{50}^3} \left(\cos\xi, \sin\xi\right), \tag{2.40}$$

where ξ is the flow angle and thus

$$\cos\xi = \frac{\overline{u}_1}{\sqrt{\overline{u}_1^2 + \overline{u}_2^2}} \quad \text{and} \quad \sin\xi = \frac{\overline{u}_2}{\sqrt{\overline{u}_1^2 + \overline{u}_2^2}}, \tag{2.41}$$

where $\overline{\mathbf{u}} = (\overline{u}_1, \overline{u}_2)$. We choose the Meyer-Peter-Müller formula to define the non-dimensional sediment rate ϕ_s. This formula is given by

$$\phi_s = \begin{cases} 0, & \theta' < \theta_{cr}, \\ \alpha_{\text{MPM}}(\theta' - \theta_{cr})^{3/2}, & \text{otherwise}, \end{cases} \tag{2.42}$$

where θ_{cr} is given by (2.36), α_{MPM} is a coefficient equal to 8, as suggested by Tassi and Villaret (2014) and θ' is the non-dimensional Shields parameter

$$\theta' = \frac{\Psi \tau_b}{(\rho_s - \rho_w) g d_{50}}, \tag{2.43}$$

with Ψ given by (2.38) and τ_b by (2.37).

This formula for $\mathbf{Q_b}$ in (2.40) is able to represent bedload transport in straight non-sloping channels, but does not account for the gravitational and helical flow effects which also affect bedload transport. Following Tassi and Villaret (2014), we take these effects into account by imposing a slope effect correction and a secondary current correction on $\mathbf{Q_b}$. Note that both the slope effect angle correction and the secondary current correction are new additions to our model implemented in this PhD work. Previous to these additions, our model could only simulate bedload transport accurately in straight channel test cases with no bed variation in the cross-flow direction. Therefore removing these limitations is a significant step forward and we detail these corrections below.

(i) Slope effect correction

The bed gradient influences the magnitude and direction of $\mathbf{Q_b}$. When the bed has a positive gradient in the transport direction, gravity acts against the sediment, causing the magnitude of $\mathbf{Q_b}$ to decrease and its direction to alter. If it has a negative gradient, the effect will be exactly the opposite.

To correct the magnitude due to this slope effect, we multiply (2.40) by

$$1 - \Upsilon \frac{\partial z_b}{\partial s}, \tag{2.44}$$

given in Soulsby (1997), where s is a direction tangential to the current and Υ an empirical coefficient set to 1.3 (Tassi and Villaret, 2014).

To correct the angle, ξ, we use the formula given in Tassi and Villaret (2014) based on Talmon et al. (1995). To derive this formula, we first define the coefficient

$$\mathring{T} = \frac{1}{\beta_2 \sqrt{\Theta}}, \tag{2.45}$$

where, throughout this thesis, the empirical coefficient, β_2, is equal to 1.5, following guidance in Tassi and Villaret (2014). Additionally, Θ is given by

$$\Theta = \frac{(\rho_w - \rho_s)\, g d_{50}}{\max\left(\frac{1}{2}\rho_w C_h \|\overline{\mathbf{u}}\|^2, 10^{-10}\right)}, \tag{2.46}$$

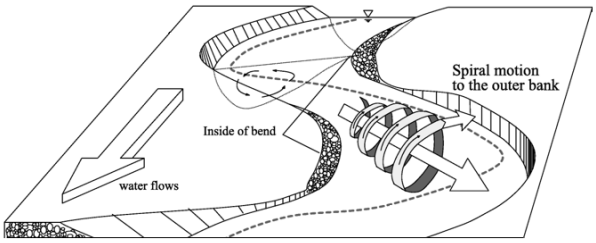


FIGURE 2.2. Secondary current in a curved channel, adapted from Park and Ahn (2019).

with C_h defined as in (2.39). Then

$$\mathbf{Q_b} = \left(\phi_s \sqrt{g\left(\frac{\rho_s}{\rho_w} - 1\right) d_{50}^3}\,(\cos\mathring{\alpha}, \sin\mathring{\alpha})\right), \tag{2.47}$$

where $\mathring{\alpha}$ is the corrected angle defined by

$$\begin{pmatrix}\sin\mathring{\alpha}\\ \cos\mathring{\alpha}\end{pmatrix} = \frac{1}{\|\mathbf{p}\|^2}\begin{pmatrix}p_1\\ p_2\end{pmatrix} = \frac{1}{\|\mathbf{p}\|^2}\begin{pmatrix}\sin\xi - \mathring{T}\left(\frac{\partial z_b}{\partial y}\right)\\ \cos\xi - \mathring{T}\left(\frac{\partial z_b}{\partial x}\right)\end{pmatrix}, \tag{2.48}$$

with $\mathbf{p} = (p_1, p_2)$.

(ii) Secondary current correction

As illustrated in Figure 2.2, there are both current and helical flow effects in curved channels, which affect the magnitude and direction of $\mathbf{Q_b}$. We account for these effects by implementing a secondary current correction in our model using

$$\tan\varsigma = 7\frac{h}{r}, \tag{2.49}$$

given in Engelund (1974), where ς is the angle between the bedload transport and the main flow direction, h the mean water depth, and r the local radius of curvature of the streamline calculated using

$$r = \frac{\alpha'|\overline{\mathbf{u}}|^2}{g\frac{\partial\eta}{\partial n}}, \tag{2.50}$$

where η is the elevation, n a direction normal to the current and α' a coefficient which lies between 0.75 (rough bed) and 1 (smooth bed).

Using (2.49), we construct the term

$$\Xi = \sqrt{\left(\tau_b\Upsilon\cos\mathring{\alpha} + \tau_b\overline{u}_2\tan\varsigma\right)^2 + \left(\tau_b\Upsilon\sin\mathring{\alpha} - \tau_b\overline{u}_1\tan\varsigma\right)^2}, \tag{2.51}$$

where τ_b is the bedload shear stress defined by (2.37), $\overline{\mathbf{u}} = (\overline{u}_1, \overline{u}_2)$ and $\mathring{\alpha}$ and Υ are the corrected flow angle and magnitude factors from the slope effect correction.

Hence, we define a new corrected bed transport flow direction $\hat{\alpha}$ with

$$\cos\hat{\alpha} = \frac{\tau_b \Upsilon \cos\mathring{\alpha} + \tau_b \overline{u}_2 \tan\varsigma}{\Xi}, \tag{2.52a}$$

$$\sin\hat{\alpha} = \frac{\tau_b \Upsilon \sin\mathring{\alpha} - \tau_b \overline{u}_1 \tan\varsigma}{\Xi}, \tag{2.52b}$$

and a new slope magnitude correction factor

$$\mathring{\Upsilon} = \frac{\Xi}{\tau_b}. \tag{2.53}$$

2.2.1.3 Exner equation

The erosion and deposition of sediment by both suspended sediment and bedload transport processes clearly impacts the underlying bed morphology (see Figure 2.1). This bedlevel change is governed by the Exner equation

$$\frac{(1-p')}{m_f}\frac{dz_b}{dt} + \nabla_h \cdot \mathbf{Q_b} = D_b - E_b, \tag{2.54}$$

where p' is the bed sediment porosity, z_b is the underlying bed, $\mathbf{Q_b}$ is the bedload transport defined by (2.40) (with the appropriate corrections), E_b and D_b are the erosion and deposition fluxes, respectively, defined by (2.26) and m_f is a morphological acceleration factor. The latter is a scalar quantity which does not appear in the real physics, but is included so that the physics can be manipulated and the rate of bedlevel changes artificially increased compared with the underlying hydrodynamics. This is justified because morphological changes are assumed to occur on longer time scales than hydrodynamic changes (Lesser et al., 2004). This factor is frequently used in the simulation of fluvial and coastal hydro-morphodynamics (Morgan et al., 2020; Dastgheib et al., 2008; Van der Wegen and Roelvink, 2008; Nicholas et al., 2013; Singh et al., 2017) because it saves significant computational time when simulating long-term morphodynamic change (Gerritsen et al., 2008): for some computational time t, with a morphological acceleration factor, the actual simulation time is $t^* = m_f t$. There are obvious limits to the magnitude of m_f but no concrete theory for determining the correct value. However explicit studies have concluded that cases with low Froude numbers ($Fr = u/\sqrt{gl}$) allow for higher morphological acceleration factors (Ranasinghe et al., 2011).

This equation concludes the derivation of our 2D depth-averaged coupled hydrodynamics and sediment transport model for a fully wet domain.

2.2.1.4 Summary of hydro-morphodynamic equations in a fully wet domain

In summary, the full equation set in the case of a fully wet domain is the following.

Hydrodynamic equations:

$$\frac{\partial \eta}{\partial t} + \nabla_h \cdot (h\overline{\mathbf{u}}) = 0; \tag{2.55}$$

$$\frac{\partial \overline{\mathbf{u}}}{\partial t} + \overline{\mathbf{u}} \cdot \nabla_h \overline{\mathbf{u}} + \mathbf{g}\nabla_h \eta = \nu \nabla_h^2 \overline{\mathbf{u}} - \frac{C_h}{h}|\overline{\mathbf{u}}|\overline{\mathbf{u}}; \tag{2.56}$$

Non-conservative sediment concentration equation:

$$\frac{\partial \overline{c}}{\partial t} + F_{\text{corr}}\overline{\mathbf{u}} \cdot \nabla_h \overline{c} = \nabla_h \cdot (\epsilon_s \nabla_h \overline{c}) + \frac{E_b - D_b}{h}; \tag{2.57}$$

Exner equation:

$$\frac{(1-p')}{m_f}\frac{dz_b}{dt} + \nabla_h \cdot \mathbf{Q_b} = D_b - E_b. \tag{2.58}$$

The properties of the variables and parameters in Eq. (2.55–2.58) are summarised in Table 2.1.

Name	**Type**	**Dependencies on model variables**
$\overline{\mathbf{u}}(\mathbf{x}, t)$	Model variable	-
$h(\mathbf{x}, t)$	Model variable	-
$\eta(\mathbf{x}, t)$	Model variable	-
$\overline{c}$	Model variable	-
z_b	Model variable	-
C_h	Empirical formula	h
F_{corr}	Empirical formula	h, $\overline{\mathbf{u}}$
E_b	Empirical formula	h, $\overline{\mathbf{u}}$
D_b	Empirical formula	$\overline{c}$, h, $\overline{\mathbf{u}}$
$\mathbf{Q}_b$	Empirical formula	h, $\overline{\mathbf{u}}$
ν	Model parameter	-
ϵ_s	Model parameter	-
$\mathbf{g}$	Fixed physical parameter	-
p'	Physical parameter	-
m_f	User-defined parameter	-

TABLE 2.1. Properties of the variables and parameters in Eq. (2.55–2.58), including their dependency on model variables.

2.2.2 Model equations for wet-dry domain

The model described in Section 2.2.1 and summarised in Section 2.2.1.4 is capable of simulating sediment transport and bedlevel change in a fully wet domain. However, in coastal problems, the domain often has a wet-dry interface which must be simulated and, therefore, we also derive a wet-dry hydro-morphodynamic model. We emphasise here that the derivation and implementation of this wet-dry hydro-morphodynamic model is completely new work conducted in this PhD and was not submitted in the author's MRes.

As we are building our hydro-morphodynamic model in *Thetis*, we use its inbuilt wetting-and-drying scheme, which follows the approach detailed in Kärnä et al. (2011). The total water depth, h, is modified using the expression

$$\tilde{H} := \eta - z_b + f(h), \tag{2.59}$$

where η is the elevation, z_b is the bedlevel and

$$f(h) := \frac{1}{2}\left(\sqrt{h^2 + \delta^2} - h\right), \tag{2.60}$$

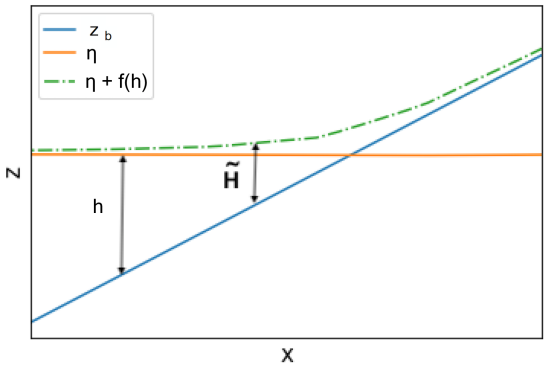


FIGURE 2.3. Wetting and drying scheme diagram showing the relationship between the total water depth h and its modified form $\tilde{H}$.

as shown in Figure 2.3.

Here, δ is the wetting-and-drying parameter and Kärnä et al. (2011) recommend setting it to approximately equal to $d|\nabla h|$, where d is the typical length scale of the mesh. Thus, the hydrodynamic equations in our wet-dry model are

$$\frac{\partial \tilde{H}}{\partial t} + \nabla_h \left(\tilde{H}\overline{\mathbf{u}}\right) = 0; \tag{2.61}$$

$$\frac{\partial \overline{\mathbf{u}}}{\partial t} + \overline{\mathbf{u}} \cdot \nabla_h \overline{\mathbf{u}} + \mathbf{g}\nabla\eta = \nu\nabla_h^2\overline{\mathbf{u}} - \frac{C_{\tilde{H}}}{\tilde{H}}|\overline{\mathbf{u}}|\overline{\mathbf{u}}, \tag{2.62}$$

where $\overline{\mathbf{u}}$ is the depth-averaged velocity, $\mathbf{g}$ gravity, ν the kinematic eddy viscosity and $C_{\tilde{H}}$ the quadratic drag coefficient simulating bed friction as defined in (2.39), but dependent on $\tilde{H}$ instead of h. Like in the case of the fully wet model in Section 2.2.1, we also have the capability here to smooth $|\overline{\mathbf{u}}|$ using (2.11) to avoid non-differentiable functions.

2.2.2.1 Suspended sediment equation

The wetting-and-drying scheme used in *Thetis* is known to interfere with scalar mass conservation (Kärnä et al., 2011). Therefore, we cannot use the non-conservative sediment equation (2.57) derived in Section 2.2.1.1 but must, instead, consider the following conservative form to ensure sediment is conserved with the wetting-and-drying scheme

$$\frac{\partial}{\partial t}(\tilde{H}\bar{c}) + \nabla_h \cdot (F_{\text{corr}}\tilde{H}\overline{\mathbf{u}}\,\bar{c}) = \nabla_h \cdot (\epsilon_s \nabla_h(\tilde{H}\bar{c})) + E_b - D_b, \tag{2.63}$$

where $\tilde{H}$ is the modified water depth given by (2.59) and F_{corr} is now dependent on $\tilde{H}$ rather than h. Instead of solving for $\bar{c}$, we solve for the depth-integrated concentration, $\tilde{H}\bar{c}$, which allows us to use the same finite element formulation in the conservative and non-conservative sediment equations (see Section 2.2.3).

To verify that our conservative scheme has improved the sediment conservation with the wetting-and-drying scheme, we consider the simple Thacker test case, in which oscillations occur in a paraboloid bowl with diameter 430 620 m (Thacker, 1981). The hydrodynamic version of this test case is presented in

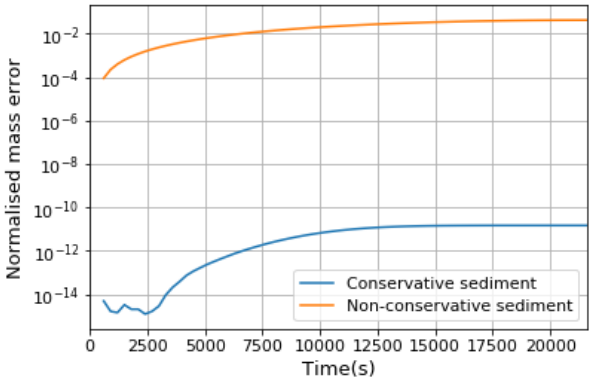


FIGURE 2.4. Comparing sediment mass conservation error when using wetting-and-drying with the non-conservative sediment equation (2.57) and with the conservative sediment equation (2.63).

Balzano (1998) and we refer the reader there for more details. The free surface is initially a paraboloid of revolution with a depth of approximately 50 m which oscillates inside the bowl with no forcing, but does not leave the bowl, *i.e.* the problem is closed hydrodynamically. We introduce a Gaussian blob of sediment in the wet part of the domain defined by the expression

$$\overline{c}(t_0) = \begin{cases} 100\exp^{-(x^2+y^2)/10^5} & \tilde{H} \geq 0 \\ 0 & \tilde{H} < 0 \end{cases} \tag{2.64}$$

and run the simulation for 6 h. The principal differences between the conservative and non-conservative schemes are due to the advection term and, therefore, we set the erosion and deposition terms to 0 to avoid unnecessary complexities. At each timestep, t_n, we calculate the normalised mass error using

$$\frac{\int \overline{c}(t_n)\tilde{H}(t_n)\,dx - \int \overline{c}(t_0)\tilde{H}(t_0)\,dx}{\int \overline{c}(t_0)\tilde{H}(t_0)\,dx}. \tag{2.65}$$

Figure 2.4 shows that the normalised mass error has order of magnitude 10^{-2} for the non-conservative sediment equation (2.57), but order of magnitude 10^{-11} for the conservative sediment equation (2.63), which is closer to computer precision. Thus, using the conservative sediment equation with the wetting-and-drying scheme conserves sediment at a much better rate than using the non-conservative one. Note that if the conservative sediment equation were used in the fully wet model, sediment would also be conserved better, but the difference between the two normalised mass errors would be much smaller.

2.2.2.2 Exner equation

To complete our wetting-and-drying hydro-morphodynamic model, we use the Exner equation to model bedlevel changes. Because the Exner equation is not explicitly dependent on depth, its general form is the same as that in the fully wet model and is given by (2.54). However, wet-dry test cases like beaches often have steep slopes and applying hydro-morphodynamic models to them can generate physically unrealistic slopes. Hence, we add a 'sediment slide' mechanism (sometimes called an avalanche mechanism) to the Exner equation in our model. A number of approaches exist in the literature and we choose to use that described in Apsley and Stansby (2008). This adds an avalanche term q_{aval} in the direction of the

maximum slope, $\mathbf{b}$, to the Exner equation (2.58) which becomes

$$\frac{(1-p')}{m_f}\frac{dz_b}{dt} + \nabla_h \cdot (\mathbf{Q_b} + q_{\text{aval}}\,\mathbf{b}) = D_b - E_b. \tag{2.66}$$

Note that, although $\mathbf{Q_b}$ and q_{aval} are both in the same divergence term, it is possible to use a sediment slide mechanism without simulating bedload transport and vice versa (see for example Section 2.5.1).

As we are solving a 2D depth-averaged problem, we ignore the z-component of $\mathbf{b}$ and thus

$$\mathbf{b} = -\frac{n_z^2}{\sin\lambda}\nabla_h z_b \tag{2.67}$$

where λ is the slope angle and n_z is the z-component of the unit normal to the surface defined by

$$n_z = \frac{1}{\sqrt{1+|\nabla_h z_b|^2}}. \tag{2.68}$$

Finally, the avalanche term, q_{aval}, in (2.66) is defined by

$$q_{\text{aval}} = \begin{cases} 0.5(1-p')\mathring{\zeta}^2\frac{(\tan\lambda - \tan\phi)}{\cos(\lambda\Delta t)}, & \tan\lambda > \tan\phi, \\ 0, & \text{otherwise}, \end{cases} \tag{2.69}$$

where $\mathring{\zeta}$ is the length scale which controls how quickly the sediment is redistributed and ϕ is the angle of repose, *i.e.* the steepest angle at which the slope is physically realistic. In Apsley and Stansby (2008), it is argued that the value of $\mathring{\zeta}$ does not need to be very precise, and thus we set it equal to the approximate mesh element size. This mechanism then restricts the slope angle exceeding ϕ, although it does not model how the sediment slides down the slope.

This completes the derivation of our 2D depth-averaged coupled hydro-morphodynamic model for a wet-dry domain to be used in coastal zone test cases.

2.2.2.3 Summary of hydro-morphodynamic equations in a wet-dry domain

In summary, the full equation set in the case of a wet-dry domain is the following.

Hydrodynamic equations:

$$\frac{\partial \tilde{H}}{\partial t} + \nabla_h\left(\tilde{H}\overline{\mathbf{u}}\right) = 0; \tag{2.70}$$

$$\frac{\partial \overline{\mathbf{u}}}{\partial t} + \overline{\mathbf{u}}\cdot\nabla_h\overline{\mathbf{u}} + \mathbf{g}\nabla\eta = \nu\nabla_h^2\overline{\mathbf{u}} - \frac{C_{\tilde{H}}}{\tilde{H}}|\overline{\mathbf{u}}|\overline{\mathbf{u}}; \tag{2.71}$$

Conservative sediment concentration equation:

$$\frac{\partial}{\partial t}(\tilde{H}\overline{c}) + \nabla_h\cdot(F_{\text{corr}}\tilde{H}\overline{\mathbf{u}}\,\overline{c}) = \nabla_h\cdot(\epsilon_s\nabla_h(\tilde{H}\overline{c})) + E_b - D_b; \tag{2.72}$$

Exner equation [with optional sediment slide mechanism]:

$$\frac{(1-p')}{m_f}\frac{dz_b}{dt} + \nabla_h\cdot(\mathbf{Q_b} + [q_{\text{aval}}\,\mathbf{b}]) = D_b - E_b. \tag{2.73}$$

The properties of the variables and parameters in Eq. (2.70–2.73) are summarised in Table 2.2.

Name	Type	Dependencies on model variables
$\overline{\mathbf{u}}(\mathbf{x},t)$	Model variable	-
$\tilde{H}(\mathbf{x},t)$	Combined model variable	$\eta,\ h,\ z_b$
η	Model variable	-
$\overline{c}$	Model variable	-
z_b	Model variable	-
$C_{\tilde{H}}$	Empirical formula	$\tilde{H}$
F_{corr}	Empirical formula	$\tilde{H},\ \overline{\mathbf{u}}$
E_b	Empirical formula	$\tilde{H},\ \overline{\mathbf{u}}$
D_b	Empirical formula	$\overline{c},\ \tilde{H},\ \overline{\mathbf{u}}$
$\mathbf{Q}_b$	Empirical formula	$\tilde{H},\ \overline{\mathbf{u}}$
$q_{\text{aval}}\mathbf{b}$	Empirical formula	z_b
ν	Model parameter	-
ϵ_s	Model parameter	-
$\mathbf{g}$	Fixed physical parameter	-
p'	Physical parameter	-
m_f	User-defined parameter	-

TABLE 2.2. Properties of the variables and parameters in Eq. (2.70–2.73), including their dependency on model variables.

2.2.3 Finite Element Method implementation in *Thetis*

A variety of different numerical methods exist to solve the equations appearing in hydro-morphodynamic models, many of which are discussed in Section 2.3. We choose to solve our hydro-morphodynamic model by implementing it in *Thetis*, a finite element coastal ocean modelling system (Kärnä et al., 2018) built using the *Firedrake* code-generation framework (Rathgeber et al., 2017). Using a *Firedrake* based model has several advantages. One of these is that the *Firedrake* finite element framework is versatile and easy to use because the user interface uses Unified Form Language (UFL) (Alnæs et al., 2014). This means the finite element forms in the code can be written at a very high level that is almost equivalent to their form in the mathematical equations. At the same time, *Firedrake* automatically generates the underlying code to solve these finite element equations and, thus, the code is robust and optimised. Moreover, as discussed in Chapter 1, implementing our model in *Firedrake* means we can easily apply the adjoint method to our model in Chapter 3 and incorporate mesh movement methods with it in Chapter 4.

As a starting point for our model, we use and build upon existing components in the inbuilt 2D depth-averaged hydrodynamic model in *Thetis* which is outlined in Vouriot et al. (2019). This 2D model includes the shallow water equations and a non-conservative depth-averaged tracer equation which we use as the basis for our depth-averaged sediment concentration equation. A new addition as part of this PhD work is that the hydrodynamic equations, sediment concentration equation and Exner equation are fully coupled together in our *Thetis* implementation, thus increasing the stability and accuracy of the model. Note that the complexity and nonlinearity of this combined system means the equations are solved alternately, as opposed to as a monolithic system and, thus, our model is not fully implicit. This does not adversely affect our results, because the timescale of the morphodynamic changes is much longer than the typical

model timestep. To solve the equations themselves, we use a discontinuous Galerkin based finite element discretisation (DG) which has several advantages in this context, as discussed in Section 2.1.

The time derivatives of our model equations are all approximated using a fully-implicit backward Euler timestepping scheme. Various studies have shown that the long nature of hydro-morphodynamic simulations means that implicit timestepping schemes are more practical than explicit ones because they allow larger timesteps to be chosen and aid stability (Clark et al., 2010; Bilanceri et al., 2011). Furthermore, the sediment concentration (2.57) and Exner (2.58) equations in the hydro-morphodynamic model are governed by a range of different time-scales, which makes the system stiff, and therefore better suited for implicit timestepping. For example in (2.57), the advection term has the timescale $[x]/[u]$ (where [] are used to represent dimensions), the diffusion term has the timescale $[x]^2/\epsilon_s$, the erosion term has the complicated timescale $\left([g]^{8/5}[h][d_{50}]^{9/5}\right) / \left([c][w_s][u]^3[\nu]^{1/5}\right)$ and the deposition term has the timescale $[h]/[w_s]$. These timescales have all been derived using dimensional analysis. An important observation from this example is that if $[u]$ increases, the timescale for erosion decreases, but the timescale for deposition and diffusion remain the same. This leads to an increase in sediment in the system which can only be advected and the increase in advection is less than the increase in erosion. This can quickly lead to numerical instabilities due to large spatial gradients and further justifies using an implicit timestepping mechanism to 'smear' these out and aid stability. Finally, we choose backward Euler because of its simplicity and robustness.

A potential disadvantage of enforcing stability is that it can introduce artificial diffusion into a model leading to inaccuracies. However, for our model this artificial diffusion is negligible compared to the diffusion already present in the system and the inaccuracies caused by instabilities are far greater. This can be seen in the two test cases studied in Section 2.4, where we compare our model results using an implicit timestepping scheme to Sisyphe model results with an explicit timestepping scheme (see Section 2.3.1). These results show that whilst our model is unaffected by timestep changes, *Sisyphe* results have large inaccuracies unless the timestep is kept small. More importantly these test cases show that for small timesteps, *Sisyphe* agrees with our model indicating that the artificial diffusion has not led to significant inaccuracies.

2.2.3.1 DG based methods in *Thetis*

To apply DG based methods, we first generate an unstructured mesh of triangular elements tesselating our full domain Ω (including the dry regions), and defining our finite element space on this mesh. Note that throughout this work we use P1-DG elements (*i.e.* piecewise linear discontinuous finite elements) and the function space is $W = \{w \in L^2 \; s.t. \; w|_K \in P_1(K) \;\; \forall K \in \mathcal{T}\}$, where $P_1(K)$ is the space of linear functions on a triangle K, and $\mathcal{T}$ is the triangulation comprising the mesh. Using a discontinuous function space requires the definition of variables on element edges (including on the domain boundary $\partial\Omega$), with the union of these edges denoted by Γ. The average operator $\{\{\cdot\}\}$ and jump operator $[[\cdot]]$ across the

interior edges on scalar and vector fields are

$$\{\{\mathbf{X}\}\} = \frac{1}{2}(\mathbf{X}^+ + \mathbf{X}^-), \qquad [[\chi]]_\mathbf{n} = \chi^+\mathbf{n}^+ + \chi^-\mathbf{n}^-,$$
$$[[\mathbf{X}]]_\mathbf{n} = \mathbf{X}^+ \cdot \mathbf{n}^+ + \mathbf{X}^- \cdot \mathbf{n}^-,$$

where $\mathbf{n} = (n_x, n_y, 0)$ is the horizontal projection of the outward pointing unit normal on the element edge and '+' and '−' denote either side of the interior edge.

Hydrodynamic equations

Thetis uses very similar techniques to solve the hydrodynamic equations and the sediment concentration equation. We focus on the latter because it is the most pertinent for this thesis and refer the reader to Kärnä et al. (2018), Pan, Kramer and Piggott (2019) and Vouriot et al. (2019) for the DG formulation for the hydrodynamic equations.

Sediment concentration equation

To define the sediment concentration on the element edges, *Thetis* uses an upwinding scheme, $\overline{c}$, which means that at each edge, $\overline{c}$ is chosen to be equal to its upstream value with respect to velocity, $\overline{c}^{\text{up}}$ (see Leveque, 1996). Recall from Section 2.2.2.1 that for the conservative sediment equation (2.63) we solve for the depth-integrated concentration, $\tilde{H}\overline{c}$, instead of $\overline{c}$. This allows us to use the same finite element formulation for both the non-conservative and conservative sediment equations. Therefore, we only derive the numerical implementation of the non-conservative equation, but the conservative implementation can be found by replacing all $\overline{c}$ with $\tilde{H}\overline{c}$ and multiplying the source term by $\tilde{H}$.

The starting point for our numerical implementation is to multiply (2.57) by ψ, the test function employed in the weak formulation of the finite element method, and integrate over the domain. Recalling that we are using backward Euler for the time discretisation, this yields the following weak form

$$\int_\Omega \psi \left(\frac{\overline{c_i}^{(n+1)} - \overline{c_i}^{(n)}}{\Delta t} \right) dx + \int_\Omega \psi F_{\text{corr}} \overline{\mathbf{u_i}}^{(\mathbf{n+1})} \cdot \nabla_h \overline{c_i}^{(n+1)} dx - \int_\Omega \psi \nabla_h \cdot \left(\epsilon_s \nabla_h \overline{c_i}^{(n+1)} \right) dx = \int_\Omega \psi \frac{\left(E^n_{b_i} - D^n_{b_i}\right)}{h} dx. \tag{2.74}$$

Several choices can be made to reach the final form used in the model. We choose to integrate the advection term by parts to obtain a boundary integral term, which allows the imposition of boundary conditions on our equation and the control of fluxes between elements on the element boundaries. Thus, the weak form of the advection term becomes

$$\int_\Omega \psi F_{\text{corr}} \overline{\mathbf{u}} \cdot \nabla_h \overline{c} dx = -\int_\Omega \overline{c} \nabla_h \cdot (F_{\text{corr}} \overline{\mathbf{u}} \psi) dx + \int_\Gamma \overline{c}^{\text{up}} \, [[\psi F_{\text{corr}} \overline{\mathbf{u}}]]_\mathbf{n} ds, \tag{2.75}$$

recalling that Γ denotes the union of all the element edges.

For the diffusivity term, we must transform the second-order derivative to a first-order one since the equations are solved using a piecewise linear function space. Following Kärnä et al. (2018), we integrate by parts, applying the Symmetric Interior Penalty Galerkin (SIPG) method given in Epshteyn and Rivière

(2007) to ensure the discretisation is stable for elliptic operators. The penalty parameter σ is given as

$$\sigma = \gamma_m \frac{p(p+1)}{L}, \tag{2.76}$$

where p is the degree of the basis function, γ_m is a mesh quality factor and L is the local element length scale pointing in the interface's normal direction. We follow the standard *Thetis* procedure (Kärnä et al., 2018) and set $\gamma_m = 5$. The weak form of the diffusivity term is then

$$\begin{aligned} -\int_\Omega \psi \nabla_h \cdot (\epsilon_s \nabla_h \bar{c}) dx = \int_\Omega \epsilon_s (\nabla_h \psi) \cdot (\nabla_h \bar{c}) dx - \int_\Gamma [[\psi]]_{\mathbf{n}} \cdot \{\{\epsilon_s \nabla_h \bar{c}\}\} ds - \int_\Gamma [[\bar{c}]]_{\mathbf{n}} \cdot \{\{\epsilon_s \nabla_h \psi\}\} ds \\ + \int_\Gamma \sigma \{\{\epsilon_s\}\} [[\bar{c}]]_{\mathbf{n}} \cdot [[\psi]]_{\mathbf{n}} ds. \end{aligned} \tag{2.77}$$

To solve the full sediment concentration weak form equation, *Thetis* formulates the equation as a matrix problem for $\bar{c}^{(n+1)}$ and uses the generalised minimal residual method (GMRES) to solve the system with additional outer Newton iterations due to the non-linearity (see Jacobs and Piggott, 2015). To aid with the solution of the matrix problem, *Thetis* also employs an SOR (Successive Over-Relaxation) pre-conditioner. The use of upwinded numerical fluxes and slope limiters means our model is robust at modelling any steep bed gradients formed (see Kubatko et al., 2006). This is evidenced by the way our model is able to cope with the steep slope gradients in the migrating trench test case in Section 2.4.1 later in this chapter.

Exner Equation

In order to avoid grid-scale noise and unstable oscillations in solving the Exner equations (2.58) and (2.73), we define the bedlevel, z_b, on a continuous function space, and thus use a continuous Galerkin based finite element discretisation (CG). We project all hydrodynamic and sediment transport variables from the DG space into the CG space before calculating the terms in the Exner equation. This causes a minor loss of accuracy in model variables, but a more stable bedlevel result overall. The weak form of the divergence term $\nabla_h \cdot \mathbf{Q_b}$ is obtained by multiplying it by a test function ψ and integrating by parts once to find

$$\int_\Omega \psi \nabla_h \cdot \mathbf{Q_b} \, dx = -\int_{\partial\Omega} (\mathbf{Q_b} \cdot \mathbf{n}) \psi \, ds + \int_\Omega (\mathbf{Q_b} \cdot \nabla_h) \psi \, dx. \tag{2.78}$$

Here, the only boundary contribution is from the domain boundary $\partial\Omega$ because we are on a continuous function space and are assuming centred fluxes on interior edges. Therefore, the values on either side of each interior edge cancel over the whole domain. Note that the radius of curvature (2.50) in the secondary current parametrisation is dependent on the surface elevation η rather than on z_b. Hence, we rewrite η as $(h + z_b)$ in (2.50) so that we can benefit from an implicit discretisation.

Recalling that we are using backward Euler for the time discretisation, the weak form of the Exner equation is

$$\int_\Omega \left(\frac{(1-p')}{m_f} \frac{z_{b_i}^{(n+1)} - z_{b_i}^{(n)}}{\Delta t} \right) \psi \, dx = G_i^{(n+1)}, \tag{2.79}$$

where $G_i^{(n+1)}$ is equal to the sum of the weak form of (2.78) and the weak form of the source term (equivalent to the weak form of the source term in (2.74) multiplied by h).

If the sediment slide mechanism is used, then $G_i^{(n+1)}$ also includes the weak form of the sediment slide term, $\nabla_h \cdot (q_{\text{aval}}\mathbf{b})$, in (2.66). Note that using (2.67),

$$\nabla_h \cdot (q_{\text{aval}}\mathbf{b}) = -\nabla_h \cdot \left(\frac{n_z^2 q_{\text{aval}}}{\sin\lambda} \nabla_h z_b \right) = -\nabla_h \cdot (\gamma_s \nabla_h z_b), \tag{2.80}$$

which represents a diffusion term with diffusivity γ_s. Multiplying (2.80) by a test function ψ and integrating by parts once, the weak form is then

$$-\int_\Omega \psi \nabla_h \cdot (\gamma_s \nabla_h z_b)\, dx = \int_\Omega \gamma_s (\nabla_h \psi) \cdot (\nabla_h z_b)\, dx. \tag{2.81}$$

Here there is no boundary integral because we assume Neumann conditions of no flux in the bed at the domain boundary and, as with (2.78), the continuous Galerkin finite element discretisation means the centered flux values on either side of each interior edge cancel each other out over the whole domain.

This concludes the set-up and solution of the our hydro-morphodynamic model in *Thetis*.

2.3 Other hydro-morphodynamic models used in this work

The main focus of Chapters 3 and 4 is our depth-averaged hydro-morphodynamic model implemented in *Thetis*, as described in the previous sections. In order to validate this model and demonstrate its capability, we compare its results against experimental data and other industry-standard model results. In this section, we briefly describe the industry-standard models Telemac-Mascaret, also referred to as *Sisyphe*, and XBeach that we use for validation purposes. Additionally, as discussed in Section 2.1, our model cannot accurately represent short wavelength wind-driven waves and, thus, in Chapters 5 and 6, we use XBeach and SFINCS to simulate test cases with complex wave spectra. Thus, in this section, we also briefly describe SFINCS. We note that this is a new review of these hydro-morphodynamic models, which was not part of the author's MRes work.

2.3.1 Telemac-Mascaret – *Sisyphe*

In this work, we use the industry-standard modelling framework Telemac-Mascaret to validate our fully wet hydro-morphodynamic model in *Thetis*. Telemac-Mascaret is an integrated suite of computational fluid dynamics models designed to solve free-surface flow problems and is developed by a consortium of organisations. It includes a 2D depth-averaged hydrodynamics finite element module, Telemac2D, and a sediment transport and bed evolution finite element module, Sisyphe, which can be coupled together to form a hydro-morphodynamic model, hereafter referred to as *Sisyphe* for simplicity. *Sisyphe* uses very similar governing equations to those in our *Thetis* model (Eqs. 2.55–2.58) to simulate hydro-morphodynamic flows in a fully wet domain and includes the slope and secondary current effect corrections discussed above.

The main difference between the two models is that whilst our *Thetis* model solves the hydro-morphodynamic equations using discontinuous Galerkin based methods, *Sisyphe* uses a continuous Galerkin finite element model (Danilov, 2013). In particular, in this thesis, we compare our results against those presented in Villaret et al. (2013) and Villaret et al. (2016), both of which couple the finite element model with the method of characteristics for the hydrodynamic advection terms and a distributive scheme for the sediment transport advection terms. Full details on these methods can be found in Hervouet (2007) and here we give a brief overview:

- The **method of characteristics** solves the hydrodynamic advection terms using a semi-lagrangian approach, *i.e.* following the fluid flow (Hervouet, 2007). For each node A in the mesh, the method integrates for Δt seconds backwards in time along the characteristic that passes through this node A. The point it reaches is the foot of the characteristic, and the method then sets the solution value at node A at the $(n+1)^{th}$ timestep to be equal to the interpolated solution value at the foot of the characteristic at the n^{th} timestep. This method has the advantage of being unconditionally stable. However, as stated in Tassi and Villaret (2014), it is not mass conservative and is diffusive for small timesteps, meaning the problem is artificially regularised with potentially spurious mixing. This sensitivity to diffusivity is discussed further in Section 2.4.1 and Chapter 3. Moreover, the foot of the characteristic is unlikely to be a node in the mesh and, thus, interpolation is necessary to obtain the value causing additional inaccuracies;
- The **distributive schemes** (also known as distribution schemes) operate by integrating the sediment transport advection terms over a mesh element and then distributing fractions of this flux balance to the nodes of that mesh element (Hervouet, 2007). These fractions are positive or negative depending on whether material enters the element at this node or exits it. The solution value at each node is then updated by assembling the contributions from all the elements associated with each node. These schemes are mass conservative, but also have high numerical diffusion and Courant number limitations to ensure stability (see Hervouet et al., 2017).

The issues with the two methods just discussed support our use of DG finite element methods in *Thetis*. Moreover, we show in Section 2.4 that the benefit of DG finite element methods leads to our *Thetis* model being more robust and accurate than *Sisyphe* for the test cases considered.

Note that in this thesis, we use version v7p2 of the Telemac-Mascaret release and, unless explicitly stated, parameter values are left at the default for this version.

2.3.2 XBeach

XBeach is a depth-averaged finite-volume coastal ocean model developed by Deltares which simulates near-shore hydrodynamics and morphodynamics (Roelvink et al., 2015). In this work, we use XBeach to help validate our hydro-morphodynamic model in a wet-dry domain. Additionally, in Chapters 5 and 6,

we also use XBeach to simulate more complex coastal test cases because it can parameterise unresolved wave propagation, such as wind-driven wave fields, which neither our model nor the standard Telemac-Mascaret model can. XBeach has been successfully used numerous times to simulate wave propagation, flow, sediment transport and morphodynamic changes in the coastal zone (see for example McCall et al., 2010; Riesenkamp, 2011; Callaghan et al., 2013; Roelvink et al., 2018; de Beer et al., 2020). Within XBeach, surface elevation and flow are modelled using the Generalised Lagrangian Mean formulation of the depth-averaged hydrodynamic equations (Andrews and McIntyre, 1978), enabling the modelling of short wave-induced mass fluxes and return flows. These equations are very similar to the hydrodynamic equations (2.55–2.56) used in our *Thetis* model and are given by

$$\frac{\partial \eta}{\partial t} + \nabla_h \cdot (h\mathbf{u}^L) = 0, \tag{2.82}$$

$$\frac{\partial \overline{\mathbf{u}}^L}{\partial t} + \mathbf{u}^L \cdot \nabla_h \overline{\mathbf{u}}^L + g\nabla_h \eta = \nu \nabla_h^2 \overline{\mathbf{u}}^L + \frac{\tau_s}{\rho_w h} - \frac{\tau_b^E}{\rho_w h} + \frac{\mathbf{F}}{\rho_w h}, \tag{2.83}$$

where $\mathbf{u}^L$ is the Lagrangian velocity, which is defined in Roelvink et al. (2015) as equal to the Eulerian velocity (used in *Thetis* and *Sisyphe*) plus the Stokes drift, and can be interpreted as the distance travelled by an individual water particle during one wave period divided by that wave period. Furthermore, in (2.82) and (2.83), ν the kinematic eddy viscosity, η the elevation, $\mathbf{F}$ the wave-induced stresses, ρ_w the water density, h the water depth and τ_s and τ_b^E the wind and bed shear stresses, respectively. Note that the bed shear stress must be dependent on the velocity experienced by the bed and thus, the Eulerian velocity (denoted by E) must be used to calculate it rather than the Lagrangian one.

Waves can be modelled using several different options in XBeach, including the stationary wave model and the surfbeat model (Roelvink et al., 2018). These methods solve (2.82) and (2.83), together with the following wave action equation which calculates the propagation of the short wave energy

$$\frac{\partial A}{\partial t} + \frac{\partial c_x A}{\partial x} + \frac{\partial c_y A}{\partial y} + \frac{\partial c_\theta A}{\partial \theta} = -\frac{D_w + D_f + D_v}{\sigma_f}, \tag{2.84}$$

where θ is the direction of the angle of incidence with respect to x, c_x is the group velocity in the x-direction, c_y the group velocity in the y-direction, c_θ the refraction speed, D_w the dissipation by wave breaking, D_f, the dissipation by bottom friction, D_v the dissipation by vegetation, σ_f the intrinsic frequency, and A is the wave energy. Note that in the stationary mode, $\frac{\partial A}{\partial t} = 0$, and the formulations of D_w and D_f are different to those in the surfbeat mode (see Bart (2017) for further details).

In real-world ocean environments, no two waves are alike, and they have varying heights, frequencies and directions. A common technique to deal with this is to use wave spectra (see Roelvink et al., 2015). In this thesis, to model real-world waves in the surfbeat mode, we assume they follow the JONSWAP wave spectrum, which is suitable for oceans in which the unobstructed distance over which the wind blows is limited, *i.e.* fetch-limited oceans (Hasselmann et al., 1973). JONSWAP is inbuilt into XBeach and Figure 2.5 shows a typical JONSWAP wave spectrum using the default XBeach values for the JONSWAP parameters (see Roelvink et al., 2015). We have set the significant wave height to be 2.5 m, where the latter is defined as the mean height of the highest third of waves (Sverdrup, 1947). From the shape of

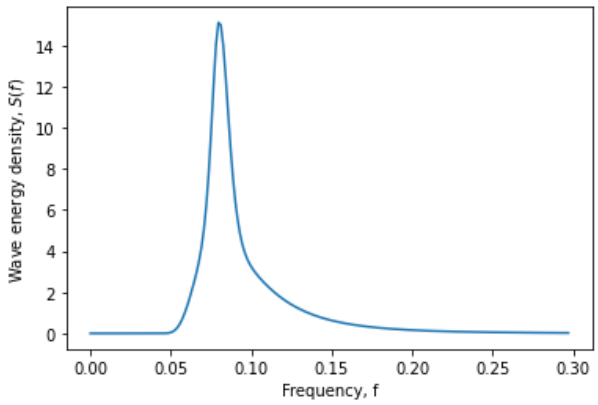


FIGURE 2.5. A typical JONSWAP spectrum in XBeach. Here the significant wave height $= 2.5\,\mathrm{m}$ and the other parameters are the XBeach defaults: $f_p = 0.08s^{-1}$, $\mathring{\gamma} = 3.3$, maximum frequency $= 0.3s^{-1}$ (also known as Nyquist frequency) and $\Delta f =$ (maximum frequency)/200.

this spectrum, XBeach generates a (random) wave time series, the length and resolution of which can be controlled. The general formula for the JONSWAP spectrum, $S(f)$, is

$$S(f) = \frac{f_p^5}{f^5} \exp\left(-\frac{5}{4}\left(\frac{f_p}{f}\right)^4\right) \mathring{\gamma}^q \tag{2.85}$$

where f is the frequency, f_p the peak frequency, $\mathring{\gamma}$ the peak enhancement factor used to alter the spectrum for fetch-limited oceans and

$$q = \exp\left(-\frac{1}{2}\left(\frac{f/f_p - 1}{\sigma_p}\right)^2\right), \tag{2.86}$$

where σ_p is the width of the enhanced peak region. The parameter σ_p can vary in value but its most probable values are 0.07 for $f \leq f_p$ and 0.09 for $f > f_p$ (Torsethaugen and Haver, 2004). For more details please refer to the XBeach source code and Roelvink et al. (2015).

Finally, to model morphological changes in Section 5.3.2.1, XBeach uses the following conservative depth-averaged sediment concentration equation

$$\frac{\partial h\overline{c}}{\partial t} + \nabla_h \cdot (h\overline{\mathbf{u}}^{\mathbf{E}}\,\overline{c}) + \nabla_h \cdot (\epsilon_s h \nabla_h \overline{c}) = \frac{h\,(c_{b*} - \overline{c})}{T_s}, \tag{2.87}$$

which is very similar to the one used in the wet-dry version of our *Thetis* model (see 2.72). In (2.87), $\overline{c}$ is the sediment concentration, $\overline{\mathbf{u}}^{\mathbf{E}}$ the Eulerian velocity, ϵ_s the diffusivity coefficient, c_{b*} the equilibrium sediment concentration and T_s the adaptation time. As discussed in Section 2.2.1.1, there are multiple alternative formulae for c_{b*} and when using XBeach, we choose the Van Thiel-Van Rijn formulation, which is the default in XBeach (see Roelvink et al., 2015). The Exner equation is then used to model bed changes

$$(1 - p')\frac{\partial z_b}{\partial t} + m_f \nabla_h \cdot \mathbf{q} = 0, \tag{2.88}$$

where z_b is the bed, p' the bed porosity, m_f the morphological acceleration factor and $\mathbf{q}$ the sediment transport rates.

Note that in this thesis, we use version 1.23.5526 of the XBeachX release and, unless explicitly stated, parameter values are left at the default for this version.

2.3.3 SFINCS

The final model we describe in this section is the hydrodynamic model SFINCS (Super-Fast INundation of CoastS), also developed by Deltares, which we use in Chapter 6 to simulate coastal problems with waves. SFINCS is the first reduced physics finite difference solver to simulate all the relevant processes for compound coastal flooding include fluvial, pluvial, tidal, wind and wave-driven processes (Leijnse et al., 2021). However, whilst SFINCS does include a wave run-up parametrisation, it does not include a wave equation and thus does not explicitly solve for short wavelength wind-driven waves. This choice was made by the developers so as to maximise computational efficiency. Despite this, when SFINCS is compared to XBeach for both theoretical and real-world test cases in Leijnse (2018) and Leijnse et al. (2021), the difference in accuracy is not extreme. Most importantly, these test cases show that SFINCS is significantly less computationally expensive than XBeach and this makes SFINCS a good low fidelity model to use in Chapter 6 for simulating coastal flooding.

The discretised momentum equation in the x-direction is

$$q_x^{t+\Delta t} = \frac{q_x^t - (gh_x^t(\Delta z/\Delta x) + \mathrm{adv}_x - \tau_{w,x}/\rho_w)\,\Delta t}{1 + g\Delta t\, n_m^2\, q_x^t (h_x^t)^{-7/3}}, \tag{2.89}$$

with the flow in the y-direction being analogous. This equation in SFINCS is adapted from that in the LISFLOOD-FP model in Bates et al. (2010), but both adv_x, an advective term to simulate wave run-up, and $\tau_{w,x}/\rho_w$, a wind-drag term, are terms which are added as part of the SFINCS framework. The other terms in equation (2.89) are as follows: q_x^t and h_x^t are the flow rate and the depth, respectively, at the previous timestep, Δz is the water level difference between two adjacent cells, n_m is the Manning friction coefficient and g is the gravitational constant.

Once the fluxes have been computed, the water elevation on each cell is calculated using

$$\eta_{m,n}^{t+\Delta t} = \eta_{m,n}^t + \left(\frac{q_{x,m-1,n}^{t+\Delta t} - q_{x,m,n}^{t+\Delta t}}{\Delta x} + \frac{q_{y,m,n-1}^{t+\Delta t} - q_{y,m,n}^{t+\Delta t}}{\Delta y} + S_{m,n} \right) \Delta t. \tag{2.90}$$

More details on the SFINCS model can be found in Leijnse (2018) and Leijnse et al. (2021). Note that, in this thesis, we use version 1.0 of SFINCS which was obtained directly from Deltares.

2.4 Validation: Test cases in fully wet domain

We can now use *Sisyphe* (described in Section 2.3.1) together with experimental data to validate our hydro-morphodynamic model in a fully wet domain.

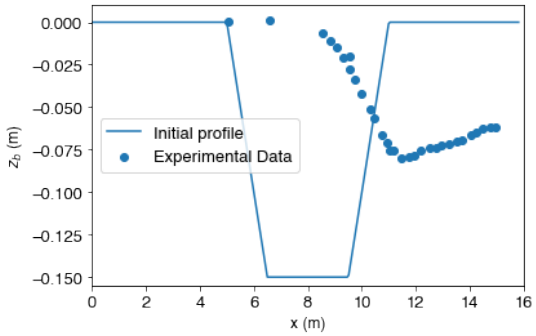


FIGURE 2.6. Initial trench profile and experimental data for the final bedlevel (after 15 h) showing bed evolution in the Trench test case.

TABLE 2.3. Parameter values for the Trench test case from Villaret et al. (2016).

Variable Name	Variable Value
Length in x-direction	16 m
Length in y-direction	1.1 m
Morphological simulation time	15 h
Depth	0.397 m
Downstream elevation	0.4 m
Upstream flux	$0.22\,\mathrm{m^3\,s^{-1}}$
Median particle size (d_{50})	1.6×10^{-4} m
Sediment density (ρ_s)	$2650\,\mathrm{kg\,m^{-3}}$
Water density (ρ_f)	$1000\,\mathrm{kg\,m^{-3}}$
Kinematic eddy viscosity (ν)	$1 \times 10^{-6}\,\mathrm{m^2\,s^{-1}}$
Kinematic molecular viscosity (ν_m)	$1 \times 10^{-6}\,\mathrm{m^2\,s^{-1}}$
Bed sediment porosity (p')	0.4
Diffusivity coefficient (ϵ_s)	$0.01\,\mathrm{m^2\,s^{-1}}$
Nikuradse friction height (k_s)	0.025 m

2.4.1 Migrating trench test case

The first test case we study is that of a migrating trench which is situated in a fully wet domain. This is a common and relatively simple test case (see, for example, Van Rijn (1980), Gerritsen et al. (2008) and Villaret et al. (2016)), for which there exists experimental data from a lab study in Van Rijn (1980). Figure 2.6 shows the initial trench profile for this test case and the final bedlevel profile from the lab study, demonstrating how the trench migrates over time. This test case has also been run with *Sisyphe* in Villaret et al. (2016) and, because *Sisyphe* has been calibrated and validated by experiments, these model results can assist in the validation of our model. In this chapter, we use the same parameter values as in Villaret et al. (2016), and summarise them in Table 2.3. Like in that work, we use a grid of mesh size $\Delta x = 0.2$ m in our *Thetis* model, but set $\Delta y = 0.22$ m rather than using the value from Villaret et al. (2016) ($\Delta y = 0.11$ m). This is because we find in Clare (2018) that, in contrast to *Sisyphe*, our *Thetis* model results are consistent with either Δy, indicating that our model is more robust and meaning that we can adopt the less computationally expensive option. Finally, we set the incoming suspended sediment flow rate so that the erosion flux, E_b, equals the deposition flux, D_b, at the upstream boundary. Hence, suspended sediment equilibrium is established at the inlet and thus the bed is essentially unaltered there, with any small changes being caused by bedload transport.

2.4.1.1 Results

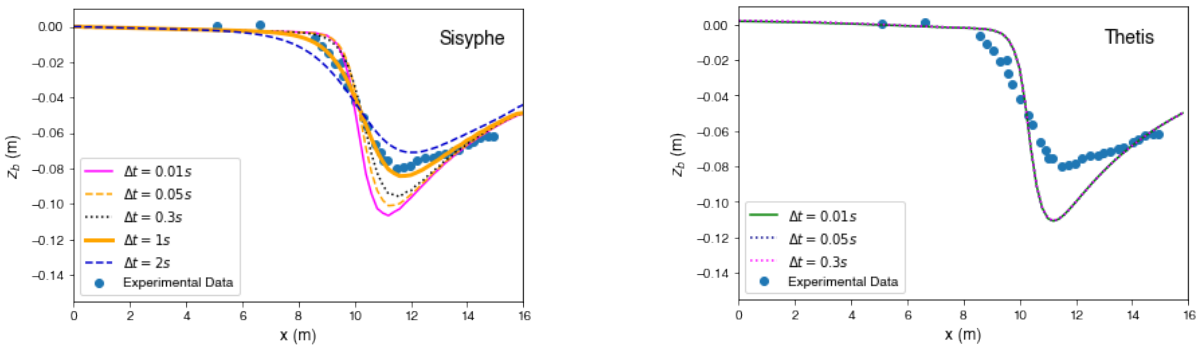
As discussed in the disclaimer at the beginning of this chapter, this test case was presented in the author's MRes (Clare, 2018), as well as published in Clare et al. (2021). To avoid re-submission of work for a different examination, we refer the reader to Clare et al. (2021) for the full test case results. However, a brief summary is presented here because this test case is fundamental for work in Chapters 3 and 4 of this thesis and provides valuable insight into both our *Thetis* model and *Sisyphe*.

We run both *Thetis* and *Sisyphe* for this test case. Before starting the simulation, we follow standard practice (for example Gerritsen et al., 2008) and spin-up the hydrodynamics by running a pure hydrodynamics simulation for 200 s. If we did not do this at the beginning of the simulation, we would be forcing a previously motionless flow and the resulting flow instabilities could trigger unrealistic bedlevel changes. Once the velocity and elevation fields have reached a quasi-steady state, we trigger sediment and bedlevel changes, modelling both suspended sediment and bedload transport. Note that we do not use either the slope effect angle correction or secondary current correction here because both are superfluous in a straight channel. Unless otherwise stated, all figures in this section are produced using a morphological acceleration factor of 10 because, in Clare et al. (2021), we find that there are no noticeable differences between the results for this test case with a morphological acceleration factor of 10 or 1. Although *Sisyphe* has an option for a morphological acceleration factor, it is not imposed here because neither Villaret et al. (2013) or Villaret et al. (2016) apply it.

2.4.1.2 Sensitivity study

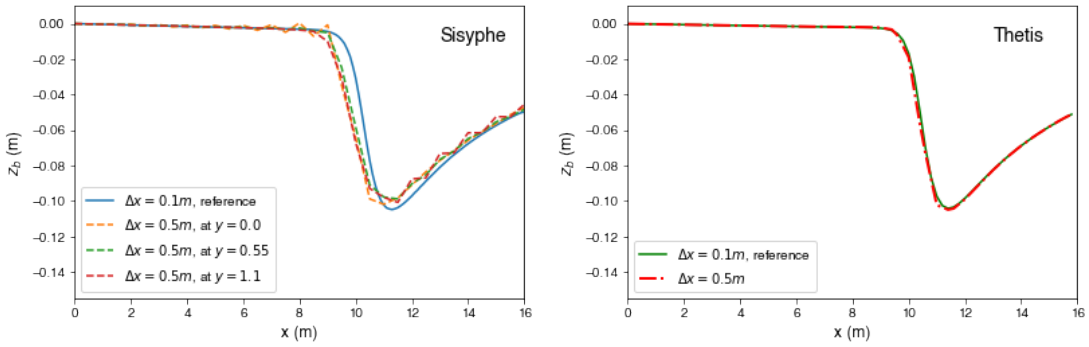
The most significant finding from this test case in Clare et al. (2021) is that *Sisyphe*'s results are very dependent on Δt. Thus, a sensitivity study was conducted on the robustness of both models to small changes in physical parameters, timestep and/or mesh step size. Figure 2.7a shows that the *Sisyphe* bedlevel results vary significantly with Δt. Only when $\Delta t = 1$s, the value used in Villaret et al. (2016), is good agreement found between *Sisyphe* and the experimental data. As Δt decreases, *Sisyphe*'s results converge to the same result as *Thetis* in Figure 2.6. By contrast, bedlevel results from *Thetis* are largely insensitive to changes in Δt, as seen in Figure 2.7b.

When we run a sensitivity study for Δx we find that, for finer meshes than those used in Villaret et al. (2016), both models are insensitive to Δx (see Clare et al., 2021). However, one of the advantages of the DG method is that it is good at dealing with sharp gradients. To illustrate this we run the test case with a significantly coarser mesh of $\Delta x = 0.5$ m. Thus, initially, in the x-direction, each side of the trench is represented by three mesh elements and the flat bottom of the trench by six mesh elements. The mesh nodes are located exactly at the vertices of the slope, meaning that the initial geometry is accurately represented by this mesh. Figure 2.8b shows that, for our *Thetis* model, there are no observable differences between the coarser and finer meshes. On the other hand, Figure 2.8a shows that *Sisyphe* fails to produce an accurate solution. Due to instabilities, the *Sisyphe* solution has also broken symmetry in the y-direction, meaning that it is no longer independent of y, indicating *Sisyphe* has not



(A) *Sisyphe* with $\Delta x = 0.2$m. (B) *Thetis* with $\Delta x = 0.2$m.

FIGURE 2.7. Sensitivity of the final bedlevel to Δt in the Trench test case. Note that the three lines in the right frame are indistinguishable, emphasising our *Thetis* model's robustness.

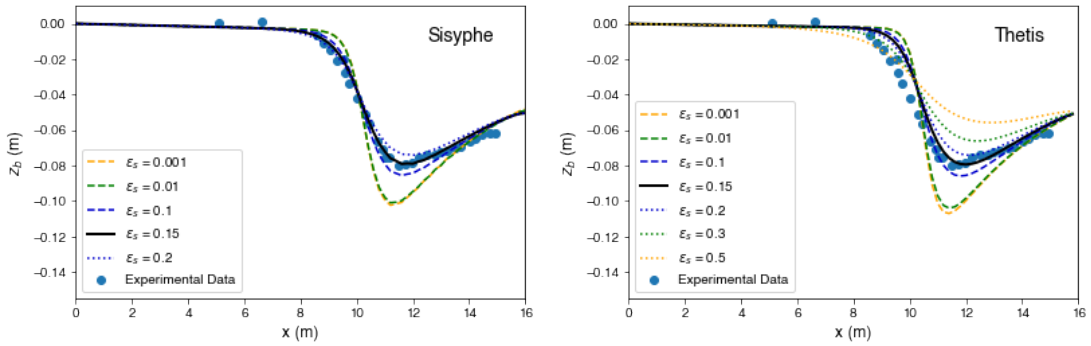


(A) *Sisyphe* ($\Delta t = 0.01\,$s). (B) *Thetis* ($\Delta t = 0.6\,$s).

FIGURE 2.8. Final bedlevel achieved using a very coarse mesh ($\Delta x = 0.5\,$m) compared to that achieved using a reference fine mesh ($\Delta x = 0.1\,$m), for the Trench test case.

converged accurately. To show the solution's dependence on y, in Figure 2.8a, we show the final bedlevel transects at the beginning ($y = 0.0\,$m), midpoint ($y = 0.55\,$m) and end ($y = 1.1\,$m) of the domain (recall that in *Sisyphe*, $\Delta y = 0.11\,$m). By contrast, for our *Thetis* model, Figure 2.8b only needs to show the transect at the midpoint because its solution is effectively independent of y. Given the robustness of *Thetis*, our *Thetis* model results for this test case hereafter apply $\Delta x = 0.5\,$m and $\Delta t = 0.6\,$s, whereas the lack of robustness in *Sisyphe* means that hereafter we must use a spatial resolution 2.5 times finer ($\Delta x = 0.2\,$m) and a timestep 60 times smaller ($\Delta t = 0.01\,$s), for *Sisyphe* results.

We conjecture that the errors caused in *Sisyphe* with larger Δt values manifest themselves as an increase in effective diffusivity in the model because Figure 2.7 shows that for small values of Δt, *Thetis* and *Sisyphe* are consistent. Thus, we conduct a sensitivity study for the diffusivity coefficient, ϵ_s. Bedlevel results from both *Sisyphe* and *Thetis* in Figures 2.9a and 2.9b show they are indeed greatly affected by ϵ_s and, importantly, that both models behave consistently. Note that, due to stability constraints, *Sisyphe* does not run with $\epsilon_s > 0.2\,\mathrm{m^2\,s^{-1}}$, unlike *Thetis*. The observed sensitivity to ϵ_s is to be expected because the grid Peclet number ($U\Delta x/\epsilon_s$) decreases with ϵ_s, making diffusion the key driver of the sediment



(A) *Sisyphe* ($\Delta t = 0.01\,\mathrm{s}$). (B) *Thetis* ($\Delta t = 0.6\,\mathrm{s}$).

FIGURE 2.9. Sensitivity of final bedlevel to diffusivity coefficient, ϵ_s, in the Trench test case.

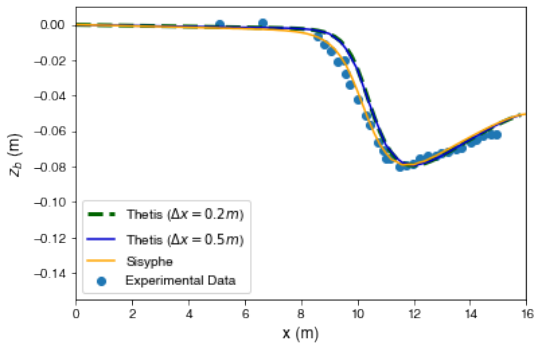


FIGURE 2.10. Final bedlevel from our *Thetis* model (at two different resolutions) and from *Sisyphe* with $\epsilon_s = 0.15\,\mathrm{m^2 s^{-1}}$ in both models, for the Trench test case.

concentration equation, rather than advection. Thus, we can use ϵ_s to calibrate both models; in *Sisyphe*, Δt can be used to similar effect. If we set $\epsilon_s = 0.15\,\mathrm{m^2\,s^{-1}}$, *Thetis* and *Sisyphe*'s converged results agree well with each other and with the experimental data, as shown clearly in Figure 2.10. Thus, we have validated *Thetis* for this simple test case.

2.4.2 Meander test case

The second test case we study is the more complex curved channel of a meander in a fully wet domain, which requires and demonstrates the implementation of a slope effect angle correction and a secondary current correction. The motivation for studying this test case is to validate these new model functionalities and confirm our model is capable of handling more complex test cases. We highlight here that this test case is new work conducted as part of this PhD and was not submitted as part of the author's MRes.

For this test case, we use the configuration from experiment 4 from Yen and Lee (1995), which has already been studied using *Sisyphe* in Villaret et al. (2013). We validate our *Thetis* model through the experimental data and *Sisyphe* results from Villaret et al. (2013). Most of the bed changes occur at the boundary so, following Villaret et al. (2013), we use a finer mesh there (Δx =0.1 m) and a coarser one ($\Delta x = 0.25\,\mathrm{m}$) along the centre of the channel, as shown in Figure 2.11. We also impose

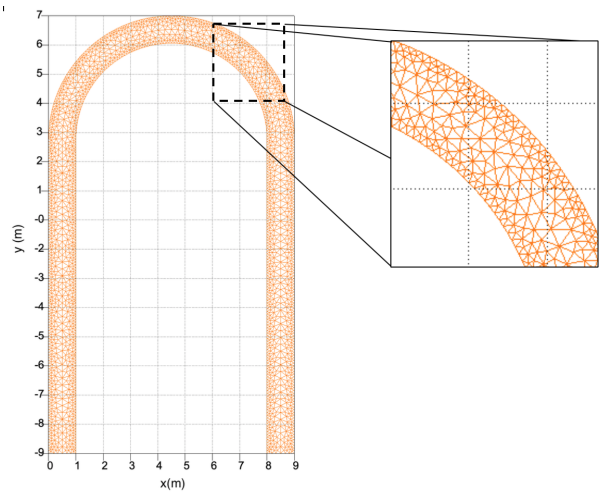


FIGURE 2.11. Mesh and domain for the Meander test case used both by us in our *Thetis* model and by Villaret et al. (2013) in *Sisyphe*.

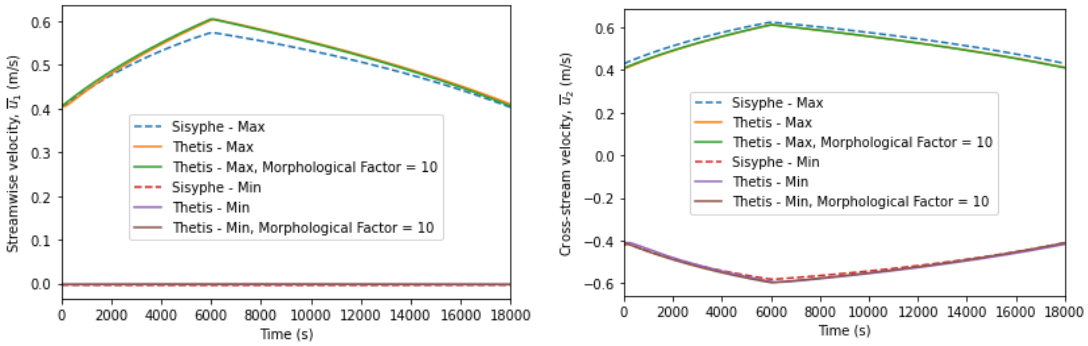
a free-slip condition on the meander boundary walls and impose the same time dependent flux and elevation boundary conditions at the inflow and outflow as in Yen and Lee (1995). The initial inflow flux and outflow elevation are $0.02\,\mathrm{m^3\,s^{-1}}$ and $0\,\mathrm{m}$, respectively. Both increase linearly until reaching their respective maximums of $0.053\,\mathrm{m^3\,s^{-1}}$ and $0.103\,\mathrm{m}$ at $100\,\mathrm{min}$, and then decrease linearly to their initial values at $5\,\mathrm{h}$. For the parameter values for this test case, we follow Villaret et al. (2013) and summarise the values used in Table 2.4. Note that, still following Villaret et al. (2013), we only simulate bedload transport because this is the principal sediment transport component in rivers and hence, we do not need to specify the diffusivity coefficient ϵ_s. In order to set the value of α' in (2.50) for the secondary current correction, we must determine the flow roughness. Following Kulkarni and Sahoo (2013), we calculate that the roughness Reynolds number, defined by $(k_s\sqrt{\tau_b})/(\nu\sqrt{\rho_f})$, is of order 80, and conclude that the case is subject to a rough turbulent flow regime. Consistently with Tassi and Villaret (2014), we therefore use $\alpha' = 0.75$.

TABLE 2.4. Parameter values for the Meander test case from Villaret et al. (2013).

Variable Name	Variable Value
Channel width	$1\,\mathrm{m}$
Inner radius	$3.5\,\mathrm{m}$
Outer radius	$4.5\,\mathrm{m}$
Straight reach at channel ends	$11.5\,\mathrm{m}$
Morphological simulation time	$5\,\mathrm{h}$
Depth	$0.0544\,\mathrm{m}$
Median particle size (d_{50})	$1 \times 10^{-3}\,\mathrm{m}$
Sediment density (ρ_s)	$2650\,\mathrm{kg\,m^{-3}}$
Water density (ρ_f)	$1000\,\mathrm{kg\,m^{-3}}$
Kinematic molecular viscosity (ν_m)	$1 \times 10^{-6}\,\mathrm{m^2\,s^{-1}}$
Bed sediment porosity (p')	0.4
Nikuradse friction height (k_s)	$0.0035\,\mathrm{m}$

2.4.2.1 Modelling the hydrodynamics

As this is a more complex case than trench migration, we first run the simulation with no morphological components in both our model and *Sisyphe* to check our model accurately simulates the hydrodynamics. We could use the same viscosity value in this test case as we did in the migrating trench one



(A) Streamwise velocity, $\overline{u}_1$.

(B) Cross-stream velocity, $\overline{u}_2$.

FIGURE 2.12. Minimum and maximum velocities of hydrodynamic flow from *Thetis* ($\nu = 0.035\,\mathrm{m^2\,s^{-1}}$, slope = 0.0035) with a morphological acceleration factor of 1 and 10, and *Sisyphe*, present study, ($\nu = 0.01\,\mathrm{m^2\,s^{-1}}$, slope = 0.002), for the Meander test case.

($\nu = 1 \times 10^{-6}\,\mathrm{m^2\,s^{-1}}$) but this test case is at a much larger scale and includes turbulence effects. As such, the viscosity is accounting for the turbulence or eddy viscosity (see 2.5) and a large value of order 1×10^{-3} (the value Vouriot et al. (2019) use for their *Thetis* test case), or even order 1×10^{-2} (the value used in Villaret et al. (2013) to simulate this test case in *Sisyphe*) is more appropriate.

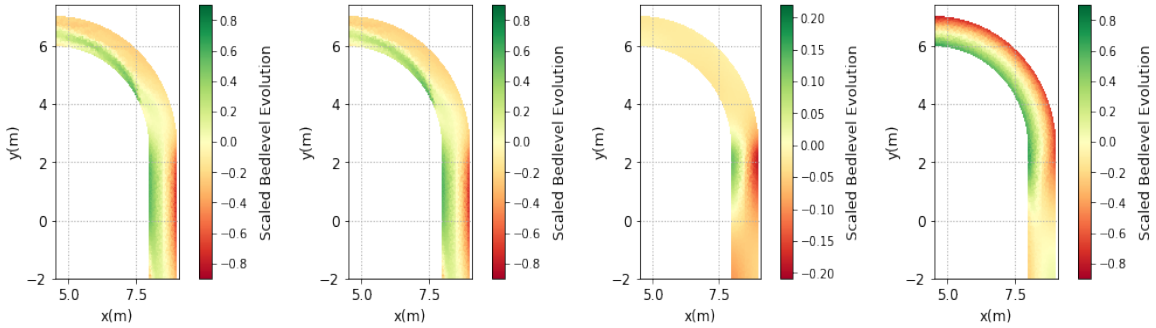
As we are not using a turbulence model, we use *Sisyphe*'s hydrodynamic results to calibrate our model and find an appropriate value of viscosity, ν. These alterations in ν change the nature of the test case, but can be balanced by altering the longitudinal bed slope. In Yen and Lee (1995), the meander has a longitudinal bed slope of 0.002. However, we find that using a longitudinal bed slope of 0.0035 and ν of $0.035\,\mathrm{m^2\,s^{-1}}$ means our model's velocities match those in *Sisyphe* reasonably well, as shown in Figures 2.12a and 2.12b. These figures also show that even with time dependent boundary conditions, using a morphological acceleration factor equal to either 1 or 10 in *Thetis* gives equivalent results. Thus, unless otherwise stated, in this section our *Thetis* results are produced with a morphological acceleration factor of 10.

2.4.2.2 Modelling sediment transport

We now introduce sediment transport into the models. As in the previous test case, we first simulate the hydrodynamics until the model reaches a quasi-steady state, so as to avoid unrealistic bedlevel changes. Our initial hydrodynamic only simulation is run for 200 s with a fixed flux inflow of $0.02\,\mathrm{m^3\,s^{-1}}$ and outflow elevation of 0 m. For our full sediment transport simulation, we use these hydrodynamics results as initial flow conditions and impose the time dependent flux and elevation boundary conditions described earlier. For our results for this test case, we follow Villaret et al. (2013) and present the scaled bedlevel evolution, defined as

$$\text{Scaled Bedlevel Evolution} = \frac{z_{\text{final}} - z_{\text{initial}}}{z_{\text{initial}}} \tag{2.91}$$

where z_{final} is the final bedlevel after 5 h and z_{initial} is the initial bedlevel of -0.0544 m. It should be noted that, even by altering the parameters in Table 2.4, we are unable to accurately recreate the results



(A) No physical corrections.

(B) Only slope effect magnitude.

(C) Both slope effect corrections (note smaller colour scale).

(D) All physical corrections.

FIGURE 2.13. Section of Meander test case showing scaled bedlevel evolution from *Thetis* with different physical corrections to $\mathbf{Q_b}$.

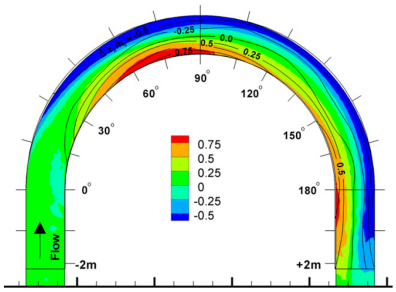


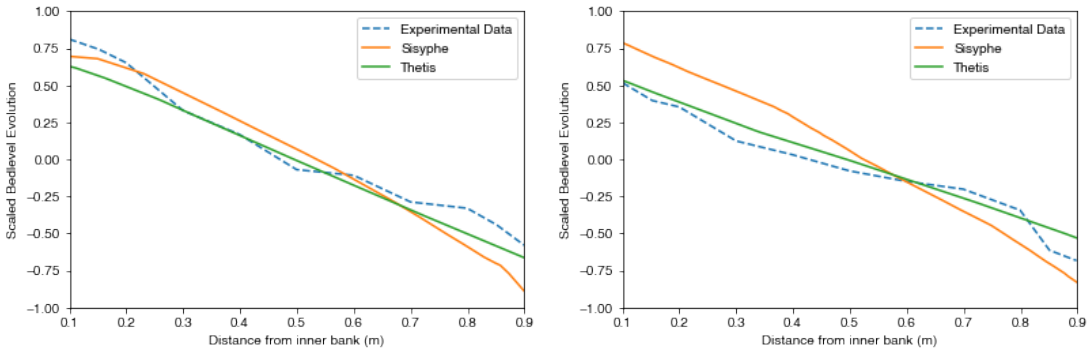
FIGURE 2.14. Full scaled bedlevel evolution from *Sisyphe* (coloured bars) compared to experimental data (black contours) for the Meander test case. Source: Villaret et al. (2013).

in Villaret et al. (2013) and thus, unless otherwise stated, for this test case, *Sisyphe* results are taken from the latter.

Figure 2.13 shows how the secondary current and slope effect corrections in our *Thetis* model affect the bedlevel evolution at the meander outflow. The slope effect magnitude correction has little effect compared to the secondary current and slope effect angle corrections, likely because the slopes in this test case are fairly gentle. Figure 2.13d shows that implementing all effects results in erosion at the outer bend and deposition at the inner bed, as expected from physical intuition. Comparing this figure with Figure 2.14 from Villaret et al. (2013), we see the *Thetis* result has the same distribution and order of magnitude as the experiment and *Sisyphe*. To compare our *Thetis* result with the experiment and *Sisyphe*'s results more accurately, we take a cross-section at the 90° and 180° angles marked on Figure 2.14. Figures 2.15a and 2.15b show our model reproduces the experimental results better than *Sisyphe*, with a particularly good improvement at the 180° cross-section and in the bedlevel erosion at both cross-sections.

2.4.2.3 Calibration study

In Section 2.4.2.1, we used the hydrodynamic results from *Sisyphe* to calibrate the viscosity and longitudinal slope in *Thetis* in the absence of observed data. However, Figures 2.15a and 2.15b show *Sisyphe*



(A) Cross-section at 90°. (B) Cross-section at 180°.

FIGURE 2.15. Scaled bedlevel evolution at specific cross-sections from *Thetis* (with $\nu = 0.035\,\mathrm{m^2\,s^{-1}}$, slope = 0.0035), *Sisyphe* (Villaret et al., 2013) and experimental data (Yen and Lee, 1995), for the Meander test case.

TABLE 2.5. Sum of error norms for different values of longitudinal slope and ν ($\mathrm{m^2\,s^{-1}}$) in the Meander test case.

Slope	$\boldsymbol{\nu = 0.025}$	$\boldsymbol{\nu = 0.035}$	$\boldsymbol{\nu = 0.05}$	$\boldsymbol{\nu = 0.075}$
0.003	0.5041	0.4934	0.4847	0.4930
0.0035	0.4911	0.4828	0.4752	0.4851
0.004	0.5253	0.5167	0.5106	0.5199
0.0045	0.5809	0.5707	0.5635	0.5686

does not agree completely with the experimental data. Hence, to improve our model's accuracy, we re-run the calibration study with the full hydro-morphodynamic model, using the experimental data as the 'real solution'. We emphasise here that the viscosity value being altered is that used in the hydrodynamic equations of the model; throughout this thesis the sediment equations use the molecular viscosity, ν_m, set equal to $1 \times 10^{-6}\,\mathrm{m^2 s^{-1}}$.

For our calibration study, we seek to minimize the error at both the 90° and 180° cross-section. Thus, instead of just minimising one norm or the other, we minimise the sum of the two norms:

$$\frac{\left\|y_{90_i} - \mathring{y}_{90_i}\right\|_2}{\left\|\mathring{y}_{90_i}\right\|_2} + \frac{\left\|y_{180_i} - \mathring{y}_{180_i}\right\|_2}{\left\|\mathring{y}_{180_i}\right\|_2}, \tag{2.92}$$

where $\mathring{y}_i$ is the experimental data and y_i our model result. The results are summarised in Table 2.5 and show that a viscosity of $0.05\,\mathrm{m^2\,s^{-1}}$ and a longitudinal slope of 0.0035 yield the best approximation to the experimental data. Minimising the sum of the two norms means that we could merely be reducing the error at one cross-section whilst allowing it to grow at the other one. Therefore as a sanity check we also record the maximum value of the 90° and 180° cross-section norms:

$$\max\left(\frac{\left\|y_{90_i} - \mathring{y}_{90_i}\right\|_2}{\left\|\mathring{y}_{90_i}\right\|_2}, \frac{\left\|y_{180_i} - \mathring{y}_{180_i}\right\|_2}{\left\|\mathring{y}_{180_i}\right\|_2}\right). \tag{2.93}$$

The results are summarised in Table 2.6 and show that although $\nu = 0.05\,\mathrm{m^2\,s^{-1}}$ and a longitudinal slope of 0.0035 do not minimize (2.93), they result in one of the smallest values.

TABLE 2.6. Maximum of the 90° and 180° cross-section error norms for different values of longitudinal slope and ν ($m^2\,s^{-1}$) in the Meander test case.

Slope	$\boldsymbol{\nu = 0.025}$	$\boldsymbol{\nu = 0.035}$	$\boldsymbol{\nu = 0.05}$	$\boldsymbol{\nu = 0.075}$
0.003	0.2734	0.2768	0.2834	0.3035
0.0035	0.2463	0.2534	0.2674	0.2978
0.004	0.2849	0.2644	0.2729	0.3134
0.0045	0.3332	0.3096	0.2862	0.3327

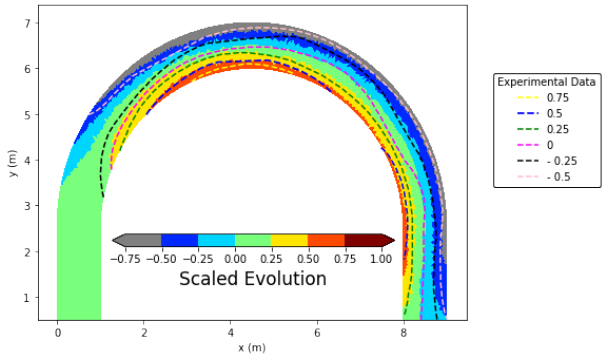


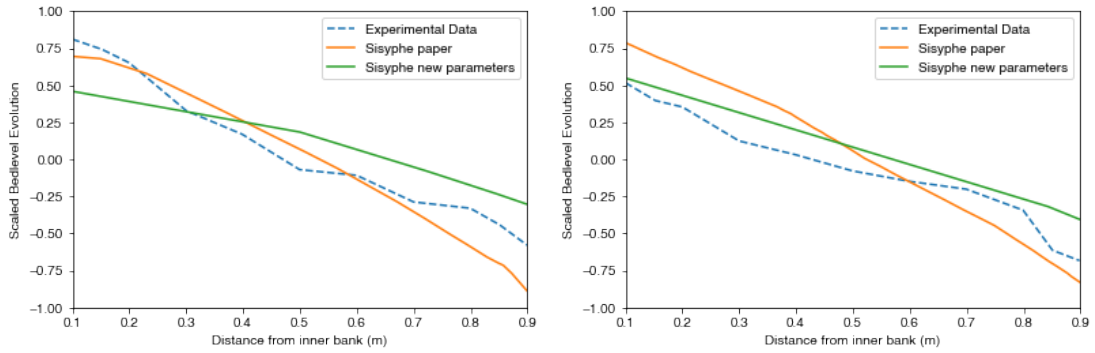
FIGURE 2.16. Full scaled bedlevel evolution from *Thetis* with $\nu = 0.05\,m^2\,s^{-1}$, slope = 0.0035 compared to experimental data (Yen and Lee, 1995), for the Meander test case.

Figure 2.16 shows that the full scaled bedlevel evolution results from our model (using these optimised ν and longitudinal slope values) agree well with the experimental data. In particular, when we compare this figure with Figure 2.14, which shows the original *Sisyphe* full scaled bedlevel evolution results from Villaret et al. (2013), we see that our model predicts the bedlevel erosion to a greater degree of accuracy, particularly at the outer bend. Our *Thetis* model also results in more uniform erosion at the inflow bedlevel (Figure 2.16), unlike *Sisyphe* (Figure 2.14), although neither model predicts the inflow bedlevel particularly accurately.

For rigour, we run *Sisyphe* with these optimised values for viscosity and longitudinal slope. The resulting bedlevel change is shown in Figures 2.17a and 2.17b at the 90°and 180° cross-sections, respectively. There is a marginal improvement in the total relative error norm (2.92), which falls from 1.144 for the results from Villaret et al. (2013) to 1.067 for these optimised values. However, the errors of *Sisyphe* are still higher than those obtained for *Thetis* (Table 2.5).

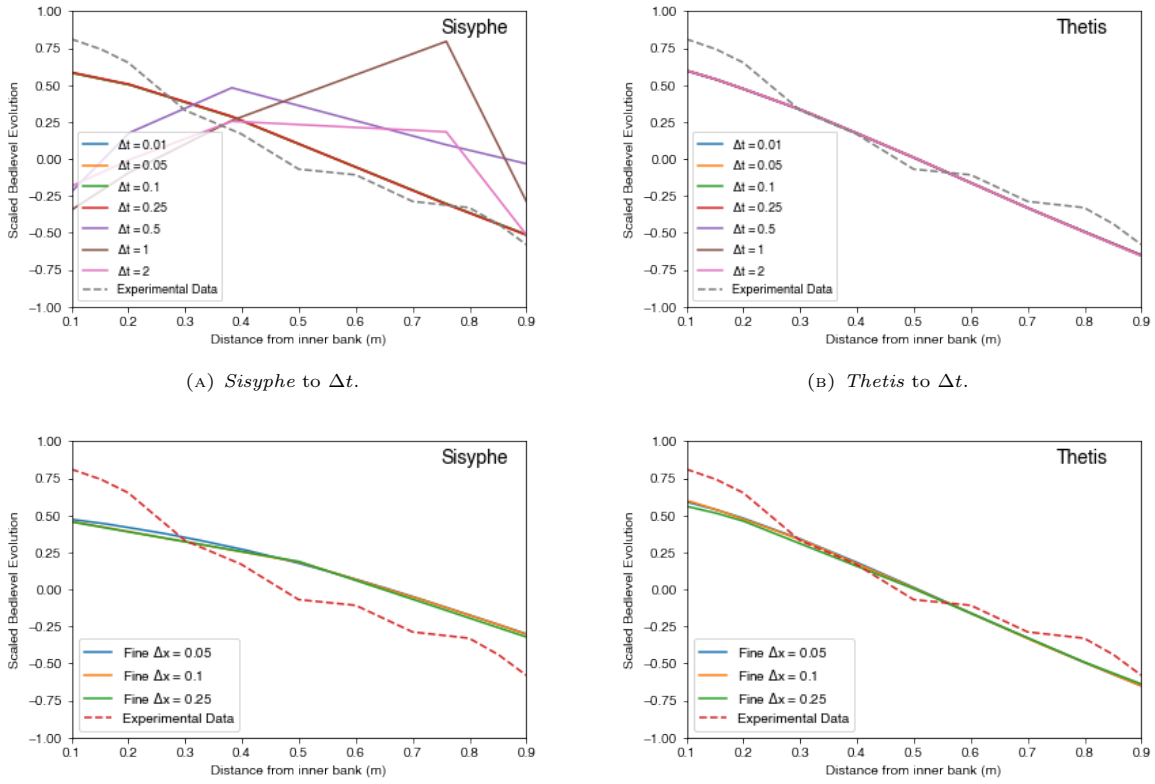
2.4.2.4 Sensitivity Study

Given *Sisyphe*'s sensitivity to Δt discussed in Section 2.4.1.2, we conduct a sensitivity study on Δt and Δx. Note that for simplicity, all quoted Δx values for this test case are the values for the fine part of the mesh and the ratio between the fine and coarse parts of the mesh is maintained at 2:5, as in Figure 2.11. We configure *Sisyphe* ourselves using our optimised viscosity and longitudinal slope values for consistency with *Thetis*. Figures 2.18a and 2.18b show that while *Thetis* is insensitive to Δt, *Sisyphe* is sensitive to Δt, as was also the case in Section 2.4.1. Although for $\Delta t \leq 0.25\,s$ *Sisyphe*'s results are robust, for larger Δt values, they are both sensitive and inaccurate. Furthermore, for this test case, *Thetis* converges for



(A) Cross-section at 90°. (B) Cross-section at 180°.

FIGURE 2.17. Scaled bedlevel evolution at specific cross-sections from *Sisyphe*, Villaret et al. (2013); *Sisyphe*, present study, (with $\nu = 0.05\,\mathrm{m^2\,s^{-1}}$ and slope = 0.0035); and experimental data (Yen and Lee, 1995), for the Meander test case.

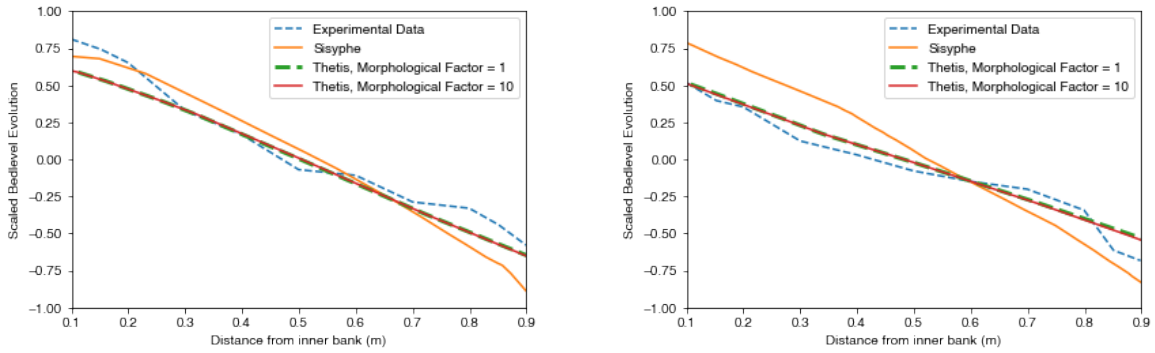


(A) *Sisyphe* to Δt. (B) *Thetis* to Δt.

(C) *Sisyphe* to Δx. (D) *Thetis* to Δx.

FIGURE 2.18. Sensitivity of bedlevel to Δx and Δt at the 90° cross-section for the Meander test case, with *Sisyphe* results from the present study. Note that the lines in the top right frame are indistinguishable, emphasising our *Thetis* model's robustness.

$\Delta t < 10\,\mathrm{s}$, meaning it is much less computationally expensive than *Sisyphe*, which requires $\Delta t \leq 0.25\,\mathrm{s}$. In contrast, Figures 2.18c and 2.18d show that both models are relatively insensitive to the mesh step size Δx. However, for both models, using $\Delta x = 0.25\,\mathrm{m}$ results in a slightly different bedlevel to that from using finer meshes, and thus we use $\Delta x = 0.1\,\mathrm{m}$ instead for our results.



(A) Cross-section at 90°.

(B) Cross-section at 180°.

FIGURE 2.19. Scaled bedlevel evolution at specific cross-sections from *Thetis* ($\nu = 0.05\,\mathrm{m^2\,s^{-1}}$ and slope = 0.0035) with a morphological acceleration factor of 1 and 10; *Sisyphe*, Villaret et al. (2013); and experimental data (Yen and Lee, 1995), for the Meander test case.

Finally, Figure 2.19 provides an overview of our results and shows not only that we have validated our model, but that it is more accurate than *Sisyphe* for this more complex test case. It also confirms that a morphological acceleration factor of 10 is appropriate with no observable difference between a morphological acceleration factor of 10 and one of 1 (equivalent to no scaling).

2.4.3 Benchmarking test cases in fully wet domain

Finally, we compare the computational times and error norms of *Thetis* and *Sisyphe* for both fully wet test cases discussed in this section and summarise the results in Table 2.7. We highlight here that the benchmarking for both test cases is new work conducted in this PhD and was not submitted as part of the author's MRes.

Our benchmarking study finds that for the more complex geometry of the meander, our model is approximately twice as accurate as *Sisyphe* using the same mesh. For the migrating trench, we find that *Sisyphe* is more accurate than our model, but this is only true for this specific choice of mesh resolution and timestep in *Sisyphe*. As the timestep increases and the mesh becomes coarser, the accuracy of the *Sisyphe* results decreases (see Figure 2.7a and Figure 2.8a, respectively), whereas the accuracy of our model results stays broadly the same.

The robustness advantages observed with *Thetis*'s DG-based discretisation deliver accurate results with both larger Δt and Δx values. Without using a morphological acceleration factor, *Thetis* is slower, partly because our DG discretisation possesses significantly more degrees of freedom than the CG discretisation used in *Sisyphe*. For the P1-DG elements we use, each mesh element has three degrees of freedom, whereas for P1-CG elements the number of degrees of freedom are the number of nodes in the mesh. For structured meshes, as well as unstructured meshes generated using standard mesh-generation algorithms such as Delaunay, the number of nodes is typically approximately twice the number of elements, meaning DG has approximately six times more degrees of freedom that CG (Cotter et al., 2009). However, the added robustness from DG means we can apply a morphological acceleration factor to our model

to reduce computational times without compromising accuracy. Table 2.7 presents the accuracy and efficiency results of applying a morphological acceleration factor. It shows that, using a morphological acceleration factor of 50 in the migrating trench and meander test cases, our model is ten times and two times more efficient than *Sisyphe*, respectively, and retains accuracy.

TABLE 2.7. Computational cost, t_s (seconds) and L2 error norm to data for the Trench and Meander test cases using *Thetis* and *Sisyphe* with different morphological acceleration factors. For the Trench, $\Delta t = 0.01\,\text{s}$ and $\Delta x = 0.2\,\text{m}$ in *Sisyphe* and $\Delta t = 0.6\,\text{s}$ and $\Delta x = 0.5\,\text{m}$ in *Thetis*; for the Meander $\Delta t = 0.1\,\text{s}$ in *Sisyphe* and $\Delta t = 2\,\text{s}$ in *Thetis* and a fine $\Delta x = 0.1\,\text{m}$ in both.

Model	**Morphological Factor**		**Trench** t_s (s)	**Meander** t_s (s)		**Trench** $L2$ error (m)	**Meander** $L2$ error (m)
Thetis	1		66,452	17,785		0.04135	0.4752
Thetis	10		6590	2140		0.04084	0.4751
Thetis	25		2646	913		0.03920	0.4722
Thetis	50		1386	450		0.03666	0.4741
Sisyphe	1		14,113	980		0.01756	1.067

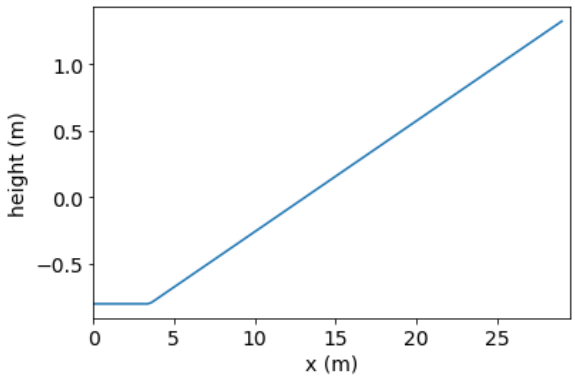
2.5 Validation: Test cases in wet-dry domain

Having validated our hydro-morphodynamic model in a fully wet domain, we now consider a test case to validate our model in a wet-dry domain, using experimental data and model outputs from *XBeach* (described in Section 2.3.2). We emphasise that this test case is new work conducted in this PhD and was not submitted as part of the author's MRes.

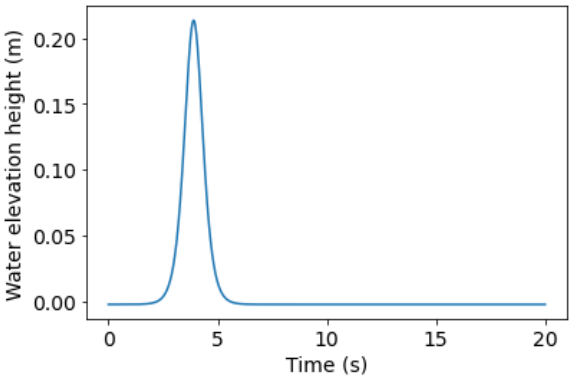
2.5.1 Tsunami-like wave test case

For our validation test case in a wet-dry domain, we consider a tsunami-like event, which is often difficult to simulate due to the large array of uncertain input parameters (as discussed in more detail in Section 3.5). This is especially true for historical tsunami events where the only record is sediment deposits (see for example Dourado et al., 2021). This makes tsunami events an ideal case to apply our hydro-morphodynamic adjoint framework in Chapter 3. For our test case, we consider the experimental data from Kobayashi and Lawrence (2004), where a tsunami-like solitary wave repeatedly breaks over a sloping beach. The initial profile of the sloping beach is shown in Figure 2.20a. It should be noted that the hydrodynamic component of the model described in Section 2.2 is governed by the non-linear shallow water equations, which are not able to exactly represent the propagation of a solitary wave (Barthélemy, 2004; Kazhyken et al., 2021). Therefore, this test case will also indicate whether using shallow water equations results in a substantial divergence from the experimental set-up. Given the small size of the domain, it is not unreasonable to expect that the shallow water equations will be able to maintain the shape of the solitary wave.

For this test case, we use the wet-dry version of our model with the conservative sediment equation to ensure sediment does not leak (see Section 2.2.2). Following (Goto et al., 2011), we do not simulate the bedload transport $\mathbf{Q_b}$ because studies have shown that sediment transport due to tsunami waves



(A) Initial bed profile.



(B) Free surface elevation boundary condition at the inflow defined by (2.94).

FIGURE 2.20. Set-up for the Tsunami test case.

mainly occurs due to suspended sediment. However, the steep beach slope means that we do use the sediment slide mechanism (see Section 2.2.2.2). Thus, this test case is used to validate the conservative sediment equation and sediment slide functionality. This case has already been modelled using XBeach in Li and Huang (2013), and we use these model outputs along with the experimental data to validate our model.In Li and Huang (2013), they also assess which suspended sediment transport formulae are most appropriate for this type of test case and conclude that the formula in Van Rijn (1984) which we use in our hydro-morphodynamic model in *Thetis* (see Section 2.2.1.1) is one of the most appropriate, which is a promising sign for our model.

Following Kobayashi and Lawrence (2004), the wave is simulated by imposing the following free surface elevation boundary condition at the inflow boundary

$$\eta(t) = H_{\text{wave}} \operatorname{sech}^2\left(\sqrt{\frac{3H_{\text{wave}}}{4h}}\frac{\sqrt{g(H_{\text{wave}}+h)}}{h}(t-t_{\max})\right) + \eta_{\text{down}}, \tag{2.94}$$

which causes a tsunami-like solitary wave to travel into the domain (see Figure 2.20 for a pictorial representation). Here, H_{wave} is the average wave height, h the still water depth, $t_{\max}$ the arrival time of the wave crest at the inflow boundary and η_{down} the initial decrease of the elevation at the beginning of the simulation (also the initial elevation in the domain). Our model cannot currently simulate shoaling and breaking waves, hence a relatively high viscosity value of $\nu = 0.8\,\text{m}^2\text{s}^{-1}$ is used in the hydrodynamics to dissipate energy. This is standard practice, and for example Li and Huang (2013) view viscosity as a model calibration parameter for energy dissipation, rather than a physical parameter.

In this test case, we simulate friction using the Chezy friction formula defined by

$$C_{\text{chezy}} = \frac{g}{n^2}, \tag{2.95}$$

where n is the Chezy friction parameter, following Li and Huang (2013). We use the latter work and Kobayashi and Lawrence (2004) to obtain the remaining parameter values summarised in Table 2.8. Note that Li and Huang (2013) run the simulation for 40 s with $t_{\max} = 23.9\,\text{s}$ for each solitary wave,

but the system is stationary for the first 20 s. Unlike with the previous test cases considered, here it is not necessary to spin up the hydrodynamics first because the initial set-up of the experiment has zero velocity and zero elevation. Therefore, out of the 40 s simulation, we discard the first 20 s and only run our model for 20 s with $t_{\max} = 3.9$ s for each solitary wave. Furthermore, in preliminary tests, we found that we can set a morphological acceleration factor as high as four without any noticeable differences between the sped-up results and the results with no morphological acceleration. This is in part because in the experiment the waves are purposefully sufficiently distant from each other that they do not interact. Hence, here we use that value, meaning we only need to model two solitary waves to approximately simulate the bed changes caused by the eight waves in the experiment.

TABLE 2.8. Parameter values for the Tsunami test case.

Variable Name	**Variable Value**
Length in x-direction	30 m
Length in y-direction	4 m
Bed slope	1/12
$\Delta x = \Delta y$	0.2 m
dt	0.05 s
Morphological simulation time	20 s × 2
Morphological acceleration factor	4 (thus 2 waves only)
Median particle size (d_{50})	1.8×10^{-4} m
Sediment density (ρ_s)	2650 kg m^{-3}
Water density (ρ_f)	1000 kg m^{-3}
Kinematic eddy viscosity (ν)	0.8 m^2 s^{-1}
Kinematic molecular viscosity (ν_m)	1×10^{-6} m^2 s^{-1}
Bed sediment porosity (p')	0.4
Diffusivity coefficient (ϵ_s)	1 m^2 s^{-1}
Chezy friction coefficient (n)	65 m$^{1/2}$s^{-1}
Angle of repose	22°
Wetting-and-drying parameter	1/30 m
Norm smoother parameter (β)	1/60 m s^{-1}
H_{wave}	0.216 m
h	0.18 m
η_{down}	−0.0025 m
$t_{\max}$	3.9 s

2.5.2 Results

Figure 2.21 compares the final bedlevel from using our model to simulate this experiment with the final bedlevel from using XBeach (taken from Li and Huang, 2013) and the experimental data (taken from Kobayashi and Lawrence, 2004). There is good agreement between our model results and both the experimental data and the XBeach result, despite the fact that XBeach has additional wave parametrisations not present in our model (see Section 2.3.2). Both models do not accurately capture the area of deposition between 5 m and 10 m and we address this issue in Chapter 3. Nevertheless, for comparison, the agreement between our model and the experimental data is more than competitive with that shown between the results and the experimental data for a similar test case in Kazhyken et al. (2021) which uses a dispersive wave model. Note that we chose not to do a full sensitivity analysis for Δx and Δt because we have already shown how robust *Thetis* is in Section 2.4. Thus, our model is validated for this test

case, suggesting that using shallow water equations instead of a dispersive wave model does not result in substantial divergence from the experimental set-up.

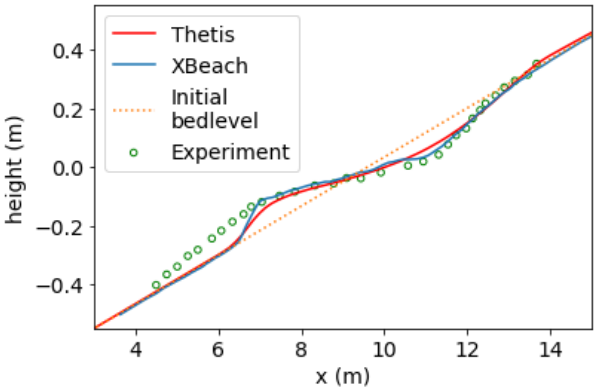


FIGURE 2.21. Final bedlevel simulated by *Thetis* compared to experimental data from Kobayashi and Lawrence (2004) after eight solitary waves for the Tsunami test case.

2.6 Conclusion

In this chapter, we have developed a new 2D depth-averaged coupled hydro-morphodynamic model within the finite element based coastal ocean model *Thetis* with a DG discretisation. Through applying our model to test cases in both fully wet and wet-dry domains, we have validated it and shown that it makes significant and novel contributions to the complex problem of modelling sediment transport. For all test cases considered, our model is accurate with respect to experimental data. Moreover, for fully wet cases, our model is more efficient and robust than the widely-used model *Sisyphe* and for wet-dry cases it is as accurate as the XBeach, despite the latter model representing more complex physics than ours.

The coupled and nonlinear nature of the hydro-morphodynamic problems makes this type of model very sensitive to parameter changes. However, *Thetis* is found to be largely insensitive to changes in timestep and mesh grid size, unlike *Sisyphe*, which is found to have a much larger variability, particularly with respect to the timestep in the test cases considered in this thesis. The robustness of *Thetis* is due, in part, to the novel use of a DG discretisation in a full hydro-morphodynamic model. Moreover, this robustness enables the application of a morphological acceleration factor to our model, increasing its computational efficiency relative to existing models, whilst still maintaining accuracy.

Code availability

The relevant code for our hydro-morphodynamic model and the validation of the migrating trench and meander test cases is stored at `https://github.com/mc4117/morphodynamic_model`. The relevant code for the validation of the tsunami-like wave test case is stored at `https://github.com/mc4117/adjoint_hydro_morphodynamic/tree/main/Section_6_tsunami-like_wave`.

CHAPTER 3

Calibration, inversion and sensitivity analysis through the application of the adjoint method

Highlights

- Development of the first freely available, fully flexible, computationally efficient adjoint hydro-morphodynamic model framework;
- Application of the adjoint method to analyse sensitivity of model outputs to spatially-varying input parameters;
- Use of the adjoint method to calibrate for multiple uncertain parameters simultaneously in an experiment;
- Inversion of a tsunami-like wave using experimental sediment deposit data.

Summary

Hydro-morphodynamic models can be affected by a high degree of uncertainty, in part due to incomplete knowledge of various physical, empirical and numerical closure related parameters in both the hydrodynamic and morphodynamic solvers. Here, we address this uncertainty through the application of the adjoint method, which has the notable advantage that the number and/or dimension of the uncertain parameters has almost no effect on the computational cost associated with estimating the model sensitivities.

We develop the first freely available, fully flexible, computationally efficient adjoint hydro-morphodynamic model framework, using our *Thetis* model presented in Section 2.2 as the hydro-morphodynamic model. The flexibility in our framework is achieved through using the *pyadjoint* library, which allows us to assess how uncertainty in any parameter affects any output functional without further code implementation. We present examples of how this framework can be used to perform sensitivity analysis, inversion and calibration for a variety of uncertain parameters based on the final bedlevel. These results are verified using so-called dual-twin experiments, where the 'correct' parameter value is used in the generation of synthetic model test data, but is unknown to the model in subsequent testing. Moreover, we show that inversion and calibration with experimental data using our framework produces physically sensible optimum parameters that always lead to better agreement between our model results and the data. In particular, we demonstrate how our adjoint framework can be applied to a tsunami-like event to invert for the tsunami wave's characteristics from experimental sediment deposit data.

Publications and Disclaimer

The work in this chapter has been submitted for publication to *Computers & Geosciences*, where it is currently under review. The preprint of this paper is available here:

Clare, M. C., Kramer, S. C., Cotter, C. J., & Piggott, M. D. (2021). 'Calibration, inversion and sensitivity analysis for hydro-morphodynamic models through the application of adjoint methods.' https://eartharxiv.org/repository/view/2599/

3.1 Introduction

Hydro-morphodynamic models can be affected by a high degree of uncertainty due to incomplete knowledge and natural variability of various physical, empirical and numerical closure related parameters in both the hydrodynamic and morphodynamic solvers. Research on methods to assess this uncertainty is ongoing and includes both numerical and statistical techniques. In general, numerical techniques solve a set of equations using a numerical approximation leading to a deterministic solution, whereas statistical techniques analyse and summarise output data leading to a stochastic solution. In this chapter, we explore numerical techniques, which are less computationally expensive but harder to implement than statistical techniques which we discuss in Chapters 5 and 6.

One type of numerical technique to assess uncertainty is the tangent linear method. This is computationally cheaper than a statistical method but still costly, especially because a separate tangent linear run is required for each uncertain parameter. Naumann and Riehme (2008) present an implementation of this method with the *Sisyphe* model (described in Section 2.3.1) which has been applied to morphodynamic test cases in Riehme et al. (2010), Kopmann et al. (2012), Hieu et al. (2015), Villaret et al. (2016) and Dalledonne et al. (2017). Note that in these works, for ease of implementation, the tangent linear method is not applied to the fully coupled model, but only to the morphodynamic component (consisting of the sediment concentration equation and the Exner equation). Hereafter, we refer to the morphodynamic component as *Sisyphe-morph*, to make a clear distinction from the fully coupled hydro-morphodynamic model *Sisyphe*.

In this chapter, we focus on the adjoint method, a numerical technique related to the tangent linear method, and one that can be used to assess model uncertainty to great advantage. As discussed in Chapter 1, the adjoint method computes gradients of model outputs with respect to input parameters and, thus, is useful for calibration, inversion and sensitivity analysis of uncertain parameter values. Furthermore, only one adjoint evaluation is required to compute this gradient, irrespective of the number or dimension of the uncertain parameters and, thus, we can consider any number of multi-dimensional uncertain parameters with almost no effect on the computational cost (Heemink et al., 2002; Chen et al., 2014; Funke et al., 2017). The adjoint method has not previously been applied to a fully coupled 2D hydro-morphodynamic model, but only to simpler uncoupled models, including a 1D hydro-morphodynamic model for turbidity currents in Parkinson et al. (2017), and *Sisyphe-morph* in Kopmann et al. (2012); Merkel et al. (2013, 2016). However, only three *Sisyphe-morph* test cases using the adjoint method have been published, and recent research appears to be limited. Like the tangent linear method, the adjoint method has never been applied to the fully coupled hydro-morphodynamic model *Sisyphe*, to the best of our knowledge. This is significant because many parameters influence both the hydrodynamic and morphodynamic components and there are many feedback effects between them. Thus, applying the adjoint or the tangent linear method only on the morphodynamic component reduces accuracy and limits the cases where these methods can be of value.

A difficulty of using either the adjoint or the tangent linear approach is that their implementation requires the derivation of new equations. This can be done manually, a time consuming and error-prone process (see Naumann and Riehme, 2008), or by using a library which automatically derives them. Both tangent linear and adjoint methods are implemented within *Sisyphe-morph* using the commercial NAG FORTRAN compiler (see Naumann and Riehme, 2005; Merkel et al., 2016), which is financially expensive. Moreover this compiler is also computationally expensive, for example, Merkel et al. (2016) report that an adjoint run is 135 times more expensive than a standard model run.

We have therefore described a definitive need for the fully flexible, free-to-use, and relatively computationally efficient adjoint framework that we present in this chapter. We use our *Thetis* model presented in Section 2.2 for the hydro-morphodynamic model, coupled with the free-to-use *pyadjoint* library (Farrell et al., 2013). This library works with the code-generation framework *Firedrake* (Rathgeber et al., 2017) to automatically derive adjoint equations using the high level abstraction of the finite element equations available within all *Firedrake* based models. Using *pyadjoint* gives our framework the flexibility to assess how uncertainty in any parameter affects any output functional without further code implementation. We demonstrate this capability by applying the adjoint method to a range of test cases, all of which have been validated for our model in Chapter 2. For all test cases considered, an adjoint run with our framework takes at most three times more than a forward model run. This is two orders of magnitude faster than the similar state-of-the-art application of the adjoint method using the commercial NAG FORTRAN compiler described above (Merkel et al., 2016). Although *pyadjoint* has previously been used with the hydrodynamic component of *Thetis*, for example in Warder et al. (2021), our work is the first time it is used in a coupled model.

The remainder of this chapter is structured as follows: in Section 3.2 we describe the adjoint method and its implementation in our hydro-morphodynamic model; in Section 3.3, Section 3.4 and Section 3.5 we apply our adjoint hydro-morphodynamic model framework to different test cases for sensitivity analysis, calibration of uncertain parameters and inversion of tsunami-like waves, respectively; and, finally, in Section 3.6 we present concluding remarks.

3.2 Adjoint method

The adjoint method computes the gradient of a model output functional with respect to a set of parameters and is thus useful for assessing uncertainty (Farrell et al., 2013). We first briefly present a derivation of this method, following Funke (2012), and establish notation.

Here, our hydro-morphodynamic model (presented in Section 2.2) is our forward model and can be written in the abstract form $F(u(\mathbf{m}), \mathbf{m}) = 0$ where $\mathbf{m}$ is a set of uncertain parameters and u is the model solution. Any given output functional, J, depends only on $\mathbf{m}$ and u and thus the total derivative is

$$\frac{dJ}{d\mathbf{m}} = \frac{\partial J}{\partial u}\frac{du}{d\mathbf{m}} + \frac{\partial J}{\partial \mathbf{m}}. \tag{3.1}$$

Furthermore, differentiating the forward model with respect to $\mathbf{m}$ and rearranging yields

$$\frac{du}{d\mathbf{m}} = -\left(\frac{\partial F}{\partial u}\right)^{-1} \frac{\partial F}{\partial \mathbf{m}}, \tag{3.2}$$

and therefore

$$\frac{dJ}{d\mathbf{m}} = -\frac{\partial J}{\partial u}\left(\frac{\partial F}{\partial u}\right)^{-1} \frac{\partial F}{\partial \mathbf{m}} + \frac{\partial J}{\partial \mathbf{m}}, \tag{3.3}$$

which can be evaluated using either the adjoint or the tangent linear approach.

If the adjoint method is used, we evaluate λ in

$$\left(\frac{\partial F}{\partial u}\right)^{T} \lambda^{T} = \left(\frac{\partial J}{\partial u}\right)^{T}, \tag{3.4}$$

$\left(\text{equivalent to } \lambda = \frac{\partial J}{\partial u}\left(\frac{\partial F}{\partial u}\right)^{-1}\right)$ and substitute it into

$$\frac{dJ}{d\mathbf{m}} = -\lambda \frac{\partial F}{\partial \mathbf{m}} + \frac{\partial J}{\partial \mathbf{m}}, \tag{3.5}$$

to find the derivative. Note that only one linear solve is ever necessary to evaluate (3.4), independent of the number of uncertain parameters or their dimension.

If the tangent linear method is used, we evaluate μ_t in

$$\left(\frac{\partial F}{\partial u}\right) \mu_\mathrm{t} = -\frac{\partial F}{\partial \mathbf{m}}. \tag{3.6}$$

$\left(\text{equivalent to } \mu_\mathrm{t} = -\left(\frac{\partial F}{\partial u}\right)^{-1} \frac{\partial F}{\partial \mathbf{m}}\right)$ and substitute it into

$$\frac{dJ}{d\mathbf{m}} = \frac{\partial J}{\partial u}\mu_\mathrm{t} + \frac{\partial J}{\partial \mathbf{m}}, \tag{3.7}$$

to find the derivative. Note that the latter is equivalent to (3.1), meaning that the tangent linear method is solving for $du/d\mathbf{m}$. The expression for μ_t can be solved using a variety of different methods including explicitly inverting the linearised PDE operator $\partial F/\partial u$ or performing an iterative solve for each dimension of $\mathbf{m}$ (see Funke, 2012). For all these methods, the computational cost of solving this expression scales linearly with the dimension of $\mathbf{m}$. By contrast, once μ_t is computed, the derivative of the output functional J can be efficiently computed for multiple different outputs. This makes the tangent linear approach well-suited to problems with a relatively small number of uncertain one-dimensional input parameters but a large number of uncertain outputs and/or an output functional which is a spatially-varying field. Conversely, the adjoint solution is ideally suited to problems where there are multiple uncertain input parameters (both scalar and multi-dimensional), which is the case for hydro-morphodynamic problems.

As discussed in Section 3.1, previous works using hydro-morphodynamic models have mainly focused on the tangent linear method because, although it can be computationally expensive, it is relatively easy to implement. We overcome this problem by using our model presented in Section 2.2, which is built within *Thetis*, a *Firedrake*-based model. This means we can use the *pyadjoint* library (Farrell et al., 2013), specifically constructed to work within the *Firedrake* environment. This library automatically derives the

adjoint equations by first 'taping' a forward model run (recording the sequence of numerical operations) and then using this tape to construct the discretised adjoint equations. This means that the actual derivative of the discrete model is used (up to numerical truncation errors and solver tolerances), rather than a discrete approximation of a continuous derivative (see Funke et al., 2017). A similar methodology can also be used to implement the tangent linear method using *pyadjoint*.

The adjoint method has already been used successfully with the hydrodynamic component of *Thetis* (see for example Warder et al., 2021), but we expand upon this here by using it with a coupled model which requires extending the *pyadjoint* code to ensure that the coupling is correctly captured. In particular, the implementation of the coupling between the components of the hydro-morphodynamic model relies on a 'split' mechanism in Firedrake, which extracts the velocity and elevation from the hydrodynamic component so that both can be passed to the morphodynamic component. The new *pyadjoint* code tapes and calculates the adjoint of this operation, thus facilitating the use of *pyadjoint* for all *Firedrake*-based coupled models (full details of the code change are given in Firedrake Project (2021)). The adjoint equations are not shown in the main body of the text because using *pyadjoint* means we do not have to derive them manually. However, for completeness, we have included a derivation of the continuous adjoint equations in Appendix A.

In order to verify the adjoint implementation derived by *pyadjoint* and, in particular, the new adjoint implementation of our coupled model, we use the Taylor remainder convergence test (Funke, 2012). This test applies a first-order Taylor expansion to the functional J and checks whether the gradient $\frac{dJ}{d\mathbf{m}}$ derived using the adjoint solution is correct by verifying that the Taylor residual converges at second-order

$$\left| J(\mathbf{m} + h\delta\mathbf{m}) - J(\mathbf{m}) - h\delta\mathbf{m}\frac{dJ}{d\mathbf{m}} \right| = \mathcal{O}(h^2) \qquad \text{as } h \to 0, \tag{3.8}$$

where $\delta\mathbf{m}$ is a random perturbation. This second-order convergence is very sensitive to implementation errors (see Funke, 2012) and thus represents a strict code verification check. All test cases outlined in this chapter pass this test, confirming that the adjoint implementation of our coupled hydro-morphodynamic model is correct.

3.2.1 Using the adjoint method to address uncertainty

As discussed in Section 3.1, the adjoint method can be used to address the uncertainty of models with respect to particular parameters through sensitivity analysis, calibration and inversion.

To analyse the sensitivity of model outputs to particular uncertain parameters, we use the adjoint method to compute the derivative $dJ/d\mathbf{m}$. The output functional, J, can take many forms and for sensitivity analysis, we choose

$$J(\mathbf{u}^{\text{model}}, \mathbf{m}) = \frac{1}{2} \sum_{j=1}^{N_{\text{out}}} \int_0^T \int_\Omega \sqrt{|u_j^{\text{model}}|^2 + \epsilon}\, dx\, dt, \tag{3.9}$$

which analyses the sensitivity of an integrated output, (for example, the integrated final bedlevel) to uncertain parameters. Here, Ω is the domain of the model, N_{out} the number of output variables used and ϵ a parameter set to 10^{-6} which we add to smooth our results if u_j^{model} is equal to zero anywhere in the domain.

In contrast, for inversion, we minimise the following problem

$$\begin{aligned} &\underset{u,\,\mathbf{m}}{\text{minimize}} && J(u,\mathbf{m}) \\ &\text{subject to} && F(u,\mathbf{m}) = 0, \end{aligned} \tag{3.10}$$

using the default L-BFGS-B algorithm available via the SciPy library (Virtanen et al., 2020). We then choose an output functional, J, of the following form

$$J(\mathbf{u}^{\text{model}},\,\mathbf{m}) = \alpha \sum_{j=1}^{N_{\text{out}}} \int_0^T \int_\Omega |u_j^{\text{true}} - u_j^{\text{model}}|^2 \, dx\, dt + \sum_{i=1}^{N_{\text{in}}} \beta_i \int_0^T \int_\Omega |m_i|^2 \, dx\, dt, \tag{3.11}$$

where N_{in} is the number of uncertain parameters and α and β_i are user-specified scaling factors. The first integral term in (3.11) is the difference between the model output and the true value, which can be either experimental/real-world data or, in the case of a dual twin experiment, synthetic data generated using a previous run of the model with known parameter values. In contrast to (3.9), here we do not take the square root of this term, in order to more severely penalise large differences. The second integral term is a Tikhonov regularisation term, which aids in the solution of ill-posed problems and can be used to prevent the magnitude of the parameters becoming unphysical (see Engl et al., 1996). The amount of regularisation is controlled by β_i and can be defined specific to each uncertain parameter.

Finally, in this chapter we choose to perform calibration by inverting our model for the uncertain parameters and using a minimisation algorithm to minimise the error between the model output and the desired output *i.e.* the same methodolody as for inversion. This is because calibration can be seen as a sub-type of inversion, where we have a better prior knowledge of the approximate values of the uncertain parameters than in standard inversion problems and therefore may choose stricter bounds on the penalty terms.

3.3 Sensitivity analysis for a Meander

We can now apply our adjoint hydro-morphodynamic model framework to a series of test cases. For our initial test case, we consider flow around the curved channel of a meander using the configuration from experiment 4 of Yen and Lee (1995) and the same set-up and parameter values as in Section 2.4.2 of this thesis where we validated this test case. Note that in all meander figures, the flow is from the bottom left to the bottom right.

3.3.1 Tangent linear approach to sensitivity analysis

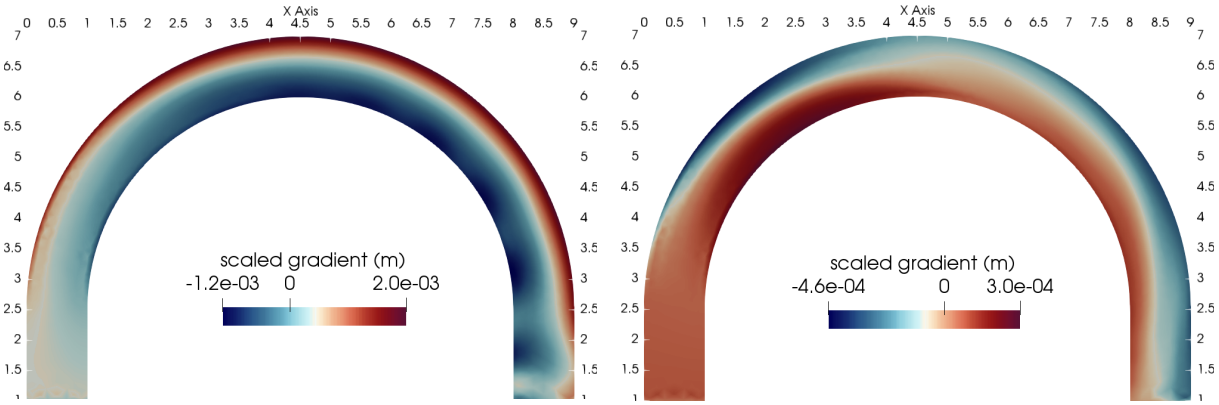
This meander test case is also studied in Riehme et al. (2010), where the sensitivity of the bedlevel evolution to uncertain parameters is analysed using the model in Telemac-Mascaret. They use the First Order Reliability Method (FORM) (a variation of the tangent linear method) to calculate the quantity σJ_m, which they refer to as the standard deviation, but we refer to as the scaled gradient to avoid confusion with the standard deviation of the parameter. The scaled gradient is defined as

$$\sigma J_m = \sigma_m \frac{dJ}{dm}\bigg|_{m=\langle m \rangle}, \tag{3.12}$$

where J is the bedlevel evolution, a spatially-varying field (*i.e.* $J = z_b^{\text{final}} - z_b^{\text{initial}}$), and $\langle m \rangle$ and σ_m represent the mean and standard deviation of the uncertain parameter, respectively, which must be estimated. Note that in Riehme et al. (2010), the standard deviation σ_m is assumed constant and thus multiplying by σ_m merely scales the sensitivity without altering its spatial pattern. The derivative in (3.12) is then calculated using the tangent linear method described in Section 3.2.1, which is suitable for this problem because although the bedlevel evolution is multi-dimensional, Riehme et al. (2010) only consider one uncertain scalar valued parameter at a time. Note, however, that throughout Riehme et al. (2010), the tangent linear method is only applied to the morphodynamic component of the model *Sisyphe-morph*, with Telemac2D (the hydrodynamic component) only used to update the hydrodynamics at periodic intervals.

As a first verification step of our adjoint hydro-morphodynamic model framework, we compare the FORM analysis (3.12) using our model with that from using *Sisyphe-morph* for the two scalar parameters of average sediment grain size, d_{50}, and bed reference height, k_s, which are both key to determining the sediment transport rate. Uncertainty quantification for these parameters is important because both are challenging to determine, particularly d_{50} which is difficult to measure in offshore environments and may change seasonally (see Jaffe et al., 2016). In addition, k_s determines the bed friction, to which hydrodynamic models are highly sensitive (see for example Merkel et al., 2013; Warder et al., 2021). Following Riehme et al. (2010), we assume a mean of 1×10^{-3} m for d_{50} and 3×10^{-3} m for k_s, and a standard deviation of 1×10^{-4} m for both.

Figure 3.1 shows the scaled gradient σJ_m of the bedlevel evolution for the scalar parameters d_{50} and k_s computed by applying the tangent linear method to our full model. Figure 3.2 is taken from Riehme et al. (2010) and shows the equivalent results when the tangent linear method is applied to *Sisyphe-morph*. Comparing the two figures reveals that the spatial pattern of the scaled gradient is similar for d_{50}, but slightly different for k_s. We hypothesise that one of the reasons for the difference in pattern for k_s is that, as already mentioned, the tangent linear method is only applied to the morphodynamic component of the model with *Sisyphe-morph*. In contrast, our adjoint hydro-morphodynamic model framework computes the gradient for the fully coupled hydro-morphodynamic model, hence producing more accurate results. This is particularly important in the case of k_s because this parameter is key in determining both the



(A) Average sediment grain size, d_{50}.

(B) Bed reference height, k_s.

FIGURE 3.1. Scaled gradient of bedlevel evolution (3.12) for the Meander test case obtained using the tangent linear method applied to the full hydro-morphodynamic model in our framework.

sediment transport rate in the morphodynamic component and the bed friction in the hydrodynamic component.

To further investigate the differences between the *Thetis*-pyadjoint and *Sisyphe-morph* results, we re-run the meander test case, but this time only apply the tangent linear method to the morphodynamic component of our adjoint hydro-morphodynamic model framework. Figure 3.3b shows that this results in the *Thetis*-pyadjoint spatial pattern for k_s agreeing much more closely with that from *Sisyphe-morph*. Hence, this helps confirm that the differences between Figure 3.1b and the right hand side of Figure 3.2 are largely due to the tangent linear method being applied to the full model rather than just the morphodynamic component. For d_{50}, applying the tangent linear method to the full model (Figure 3.1a) or just the morphodynamic component (Figure 3.3a) does not have a large impact on the results because the hydrodynamics do not explicitly depend on d_{50}. Nevertheless, we note that the scaled gradient is smoother at the outflow in Figure 3.3a (morphodynamic component only), which is more consistent with the *Sisyphe-morph* results.

We must also compare the magnitude of the scaled gradient in Figures 3.2 and 3.3 to establish agreement between the two models. The figures show that for both d_{50} and k_s, the magnitude of the scaled gradient determined by our model is similar to that determined by *Sisyphe-morph*, but at the inner bend, *Sisyphe-morph* predicts a greater magnitude. Small magnitude differences such as these are to be expected because the models are constructed slightly differently and the *Thetis* final bedlevel results are more accurate even when the full hydro-morphodynamic model version of Telemac-Mascaret (*Sisyphe*) is run (see Figure 2.19 in Chapter 2).

In summary, the spatial patterns of the scaled gradients of our adjoint hydro-morphodynamic model framework agree well with those from *Sisyphe-morph*. Moreover, the magnitudes of the scaled gradients also agree fairly closely, with slight differences due to differences in model set-up. Therefore, these

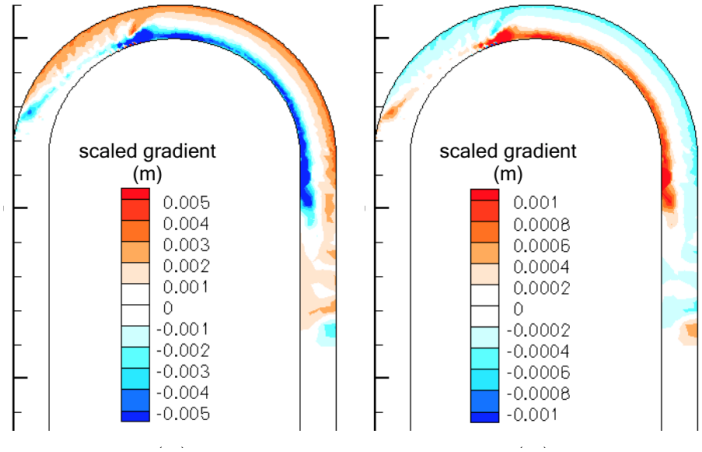
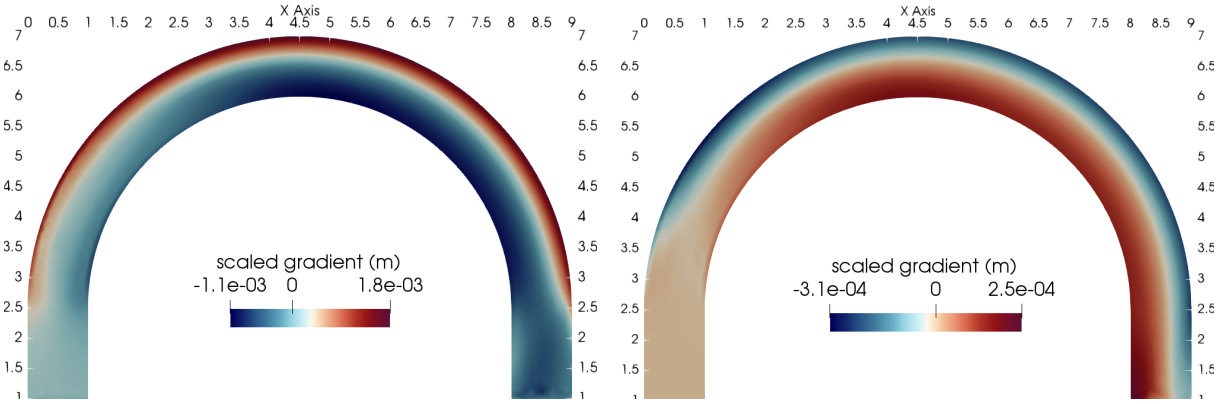


FIGURE 3.2. Scaled gradient of bedlevel evolution (3.12) for the Meander test case obtained using the tangent linear method applied to *Sisyphe-morph*. Left: d_{50}; Right: k_s. Source: Riehme et al. (2010).



(A) Average sediment grain size, d_{50}.

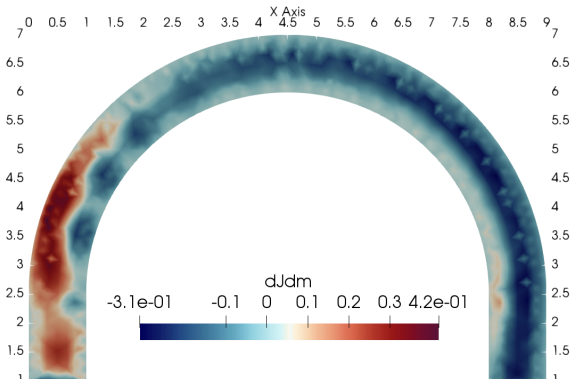
(B) Bed reference height, k_s.

FIGURE 3.3. Scaled gradient of bedlevel evolution (3.12) for the Meander test case obtained using the tangent linear method **only** on the morphological component of our hydro-morphodynamic model.

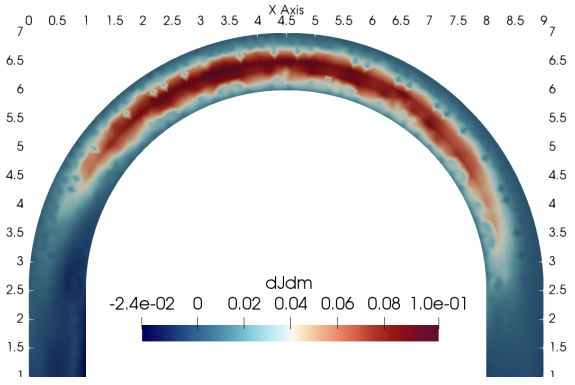
results provide confidence in our framework implementation and indicate we can use it to analyse further problems.

3.3.2 Adjoint approach to sensitivity analysis

The tangent linear analysis above calculates the sensitivity of the bedlevel everywhere to a single scalar parameter m. In contrast, a key advantage of using the adjoint approach is that we can determine the bedlevel sensitivity to more than one scalar and/or spatially-varying parameters using a single run. In this section, we choose the latter option and analyse the sensitivity of the meander bed to the now spatially-varying parameters d_{50} and k_s, using (3.9) as the output functional where $\mathbf{u}^{\text{model}}$ is the final bedlevel.



(A) Average sediment grain size, d_{50}.



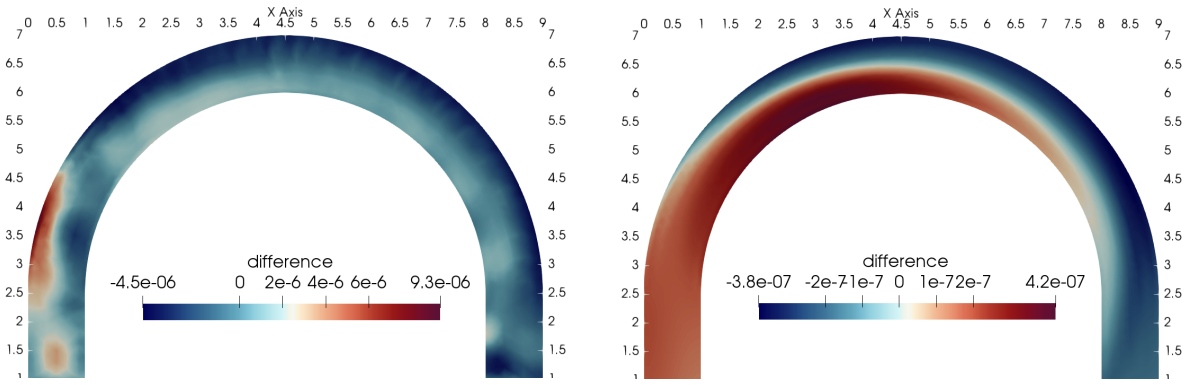
(B) Bed reference height, k_s.

FIGURE 3.4. Sensitivity of integrated bedlevel for the Meander test case applying the adjoint method using our framework. A positive value indicates increasing the uncertain parameter at this location increases the bedlevel change and vice versa for a negative value.

Figures 3.4a and 3.4b show the sensitivity of the bedlevel to the spatially-varying parameters d_{50} and k_s, respectively, where we have evaluated the derivative of the output functional (3.9) at $d_{50} = 1 \times 10^{-3}\,\mathrm{m}$ everywhere and at $k_s = 3 \times 10^{-3}\,\mathrm{m}$ everywhere, since these are the values used in the original test case (see Villaret et al. (2013) and Section 2.4.2). Given (3.9) and that the meander bedlevel is centered around zero, in both figures, positive sensitivity means perturbing the uncertain parameter here causes more overall bed movement compared to the unperturbed final bedlevel, and negative sensitivity means less overall bed movement. The figures show that the most positive sensitivity region for k_s is at the centre of the channel, whereas for d_{50} it is on the left, at the flow input.

To better understand the computed sensitivities, we perturb both uncertain spatially-varying parameters from their original value in the direction of the derivative by adding their respective gradient fields (depicted in Figures 3.4a and 3.4b) multiplied by 10^{-6}. We then calculate the difference between the original and perturbed final bedlevels as spatially varying fields. Note that a positive value indicates that perturbing the uncertain parameter results in more deposition and equivalently a negative value indicates more erosion. Figure 3.5a shows that perturbing d_{50} causes increased deposition at the outer bend of the inflow. This can be explained physically: from Figure 3.4a, the perturbation results in a larger sediment grain size at the inflow, which only the faster velocity at the outer bend can erode. This sediment also gets deposited quickly because of its mass, meaning the overall effect is increased deposition in this area. In the rest of the domain, the perturbation in d_{50} accentuates the sediment transport patterns already present in the original final bedlevel (see Figure 2.16 in Chapter 2), which is a physically sensible result. Similarly, Figure 3.5b shows that perturbing k_s causes increased deposition at the inflow and increased erosion at the outflow and within this trend, it causes more deposition at the inner bend and more erosion at the outer bend. Physically, decreasing k_s has two contrasting effects: decreasing C_h in τ_b (2.37) thus decreasing the fluid's ability to erode but also increasing the velocity term in τ_b thereby increasing the fluid's ability to erode. At the inflow, we see that the decreasing friction leads to less erosion and more deposition as C_h dominates. However, as the velocity increases around the bend, the velocity term starts

to dominate leading to increased erosion. This also accounts for the different behaviour at the inner and outer parts of the bend.



(A) Average sediment grain size, d_{50}.

(B) Bed reference height, k_s.

FIGURE 3.5. Spatially-varying difference between the original and the perturbed final bedlevels simulated using our *Thetis* model for the Meander test case. The perturbed bedlevel is obtained by perturbing the uncertain spatially-varying parameters from their original value in the direction of their derivative (calculated using the adjoint method). Note that a positive value indicates more deposition and a negative value more erosion.

Thus, the results in this section demonstrate that our adjoint hydro-morphodynamic model framework can accurately analyse the sensitivity of our model to uncertain spatially-varying parameters for a complex test case.

3.4 Optimum parameter calibration for a migrating Trench

The adjoint method can be used not only for sensitivity analysis but also to calibrate for uncertain parameters. We illustrate this by considering a migrating trench for which experimental data exists in Van Rijn (1980) and which has already been validated for our model in Section 2.4.1. This test case is very sensitive to input parameters, in particular to diffusivity (see Figure 2.9 in Section 2.4.1), which makes it an ideal test case for the adjoint method. Throughout this section, unless otherwise stated, we use the set-up from Section 2.4.1 with a mesh of $\Delta x = 0.25\,\text{m}$ and $\Delta y = 0.2\,\text{m}$ and a morphological acceleration factor of 100.

3.4.1 Dual twin experiment

We first conduct a dual-twin experiment, where the 'true' output is generated by a previous model run. Thus, we know the value of the 'uncertain' parameter and can verify that our framework can reconstruct it. Given we are not trying to match with experimental data and for reasons of time, the simulation is only run for 5 h instead of the full experimental time of 15 h.

To be consistent between the dual twin experiment and the calibration in Section 3.4.2, we assume here that the z_b^{true} profile is only known at certain locations, as is the case with real-world data. Following

Saito et al. (2011), we extract the bedlevel at these locations by multiplying the model bedlevel and the 'true' z_b by a Gaussian function centred at the experimental data locations, x_i

$$\mathring{z}_b(x; x_i) = z_b \times \left(\exp^{-50(x-x_i)^2}\right). \tag{3.13}$$

Here the exponent is scaled by 50 to ensure the base of the Gaussian function is narrow around x_i and we use our knowledge of the test case to assume no variation in the $y-$direction. Thus, the general functional (3.11) from Section 3.2.1 becomes

$$J(z_b, \mathbf{m}) = \frac{1}{2}\sum_{i=1}^{k} \frac{\int_\Omega |\mathring{z}_b^{\text{model}}(x; x_i) - \mathring{z}_b^{\text{true}}(x; x_i)|^2 \, dx}{\int_\Omega |\exp^{-50(x-x_i)^2}|^2 \, dx}, \tag{3.14}$$

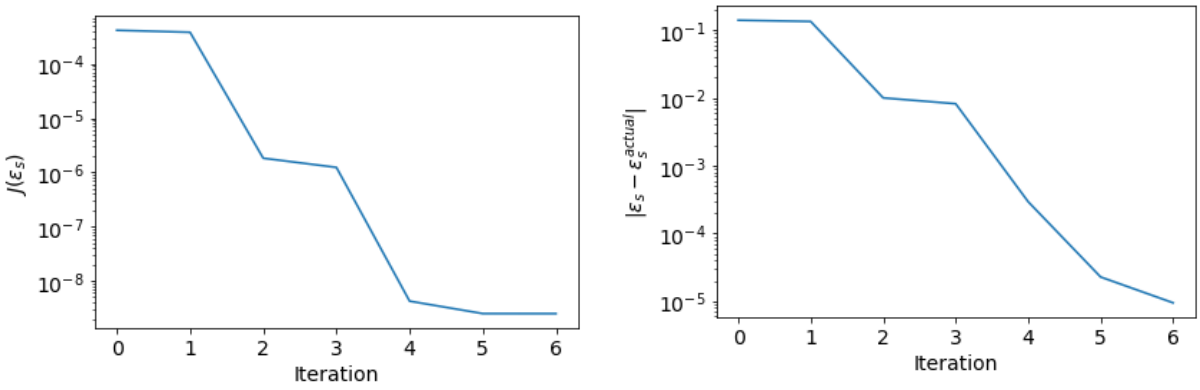
where k is the number of experimental data points and the Gaussian function has been normalised. The integral over time in (3.11) is unnecessary here because the experimental data only exists at one point in time.

3.4.1.1 Finding reference diffusivity coefficient value

For our first test, we consider the uncertain input parameter to be ϵ_s because our results in Section 2.4.1 show that this test case is very sensitive to this parameter. We use $\epsilon_s = 0.01\,\mathrm{m^2\,s^{-1}}$ to generate the 'true' output because this is the original value used in Villaret et al. (2016), and use $\epsilon_s = 0.15\,\mathrm{m^2\,s^{-1}}$ as the initial guess to start the minimisation algorithm. Figure 3.6a shows the functional decreases with each iteration, eventually flattening out, thus causing the minimisation algorithm to stop. Figure 3.6b shows that the error between the simulated value and the reference value also decreases with the number of iterations, showing that the minimisation of the output functional works as expected. At the final iteration, the error between the simulated diffusivity value and the actual value is $9.47 \times 10^{-6}\,\mathrm{m^2\,s^{-1}}$. Realistically, this error is already much smaller than any model or discretisation error, but, if required, an even smaller error could be achieved by further reducing the tolerance in the minimisation algorithm.

3.4.1.2 Finding multiple reference parameters

One of the advantages of the adjoint method is that the number of uncertain parameters has almost no effect on the computational cost of using it to calculate the gradient (see Section 3.2). Therefore, in this section we choose to reconstruct multiple uncertain reference parameters simultaneously and consider them to be spatially-constant. For our uncertain parameters, we choose d_{50} and k_s, because they are both key in determining sediment transport rate (see Section 3.3); the sediment density ρ_s, because it follows that if sediment size is uncertain then sediment density may also be uncertain; and the diffusivity coefficient ϵ_s, as in the previous section. To generate the 'true' output, we use the values $\rho_s = 2000\,\mathrm{kg\,m^{-3}}$, $d_{50} = 2 \times 10^{-4}\,\mathrm{m}$, $k_s = 0.01\,\mathrm{m}$ and $\epsilon_s = 0.01\,\mathrm{m^2\,s^{-1}}$ and then use $\rho_s = 2650\,\mathrm{kg\,m^{-3}}$, $d_{50} = 1.6 \times 10^{-4}\,\mathrm{m}$, $k_s = 0.025\,\mathrm{m}$ and $\epsilon_s = 0.15\,\mathrm{m^2\,s^{-1}}$ as the initial guesses to start the minimisation algorithm. Note that these four parameters have different orders of magnitude and therefore must be scaled to ensure the minimisation algorithm works efficiently. We choose to multiply each of them by



(A) Minimisation of output functional.

(B) Error convergence between the model and reference parameters.

FIGURE 3.6. Using the adjoint method with a minimisation algorithm to find the reference diffusivity coefficient ϵ_s for the Trench test case.

one over their order of magnitude so that the scaled parameters all have an order of magnitude of 1. Naturally, within the forward model, these are then re-scaled to ensure physically correct results.

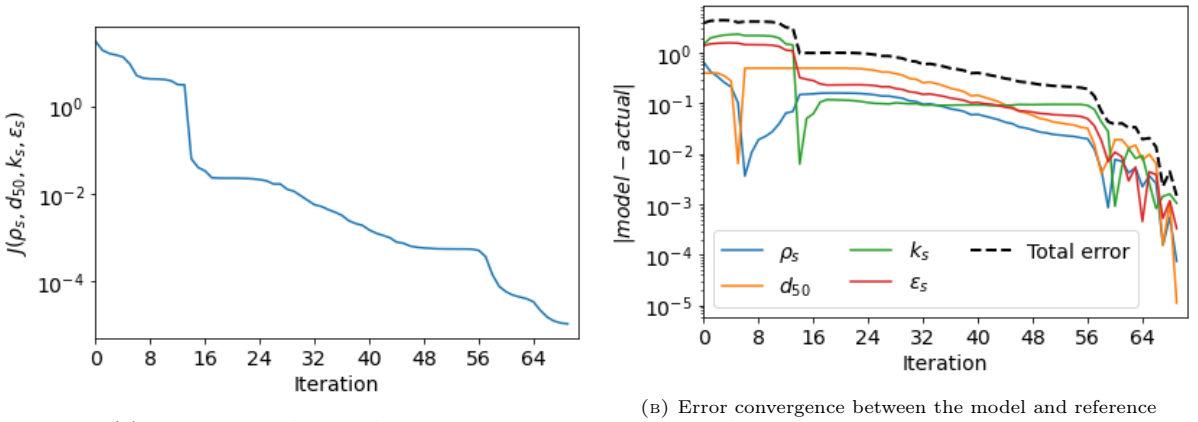
Figure 3.7a shows that the output functional (3.14) decreases at each iteration of the minimisation algorithm (3.10), which results in a general error reduction in the scaled ρ_s, d_{50}, k_s and ϵ_s as shown in Figure 3.7b. However, this reduction is not always smooth, as is common in similar non-linear multi-parameter problems. A better metric to look at is the total error (*i.e.* the sum of the errors from the four individual parameters) which Figure 3.7b shows does decrease more uniformly. Furthermore, at the final iteration, the error between the reconstructed and actual scaled values has an order of magnitude of 10^{-3} or better for all four parameters, corresponding to an unscaled order of magnitude error of approximately $1 \times 10^{-2}\,\mathrm{kg\,m^{-3}}$ for ρ_s, $1 \times 10^{-9}\,\mathrm{m}$ for d_{50}, $1 \times 10^{-5}\,\mathrm{m}$ for k_s and $1 \times 10^{-5}\,\mathrm{m^2 s^{-1}}$ for ϵ_s. Again, a smaller error could be achieved by reducing the tolerance in the minimisation algorithm, but realistically this error is already much smaller than mesh or model error. Thus, this test case demonstrates that the adjoint method can be used to calibrate for multiple spatially-constant parameters in our model simultaneously.

3.4.2 Calibration of parameters for a laboratory test case

Following the verification of our adjoint framework, we can now apply the same method to perform uncertain parameter calibration with experimental data. Using the same migrating trench test case set-up, we now run the simulation for the full experimental time (15 h) and the true values in (3.14) are the experimental data from Van Rijn (1980).

3.4.2.1 Finding optimum diffusivity coefficient, ϵ_s

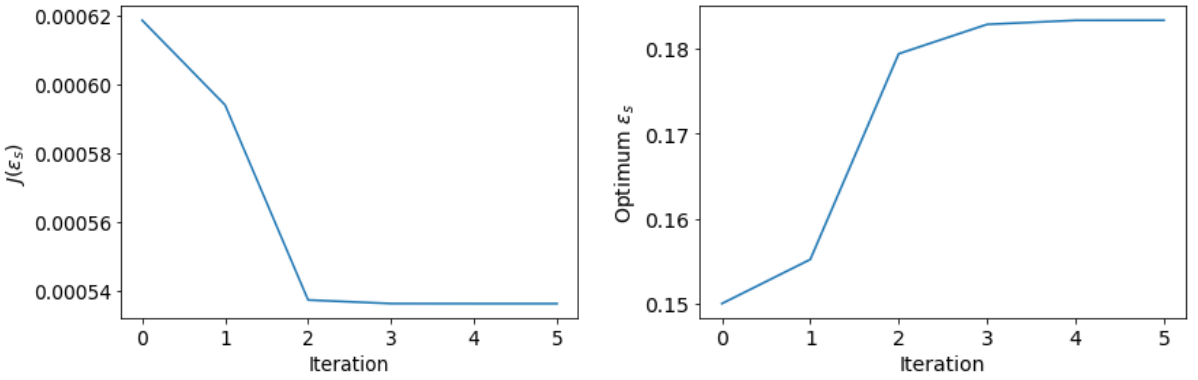
In Section 2.4.1, we used a simple sensitivity study to estimate that the optimum value of ϵ_s for this trench test case is $0.15\,\mathrm{m^2\,s^{-1}}$. Therefore, as a first calibration test, we consider ϵ_s to be uncertain and calibrate for it using this value as the initial guess and (3.14) as the output functional with the true values being the experimental data given in Van Rijn (1980). Figure 3.8a shows the functional decreases



(A) Minimisation of output functional.

(B) Error convergence between the model and reference parameters for both the individual errors and the total error (*i.e.* the sum of the errors from the four individual parameters).

FIGURE 3.7. Using the adjoint method with a minimisation algorithm to find the reference sediment parameters d_{50}, ρ_s, k_s and ϵ_s for the Trench test case.



(A) Minimisation of output functional.

(B) Convergence to optimum ϵ_s.

FIGURE 3.8. Using the adjoint method with a minimisation algorithm to find the optimum diffusivity coefficient, ϵ_s, for the Trench test case.

with each iteration and Figure 3.8b shows this leads to the convergence of ϵ_s to an optimum value of $0.183\,\mathrm{m^2\,s^{-1}}$, a physically reasonable value.

In order to test the robustness of this optimum value for ϵ_s, we use it in the hydro-morphodynamic model *Sisyphe* (see Section 2.3.1) that has previously been used to simulate this test case in Villaret et al. (2016) and Section 2.4.1. To reiterate, as this is just a simple forward model run, we can use *Sisyphe*, the full coupled hydro-morphodynamic version of Telemac-Mascaret. Figure 3.9 compares the final bedlevel from the *Thetis* and *Sisyphe* models using the optimum ϵ_s and using the original value from Villaret et al. (2016) ($\epsilon_s = 0.01\,\mathrm{m^2\,s^{-1}}$). It clearly shows that, in both *Thetis* and *Sisyphe*, the optimum ϵ_s value improves the accuracy of the final bedlevel, thus suggesting that our optimum diffusivity coefficient is not model dependent and giving further confidence in our adjoint framework.

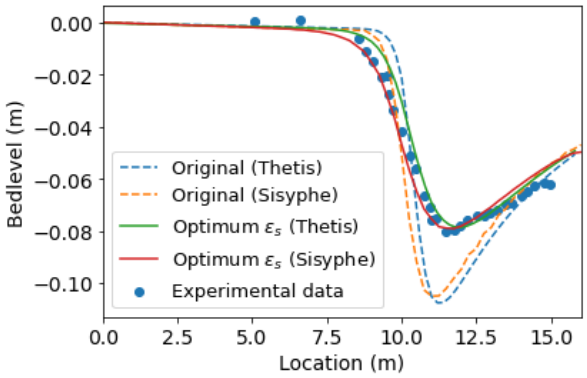


FIGURE 3.9. Comparing the final bedlevel from both the hydro-morphodynamic models *Thetis* and *Sisyphe* when using optimum ϵ_s compared to the original $\epsilon_s = 0.01\,\mathrm{m^2\,s^{-1}}$ used in Villaret et al. (2016) for the Trench test case.

3.4.2.2 Finding multiple optimum parameters

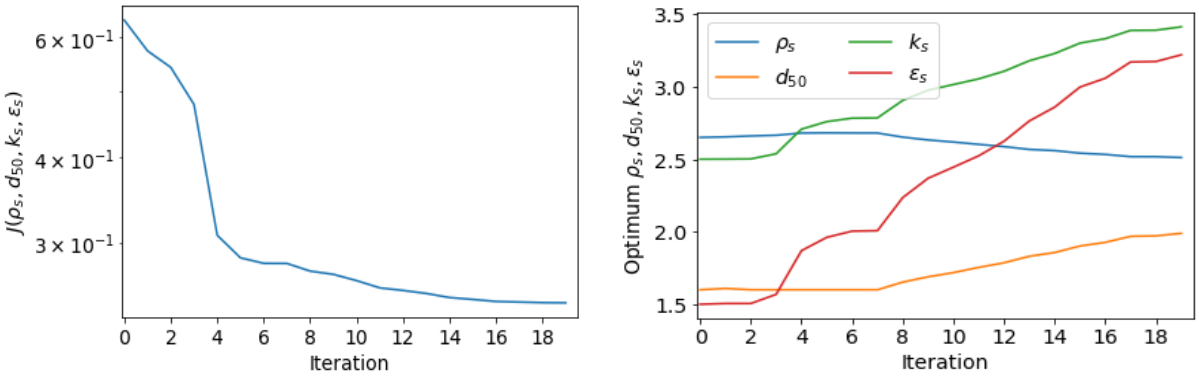
Previously, in Section 3.4.1.2, we showed that the adjoint method can be used to reconstruct the uncertain parameters ρ_s, d_{50}, k_s and ϵ_s in the same simulation. In this section, we seek the optimum values of all four of these parameters using the experimental data from Van Rijn (1984) as the true values. To ensure the optimum values obtained are physical and do not blow up during the minimisation, we add a Tikhonov regularisation term to (3.14) to obtain

$$J(z_b, \mathbf{m}) = \alpha \sum_{i=1}^{k} \frac{\int_\Omega |\mathring{z}_b^{\mathrm{model}}(x; x_i) - \mathring{z}_b^{\mathrm{true}}(x; x_i)|^2\, dx}{\int_\Omega |\exp^{-50(x-x_i)^2}|^2\, dx} + \sum_{i=1}^{N_{\mathrm{in}}} \beta_i \int_\Omega |m_i|^2\, dx, \tag{3.15}$$

and set α equal to 1000, and β_i equal to 10^{-4} for ρ_s, d_{50} and k_s, and 5×10^{-5} for ϵ_s, reflecting that ϵ_s has a large impact on the final result (as discussed in Section 2.4.1). The initial guesses for the minimisation algorithm are the parameter values for this test case in Section 2.4.1, rescaled as before so that they all have an approximate order of magnitude of 1.

Figure 3.10a shows that the functional value decreases with each iteration and Figure 3.10b shows that this leads to the convergence of all four parameters to optimum values. The parameters shown in the figure are the scaled ones and therefore the actual optimum values are $2511\,\mathrm{kg\,m^{-3}}$ for ρ_s; $1.99 \times 10^{-4}\,\mathrm{m}$ for d_{50}; $0.0341\,\mathrm{m}$ for k_s and $0.321\,\mathrm{m^2\,s^{-1}}$ for ϵ_s. Note the optimum ϵ_s value here is much higher than its value when we only optimised for ϵ_s, showing a clear difference in results if parameters are optimised individually or jointly. To summarise, the optimum sediment is less dense, larger, erodes less easily and diffuses at a greater rate than assumed in the original simulation.

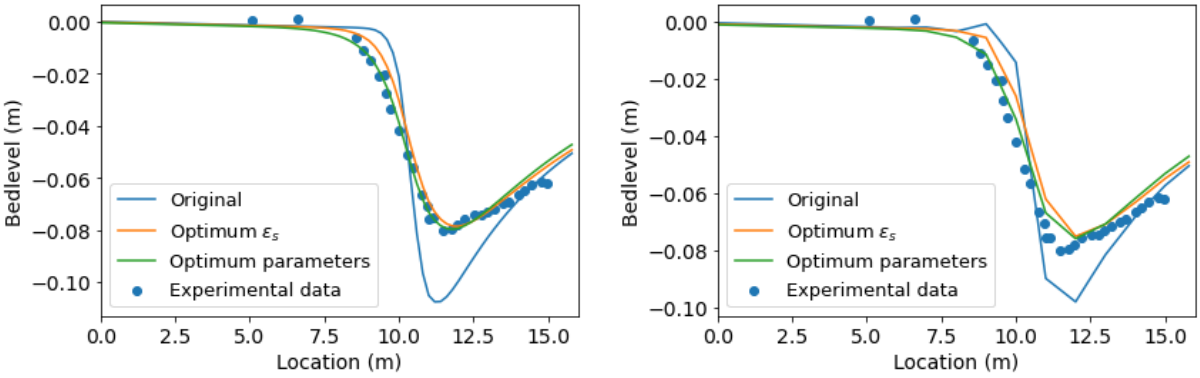
Figure 3.11a compares the final bedlevel obtained using either all four optimum parameters, just the optimum diffusivity coefficient ϵ_s, or the original values from Villaret et al. (2016). It shows that ϵ_s has the largest impact on accuracy, but that using optimum choices for all four parameters improves the accuracy further, in particular the gradient of the slope. In order to test the robustness of these optimum parameters, we re-run the test case using a coarser mesh of $\Delta x = 1\,\mathrm{m}$ on which the initial trench profile is



(A) Minimisation of output functional.

(B) Convergence to optimum $\rho_s, d_{50}, k_s, \epsilon_s$.

FIGURE 3.10. Using the adjoint method with a minimisation algorithm to find the optimum sediment density ρ_s, sediment size d_{50}, bed reference height k_s and diffusivity coefficient ϵ_s for the Trench test case.



(A) Fine.

(B) Coarse.

FIGURE 3.11. Comparing the final bedlevel on both a fine and coarse mesh as a result of using the original parameters from Villaret et al. (2016); the optimum ϵ_s parameter; and the optimum ρ_s, d_{50}, k_s and ϵ_s parameters for the Trench test case. The experimental data is also shown demonstrating the accuracy improvements from using the adjoint method optimum values.

not accurately defined and therefore its shape is altered. Figure 3.11b shows the accuracy improvements with the coarser mesh are similar to those on the finer mesh, including the improved gradient from using four optimum parameters. This suggests these optimum parameters are not resolution dependent and, thus, our adjoint framework can accurately calibrate multiple uncertain scalar parameters in this test case.

3.5 Tsunami inversion

Our final test case in this chapter is a tsunami-like event. A tsunami event is often difficult to simulate due to the large array of uncertain parameters, especially for historical scenarios where the only record is in the form of sediment deposits (see for example Tang et al., 2018; Dourado et al., 2021). These uncertain parameters are often estimated by feeding educated guesses into a model and adjusting their values

accordingly by comparing the model results with the real data (see for example Dourado et al., 2021). Therefore, it is an ideal event on which to apply our adjoint hydro-morphodynamic model framework.

For our test case, we consider the experiment in Kobayashi and Lawrence (2004), where a series of eight tsunami-like solitary waves break over an initially linear sloping beach. We have already validated this test case in Section 2.5.1 where we showed that our model agrees well with both the experimental data and the other model results. As discussed there, the hydrodynamic component of our model is not best suited for representing the propagation of solitary waves. However, the good agreement between our model and the experimental data suggests that this limitation does not result in substantial divergence from the experimental set-up and that our results are valid. Throughout this section, unless otherwise stated, we use the set-up from Section 2.5.1 for this test case.

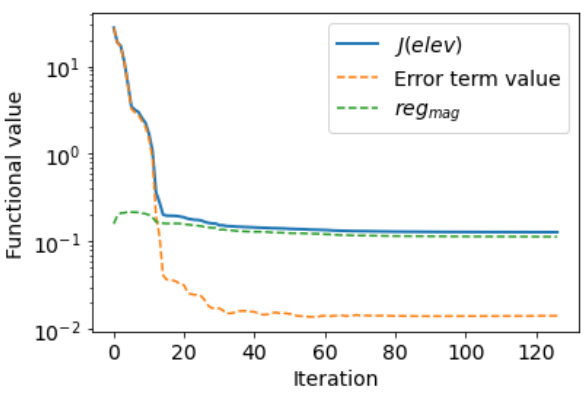
3.5.1 Reconstructing reference wave from sediment deposits

As discussed in Section 3.2, the adjoint method is ideally suited to cases which have multidimensional parameters that vary in space and/or time. Thus, in this section, we consider as our uncertain parameter the time-varying inflow tsunami-like wave boundary condition. We use a dual twin experiment to verify the ability of our adjoint framework to reconstruct it, and consider the bedlevel generated by the reference wave (2.94) at every timestep to be the 'true' data. Instead of simulating the full series of eight waves from the experiment, we only simulate one wave because of time constraints and because we are not matching with experimental data. However, as in Section 2.5.1, we use a morphological acceleration factor of four, and hence our model run is approximately equivalent to simulating four waves in the experiment, because the large separation between the waves in the experiment means that they do not interact with each other. For our minimisation algorithm initial guess, we assume the wave is caused by a sudden rupture in the Earth's crust creating the discontinuous wave profile

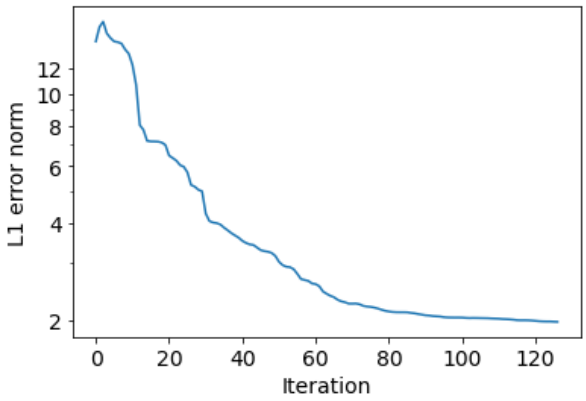
$$\eta_{\text{initial}}(t) = \begin{cases} 0.05\,\text{m} & t < 7.5\,\text{s}, \\ 0\,\text{m} & \text{otherwise}, \end{cases} \tag{3.16}$$

as shown in Figure 3.13. Note that only the first 10 s of the wave are considered to be uncertain because we know that the 'true' free surface perturbation is only non-zero at the boundary between 2 s and 6 s. Therefore, it is unnecessary to consider the whole time region. For the output functional, we use (3.15) with an extra time integral, and set x_i in the Gaussian function $\mathring{z}_b(x; x_i)$ (defined in (3.13)) to be the locations of the experimental data from Kobayashi and Lawrence (2004). Additionally, we set $\alpha = 1$ in the error term and $\beta_i = 10^{-4}$ in the Tikhonov regularisation term. Note that, for simplicity, hereafter we use reg_{mag} to denote the Tikhonov regularisation term.

Figure 3.12a shows that the output functional, together with the error term and reg_{mag} within the functional, decreases as the number of iterations of the minimisation algorithm increases. Notably, even when the value of reg_{mag} is larger than that of the error term, the latter is still decreasing. This minimisation of the functional results in the convergence of the $L1$ error norm between the reference input



(A) Minimisation of the output functional. The convergence of the error term and reg_{mag} in (3.11) are also shown.



(B) Convergence of the $L1$ error norm between the model wave and the reference tsunami-like wave.

FIGURE 3.12. Using the adjoint method with a minimisation algorithm to reconstruct the reference wave for of the Tsunami test case.

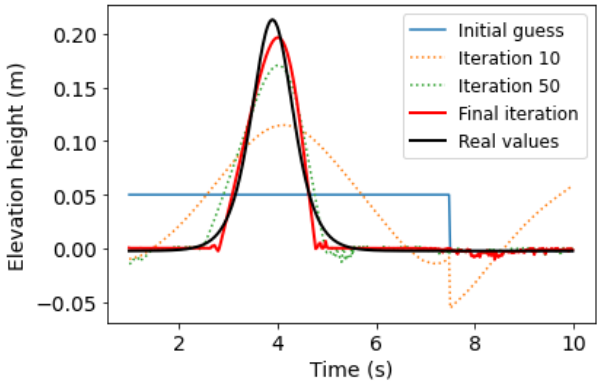


FIGURE 3.13. Comparing the reconstructed wave at a series of different iterations of the minimisation algorithm (including the final iteration) with the reference wave in the Tsunami test case.

and model input waves, as shown in Figure 3.12b. The effect of the minimisation on the reconstructed wave itself is shown in Figure 3.13, revealing that the wave has the correct general shape by iteration 50. Figure 3.13 also shows that the final iteration approximates the reference wave very well, confirming that our adjoint framework can reconstruct a known solitary input wave.

3.5.2 Finding the optimum wave from sediment deposits

Given the success of the dual twin experiment, we now consider the inversion of the wave from the experimental sediment deposit data in Kobayashi and Lawrence (2004). This data only exists for the final bedlevel and thus the problem would be ill-posed without regularisation terms in our output functional. To ensure stability, the wave elevation is set to be 0 m at both the start and end of the simulation and the wave is only considered uncertain during the middle, where we initialise the minimisation algorithm using an initial elevation of 0.05 m (see Figure 3.15). This replicates the initial guess of a discontinuous wave profile from Section 3.5.1. Here we use the same model parameters as for the dual twin experiment, but

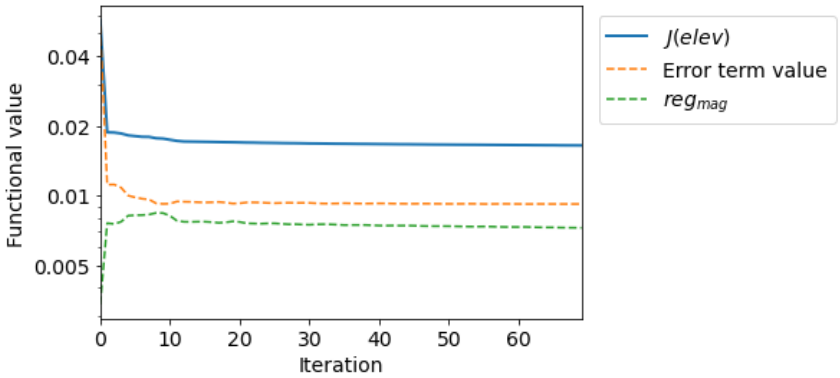


FIGURE 3.14. Minimisation of the output functional with only reg_{mag} for the optimum wave for the Tsunami test case. The convergence of the error term and the change in reg_{mag} in (3.17) are also shown separately.

simulate two solitary waves instead of one. Given that we are using a morphological acceleration factor of four, simulating two waves is approximately equivalent to simulating the eight waves in the experiment (see Section 2.5.1) and thus means our model results can be compared against the experimental data.

As a first test, we use the same output functional as in the dual twin experiment (3.15), recalling that for simplicity we use reg_{mag} to denote the Tikhonov regularisation term. In (3.15), the true values are now the final bedlevel experimental data from Kobayashi and Lawrence (2004) (shown in Figure 3.16) and, as before, the x_i in the Gaussian function, $\mathring{z}_b(x; x_i)$ (3.13), are centered at the locations of the experimental data. Finally, we set $\alpha = 1$ and $\beta_i = 5 \times 10^{-6}$ and note that there is no need for a time integral here because the experimental data only exists at one point in time.

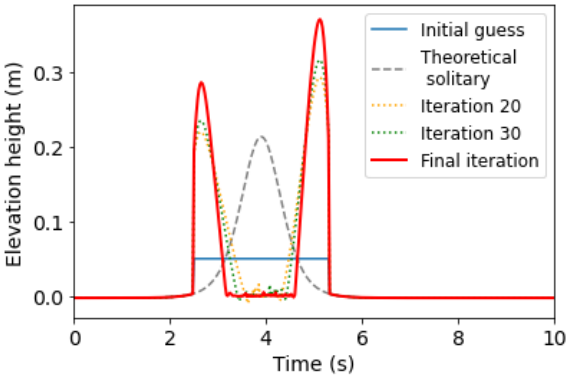


FIGURE 3.15. Optimum wave (with only reg_{mag}) compared to selected intermediary iterations of the minimisation, and to the theoretical solitary wave (2.94) from Kobayashi and Lawrence (2004), for the Tsunami test case.

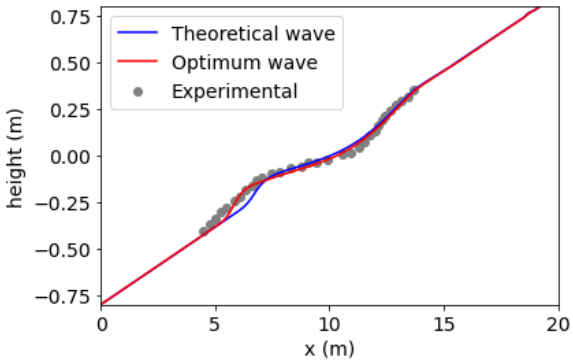


FIGURE 3.16. Final bedlevel simulated using the optimum wave (with only reg_{mag}) compared with that from the experiment and that simulated using the theoretical solitary wave (2.94) from Kobayashi and Lawrence (2004), for the Tsunami test case.

Figure 3.14 shows how the output functional (3.15) decreases as the number of iterations of the minimisation algorithm increases. This causes the error term between the experimental data and the model output to decrease, but, in contrast the reg_{mag} term increases because the magnitude of the initial guess is too small to create sufficient bedlevel changes. After about 20 iterations, the minimisation algorithm is close to convergence, which is further illustrated by Figure 3.15, where the wave at iteration 20 is very similar

to the optimum wave. This figure also shows that the optimum wave has a very different shape to the theoretical solitary wave (2.94) used in Kobayashi and Lawrence (2004) to describe the incoming wave observed in their experiment. In particular, the optimum wave has large unphysical jumps in waveheight and two peaks, although we know it should be single peaked. Figure 3.16 shows that this optimum wave results in a much more accurate bedlevel relative to the experimental data than the theoretical solitary wave, particularly in the area of deposition between 5 m and 10 m. However the unphysical nature of the optimum wave means this is not an acceptable solution.

To deal with this issue, we re-run the inversion with an extra continuity regularisation term in the output functional, which enforces continuity in the incoming wave (for simplicity this term is hereafter referred to as reg_{cont}). The output functional is then

$$J(z_b, \mathbf{m}) = \alpha \sum_{i=1}^{k} \frac{\int_\Omega |\mathring{z}_b^{\text{model}}(x; x_i) - \mathring{z}_b^{\text{true}}(x; x_i)|^2 \, dx}{\int_\Omega |\exp^{-50(x-x_i)^2}|^2 \, dx} + \sum_{i=1}^{N_{\text{in}}} \beta_i \int_\Omega |m_i|^2 \, dx + \sum_{i=2}^{N_{\text{in}}} \gamma_i \int_\Omega |m_i - m_{i-1}|^2 \, dx, \tag{3.17}$$

where m_i represents the input wave at time t_i and the scalar parameters are $\alpha = 1$, $\beta_i = 5 \times 10^{-6}$ and $\gamma_i = 5 \times 10^{-2}$. The rest of the set-up is kept the same as before.

Figure 3.17 shows that again the output functional (3.17) decreases as the number of iterations of the minimisation algorithm increases, causing the value of its error term to also decrease. The trend in reg_{mag} and reg_{cont} is less uniform than when we were only using one regularisation term (see Figure 3.15) because here the two regularisation terms must trade off against each other, as well as the error term. The pattern can be loosely interpreted as follows: for the first 20 iterations, the minimisation algorithm minimises reg_{cont}; between iterations 20 to 30, it minimises the error term at the expense of reg_{mag}; and after iteration 30, it again minimises reg_{cont}. This interpretation is confirmed by the differences between iterations 20, 30 and the final iteration in Figure 3.18. This figure shows that adding reg_{cont} makes the optimum wave physically realistic and single peaked, and hence shows the importance of using sensible regularisation terms.

Figure 3.18 also compares the optimum wave found in this section with the theoretical solitary wave (2.94) that Kobayashi and Lawrence (2004) used to describe the incoming wave observed in their experiment. In particular it reveals that our optimum wave has a wider base and a smaller amplitude than (2.94). Consequently, Figure 3.19 shows the simulated final bedlevel from the optimum wave agrees very well with the experimental data and this agreement is much better than that achieved with the theoretical wave, particularly in the area of deposition between 5 m and 10 m. This figure also shows that including reg_{cont} leads to a slightly worse agreement between the final simulated bedlevel and the experimental data than if it is not included. However, this slight disadvantage is outweighed by the advantage that this new optimum wave is realistic.

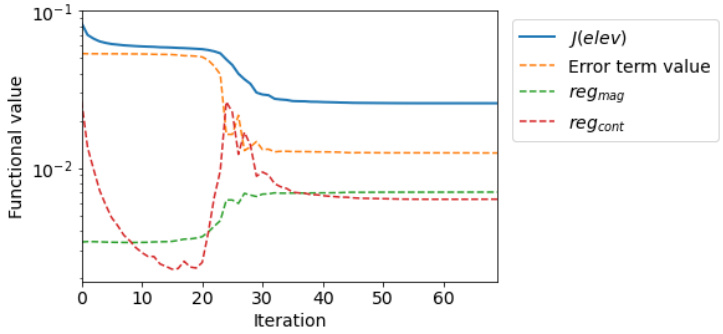


FIGURE 3.17. Minimisation of the output functional with both reg_{mag} and reg_{cont} for the optimum wave for the Tsunami test case. The convergence of the error term and the change in reg_{mag} and reg_{cont} in (3.17) are also shown separately.

In summary, this test case shows that our adjoint framework can be used to accurately invert for the shape and magnitude of tsunami-like waves from final bedlevel sediment deposits. This is a very promising result because it means our framework can be applied to historical tsunami sediment deposit data to invert for the tsunami wave's characteristics.

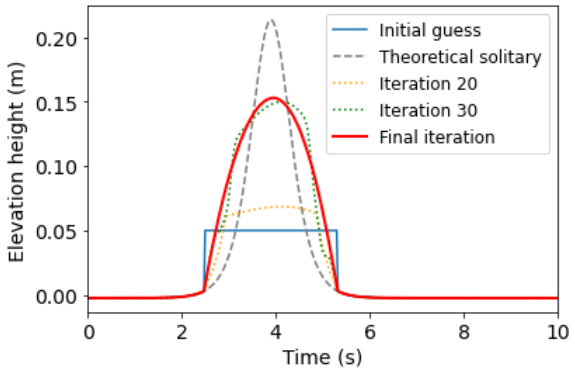


FIGURE 3.18. Optimum wave (with reg_{mag} and reg_{cont}) compared to selected intermediary iterations of the minimisation algorithm and to the theoretical solitary wave (2.94) for the Tsunami test case.

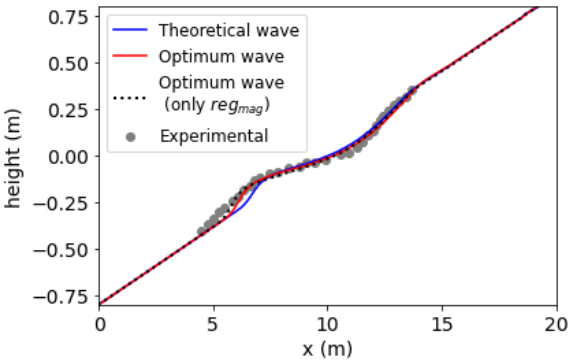


FIGURE 3.19. Final bedlevel simulated using the optimum wave obtained with reg_{mag} and reg_{cont} compared with that from the experiment, that simulated using the optimum wave obtained with reg_{mag} and that simulated using the theoretical solitary wave (2.94) for the Tsunami test case.

3.6 Conclusion

In this chapter, we have developed the first freely available and fully flexible adjoint hydro-morphodynamic model framework. By fully flexible, we mean that the use of *pyadjoint* allows us to investigate how uncertainty in any parameter in the hydro-morphodynamic model affects any output functional. Hence, we have shown that a single run of our adjoint framework can perform calibration, inversion or sensitivity analysis for multiple scalar and/or spatially-varying parameters. We have verified these capabilities using dual-twin experiments and shown that these parameters produce physically sensible results. In particular, the optimum parameter values obtained through applying these inversion and calibration capabilities to experimental data always result in better agreement between the model outputs and the data. A notable

achievement of our work is that our approach is capable of reconstructing the shape and magnitude of incoming tsunami-like solitary waves from experimental sediment deposit data. In future work, our framework will be applied to historical tsunami sediment deposit data to invert for the characteristics of historical tsunami waves. This will lead to a significant step forward in the understanding of historical tsunami events and hence help predict the impact of future events, thus mitigating the risk to coastal communities who live in tsunami-prone regions.

Finally, the type of uncertainty information obtained using the adjoint method would be either impractical or much more computationally expensive to obtain using other methods, such as Monte Carlo or the tangent linear method. Therefore, the knowledge gained by our computationally efficient adjoint framework can be invaluable to a variety of users and stakeholders in understanding and mitigating the impacts of coastal and fluvial hazards.

Code availability

The relevant code for the adjoint hydro-morphodynamic model framework presented in this chapter is stored at https://github.com/mc4117/adjoint_hydro_morphodynamic.

CHAPTER 4

Applying mesh movement methods to help overcome multi-scale challenges

Highlights

- First application of mesh movement with a full hydro-morphodynamic model;
- Mesh movement results in clear improvements in accuracy and computational cost;
- Mesh movement monitor functions are proposed which do not require significant fine-tuning.

Summary

Hydro-morphodynamic models can be required to resolve spatial scales ranging from sub-metre to hundreds of kilometres, and are computationally expensive. Here, in order to tackle both these issues, we apply mesh movement methods to a depth-averaged hydro-morphodynamic model for the first time. Mesh movement methods are particularly well-suited to hydro-morphodynamic problems because they allow the mesh to move in response to evolving flow and morphology structures.

The capability of this new framework is demonstrated using test cases that exhibit complex evolving bed profiles and/or have moving wet-dry interfaces. For all test cases considered, mesh movement methods reduce discretisation error and computational cost. We also show that the best choice of mesh movement monitor function is fairly predictable and does not require significant fine tuning between cases, thus facilitating the use of mesh movement methods on further hydro-morphodynamic problems.

Publications and Disclaimer

The work in this chapter has been submitted for publication to *International Journal on Geomathematics*, where it has been accepted pending minor corrections. The preprint of this paper is available here:

Clare, M. C., Wallwork, J. G., Kramer, S. C., Weller, H., Cotter, C. J., & Piggott, M. D. (2022). 'Multi-scale hydro-morphodynamic modelling using mesh movement methods'. GEM-International Journal on Geomathematics, 13(1), 1-39. https://doi.org/10.1007/s13137-021-00191-1

Note that this work was conducted in collaboration with Joseph Wallwork, who submitted the following thesis to Imperial College London.

Wallwork, J. G. (2021). 'Mesh Adaptation and Adjoint Methods for Finite Element Coastal Ocean Modelling'. PhD thesis, Imperial College London, London, UK.

In his thesis, Wallwork presents mesh movement applied to the quasi-1D migrating trench test case. This test case is also presented in Section 4.3 of this chapter, but using a different monitor function and focusing on parameters more pertinent to this thesis. All results presented here are produced by the author of this thesis.

4.1 Introduction

Hydro-morphodynamic models are computationally expensive, especially because they are often required to simulate very long-term morphological effects with relatively small timestep sizes in order to resolve hydrodynamic features such as waves and tides. This computational cost is often further increased by the need to run the model more than once for calibration purposes (see for example Villaret et al., 2016; Harris et al., 2018; Clare et al., 2021). Furthermore, when applied to coastal regions, hydro-morphodynamic models must resolve problems with complex and fundamentally multi-scale domains. Unstructured mesh models, such as those based on finite element methods (Piggott et al., 2008a,b), are ideally suited to providing the required mesh flexibility. Multiple tools exist for generating multi-scale fixed meshes, which are suitable for a wide range of geophysical applications (for example, Avdis et al., 2018). However, these are insufficient in cases with significant morphology changes, as the regions that require finer mesh resolution vary throughout the simulation. Therefore, many fluvial and coastal scenarios are unfeasible to model using standard fixed meshes of appropriate resolution.

A solution to this time-dependent multi-scale issue is to use mesh adaptation methods, such as mesh movement (also known as r-adaptation) or h-refinement. Such methods provide the potential to improve result accuracy and/or reduce computational cost (Budd et al., 2009). The *Firedrake* framework used in our hydro-morphodynamic model (see Section 2.2) is well suited to mesh adaptation methods because it can readily solve the complex non-linear PDEs often used in these methods (see McManus et al., 2017). Hence, both h-refinement and mesh movement have been successfully used in the wider Firedrake framework (see, for example, Barral et al., 2016; McManus et al., 2017; McRae et al., 2018; Wallwork et al., 2020a,b). The focus and novelty of this chapter is that we apply mesh movement methods to our depth-averaged hydro-morphodynamic model, the first implementation of these type of methods with a depth-averaged hydro-morphodynamic model and also the first implementation of mesh movement methods in the coastal ocean *Firedrake*-based model *Thetis*. Through applying these methods, we aim to lower the computational cost of our model for the same or increased accuracy, as compared to using fixed uniform meshes. Any computational cost reduction means more problems are feasible to simulate and more calibration runs can be performed for the same cost.

Mesh movement methods work by fixing the mesh topology and dynamically moving the (fixed number of) mesh nodes, thus allowing different regions of the domain to vary between low and high resolution as flow structures pass through them. In contrast, with h-refinement, the mesh topology is substantially changed, with cells locally created or destroyed and mesh connectivity altered (Piggott et al., 2005, 2009). In this chapter, we apply mesh movement with a hydro-morphodynamic model, but previous works apply h-refinement instead (see for example Mayne et al., 2002; Delandmeter, 2017; Benkhaldoun et al., 2013). Whether mesh movement methods or h-refinement are more appropriate is dependent on the case considered. If the regions warranting the highest mesh resolution move continuously with the flow, as is the case with most hydro-morphodynamic problems, then mesh movement is arguably more appropriate because it can track the features. However, if features were to appear suddenly requiring

extra degrees of freedom then h-refinement, or combined mesh movement with h-refinement, may be more appropriate (Piggott et al., 2005). Another advantage of mesh movement is that the number of mesh nodes remains fixed during the simulation, thus simplifying data structures.

Our model also has key advantages and/or differences compared with the hydro-morphodynamic models for which h-refinement has been successfully applied. For example, Mayne et al. (2002) show that h-refinement can accurately capture sharp differences in sediment concentration, but their model simulates cohesive sediments (*i.e.* clay and silt) rather than the non-cohesive sediments that are present in the fluvial and coastal environments which we model. Similarly, Delandmeter (2017) show that h-refinement can accurately capture sharp gradients in hydrodynamics, but their model does not simulate the bedlevel changes caused by sediment transport meaning their results are inaccurate especially in cases such as the ones we consider, where there is a lot of bed movement. Finally, Benkhaldoun et al. (2013) again shows that h-refinement can accurately capture sharp gradients in sediment concentrations, but their model uses finite volume methods, whereas our model uses a discontinuous Galerkin finite element discretisation, shown in Section 2.1 to have several advantages for hydro-morphodynamic problems. Moreover, all the previous applications of mesh adaptation focus on refining the mesh based on the hydrodynamics or the sediment concentration, whereas in this chapter we refine the mesh based on the bed profile. This is an arguably better metric to use when considering erosion, because in practice the variable of most interest when simulating erosion is the bed.

In this chapter, we test our novel hydro-morphodynamic moving mesh framework using multi-scale test cases with complex evolving bed profiles, which are difficult to capture accurately on coarse and/or fixed meshes. We also consider test cases with a moving wet-dry interface, a frequent scenario in the modelling of coastal zones, which can again be difficult to capture on a fixed mesh. Through these test cases, we are able to establish a generalised mesh movement methodology that can be applied to a variety of fluvial and coastal zone test cases.

The remainder of this chapter is structured as follows: in Section 4.2 we outline the mesh movement methods used; in Section 4.3, Section 4.4 and Section 4.5 we apply mesh movement to test cases with a complex bed profile in the fully wet domain, a simple bed profile in the wet-dry domain and a complex bed profile in the wet-dry domain respectively; and, finally, in Section 4.6 we conclude this chapter.

4.2 Mesh Movement

Mesh movement can be applied using a variety of different approaches, including imposing a prescribed mesh velocity (Donea et al., 2017), re-interpreting the mesh as a structure of stiff beams (Farhat et al., 1998) or enforcing mesh transformations using monitor functions (Huang et al., 1994; Budd and Williams, 2009a). In this chapter, we take the monitor function based approach and implement it using *Firedrake*. This section draws on Wallwork (2021), which we refer the reader to for further detail.

The monitor based function approach determines the way in which the mesh is moved by defining a discrete representation of a sufficiently smooth map

$$\mathbf{x} : \Omega_C \to \Omega_P, \qquad \mathbf{x}(\mathcal{H}_C) = \mathcal{H}_P. \tag{4.1}$$

Here Ω_C is the *computational* reference domain with an associated computational mesh, $\mathcal{H}_C$, and Ω_P is the *physical* domain with an associated physical mesh, $\mathcal{H}_P$, on which our hydro-morphodynamic model is solved. The computational domain and its associated mesh $\mathcal{H}_C$ are fixed, but the physical domain and $\mathcal{H}_P$ are allowed to vary with time throughout the simulation. Note that the physical mesh $\mathcal{H}_P$ has an identical topology to $\mathcal{H}_C$ and hence, initially, we set $\mathcal{H}_P$ equal to $\mathcal{H}_C$, following standard practice. The mesh's 'density' is then prescribed by a user-specified monitor function $m : \Omega_P \times [0, T] \to (0, \infty)$, a concept first proposed in White (1979). This monitor function determines the way in which mesh resolution is distributed over the domain and must therefore be chosen carefully.

We could combine the hydro-morphodynamic model and mesh movement equation into a single system of equations solved at each timestep in an *Arbitrary Lagrangian-Eulerian (ALE)* type approach (Piggott et al., 2005; Budd and Williams, 2009a; Donea et al., 2017). However, for a hydro-morphodynamic problem, this approach is unnecessarily computationally expensive because it requires the mesh to be moved at every timestep which may be superfluous for many problems (see Figure 4.4, for example). Therefore, we take an alternative approach and only move the mesh after a user-defined number of timesteps. A consequence of this choice is that we can no longer assume we are solving our problem in a moving reference frame, and thus the solution variables must be interpolated between meshes during the PDE time integration loop. Here this interpolation is implemented using *libsupermesh* (Maddison et al., 2017) and performed using a conservative Galerkin projection based approach. Although the simple act of interpolation can introduce additional errors, our conservative Galerkin approach ensures that quantities are conserved in the mesh-to-mesh transfer and it is generally more accurate than the alternative simpler interpolation methods available in *Firedrake*.

A core idea in the monitor function based approach is that the monitor function must be equidistributed across the mesh elements. This means that, in regions where the value of the function is high, the elements have a small area and vice versa where the value is small. In mathematical terms this can be derived by considering an arbitrary non-empty set $A \subset \Omega_C$ with a corresponding image $x(A) \subset \Omega_P$. Following Budd and Williams (2009b) the map x then equidistributes the monitor function m if

$$\frac{\int_A d\xi}{\int_{\Omega_C} d\xi} = \frac{\int_{x(A)} m(\mathbf{x}, t)\, d\mathbf{x}}{\int_{\Omega_P} m(\mathbf{x}, t)\, d\mathbf{x}}, \tag{4.2}$$

which with a change of variables becomes

$$\frac{\int_A d\xi}{\int_{\Omega_C} d\xi} = \frac{\int_A m(\mathbf{x}, t) \det \underline{\mathbf{J}}\, d\xi}{\int_{\Omega_P} m(\mathbf{x}, t)\, d\mathbf{x}}, \tag{4.3}$$

where

$$\underline{\mathbf{J}} = \frac{\partial \mathbf{x}}{\partial \boldsymbol{\xi}} \tag{4.4}$$

is the Jacobian transform with respect to the computational coordinate $\boldsymbol{\xi} \in \Omega_C$. As A is arbitrary, an equidistributed map $\mathbf{x}$ must satisfy

$$m \det \underline{\mathbf{J}} = \theta, \tag{4.5}$$

where

$$\theta(t) = \frac{\int_{\Omega_P} m(\mathbf{x}, t)\, \mathrm{d}x}{\int_{\Omega_C} \mathrm{d}\xi}, \tag{4.6}$$

is the normalisation coefficient. In this chapter, we do not consider any boundary deformations and hence $\Omega_P = \Omega_C =: \Omega$, so θ only depends on the monitor function. If (4.5) is solved to a sufficiently high accuracy, $\det \underline{\mathbf{J}}$ remains positive because both m and θ are strictly positive, which prevents mesh tangling (where one or more mesh elements become inverted) because no two vertices, or two edges, of the original mesh can meet. Additionally, it means that elements cannot become degenerate, because their area must remain strictly positive.

As our hydro-morphodynamic model is in $n = 2$ spatial dimensions (see Section 2.2), equation (4.5) is ill-posed because there are n unknowns but only one equation. There are various methods to manage this issue including using the variational method (Huang and Russell, 1998) or requiring that the map (4.1) be a harmonic function (Tang, 2005), but here, following McRae et al. (2018), we use an optimal transport approach. This approach finds the map which both minimises the displacement between $\mathcal{H}_C$ and $\mathcal{H}_P$, and equidistributes the monitor function. In mathematical terms this can be expressed as solving the minimisation problem

$$\min_{\mathbf{x}} ||\mathbf{x} - \xi||_{L^2(\Omega_c)} \tag{4.7}$$

subject to (4.5). The solution to this problem is unique and has the form

$$\mathbf{x}(\boldsymbol{\xi}) = \boldsymbol{\xi} + \nabla \phi_{\boldsymbol{\xi}}(\boldsymbol{\xi}), \tag{4.8}$$

for some scalar potential, $\phi : \Omega_C \times [0, T] \to \Omega_P$ (see McRae et al., 2018). As an aside, we note the interesting similarities between this problem and minimising a flow's kinetic energy subject to a spatially-varying divergence to produce a potential flow.

Substituting the expression (4.8) into (4.5) gives a nonlinear PDE of so-called Monge-Ampère type

$$m \det \left(\underline{\mathbf{I}} + \underline{\mathbf{H}}(\phi)\right) = \theta, \tag{4.9}$$

where $\underline{\mathbf{I}}$ is the identity matrix in $\mathbb{R}^{n \times n}$ and $\underline{\mathbf{H}}(\phi)$ denotes the Hessian of the potential with respect to the computational coordinates. The solution of (4.9) is always an equidistributed mesh (see for example Budd and Williams, 2009a; Budd et al., 2013; Weller et al., 2016; McRae et al., 2018) and a proper choice of monitor function m will produce a mesh that resolves features of interest such as travelling waves or steep gradients.

A disadvantage of using the Monge-Ampère type equation (4.9) is that it requires the domain to have a degree of convexity, although this is a sufficient (and not necessary) condition for solvability (Budd and Williams, 2009a). This convexity is not always present in real-world coastal test cases, which often have islands and/or coastal protrusions. The wetting-and-drying scheme in our model (see Section 2.2) overcomes this issue by allowing our domain to be a convex superset of the 'wet' region. Thus, the domain is no longer forced to follow topographic contours and, by meshing the entire domain (both 'wet' and 'dry' regions), we can avoid any voids in the mesh. This does lead to unnecessary nodes in the 'dry' region, but in our wetting-and-drying scheme the water depth there is trivial and hence these nodes do not significantly add to the computational cost of solving the model.

4.2.1 Solution Strategy

Mesh movement is achieved by solving (4.9) for the scalar potential on the current physical mesh and then transforming this mesh using (4.8). However, (4.9) is an elliptic PDE, with two sources of nonlinearity: one from the determinant and another from the product with the monitor function because the physical coordinates depend on the computational coordinates. Therefore, to solve (4.9), we follow McRae et al. (2018) and parabolise it as

$$-\frac{\partial}{\partial\tau}\Delta\phi = m\,\det\left(\underline{\mathbf{I}}+\underline{\mathbf{H}}(\phi)\right)-\theta, \tag{4.10}$$

where τ is the 'pseudotime'. In Awanou (2015), it is proven that if the 'pseudotimestep' $\Delta\tau > 0$ is sufficiently small and the initial guess for ϕ is sufficiently close to the solution, then as ϕ approaches a steady state (*i.e.* $\frac{\partial}{\partial\tau}\Delta\phi$ tends to zero), the solution of (4.10) converges to that of the original equation (4.9). Note that the second order derivative of ϕ in the pseudo-timestep term is used so that the L^2 projection in (4.12) is well-defined (see McRae et al. (2018) for more details). Wallwork (2021) finds a pseudotimestep of $\Delta\tau = 0.1$ to be sufficient for the type of test cases considered in this chapter and therefore we adopt this value.

Following McRae et al. (2018), we solve (4.10) with a continuous Galerkin finite element discretisation (CG) using the piecewise linear basis functions $P1$ in the scalar function space $\Psi \subset H^2(\Omega)$ and $P1^{2\times 2}$ in the tensor function space $\Sigma \subset L^2(\Omega)^{2\times 2}$. Whilst McRae et al. (2018) use periodic domains, we solve the PDE on non-periodic rectangular domains. Applying the forward Euler scheme, the weak form of (4.10) at pseudotimestep k is

$$\int_\Omega \nabla\psi\cdot\frac{\nabla\phi_{k+1}-\nabla\phi_k}{\Delta\tau}\,dx = \int_\Omega \psi\left(m\,\det(\underline{\mathbf{I}}+\underline{\mathbf{H}}_k)-\theta\right)dx \quad \forall\psi\in\Psi \tag{4.11}$$

where ψ is a test function. Following (Lakkis and Pryer, 2013), $\underline{\mathbf{H}}_{k+1}$ (the Hessian of ϕ_{k+1}) is then recovered via an L^2 projection

$$\int_\Omega \underline{\boldsymbol{\sigma}}\cdot\underline{\mathbf{H}}_{k+1}\,dx = -\int_\Omega(\nabla\cdot\underline{\boldsymbol{\sigma}})\,\nabla\phi_{k+1}\,dx + \int_{\partial\Omega}\nabla\phi_{k+1}\underline{\boldsymbol{\sigma}}\cdot\mathbf{n}, \quad \forall\underline{\boldsymbol{\sigma}}\in\Sigma \tag{4.12}$$

(where $\underline{\boldsymbol{\sigma}}$ is a test function) and then used for $\underline{\mathbf{H}}_k$ in the next iteration of (4.11) and so on. Note that at the first iteration this Hessian equals zero because the initial physical mesh is equal to the computational

mesh. These equations are solved to equilibrium, subject to a relative tolerance `tol` on the residual of (4.11). We then obtain the gradient of ϕ needed for our approach by using a vector version of the L^2 projection in (4.12), where we also enforce strong boundary conditions that mean the movement of the boundary nodes must be tangential to the corresponding segment. The mesh coordinates can then be updated according to the map (4.8).

Solving (4.11) and (4.12) repeatedly can be computationally expensive because each solve requires establishing a new map from Ω_C to Ω_P. Therefore, we employ several strategies to reduce this cost. Firstly, we use the final value of ϕ computed during one iteration as the initial guess for the next one, an appropriate choice due to the slow nature of morphodynamic changes that mean the bed profile does not change too significantly between mesh movement iterations (see Section 4.3–4.5). These slow changes also mean it is excessive to apply mesh movement at every timestep, hence our second cost reducing strategy is to choose an appropriate mesh movement frequency. This can be inferred either by small sensitivity studies or by using test case knowledge to infer the approximate rate at which the bed is moving. Finally, whilst obtaining a very accurate solution of (4.11) and (4.12) is mathematically attractive, it is unnecessary in practice and we can increase the relative solver tolerance `tol` to reduce computational costs. In Wallwork (2021), it is found that $\texttt{tol} = 10^{-3}$ results in a good compromise between computational cost and accuracy and thus this value is adopted throughout this chapter.

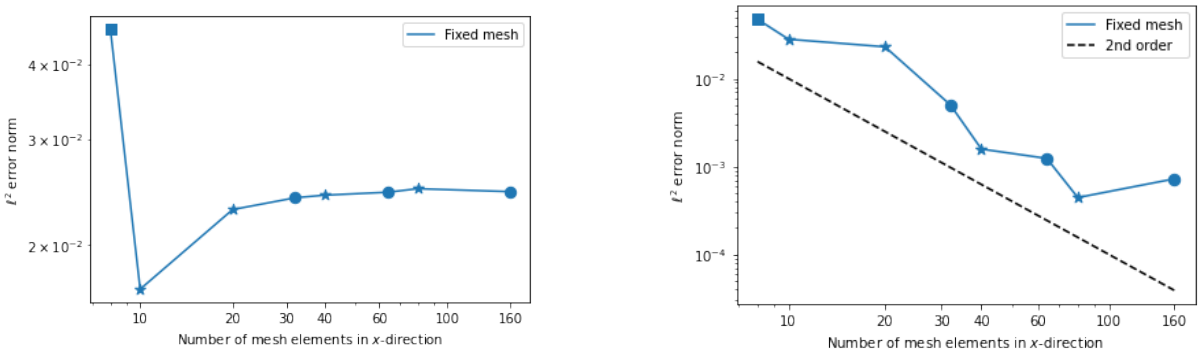
4.3 Complex bed profile test case: Migrating Trench

As a first test case, we consider the complex bed profile case of a migrating trench, based on an experiment in Van Rijn (1980). This test case has already been validated in Section 2.4.1 for a fixed uniform mesh and here we use the same setup and parameter values, in particular, the fully wet version of the model and a morphological acceleration factor of 100. The only difference is that we now use the calibrated value for the diffusivity coefficient $\epsilon_s = 0.183\,\mathrm{m}^2\,\mathrm{s}^{-1}$ found in Section 3.4.2.1.

4.3.1 Fixed uniform meshes

Before applying mesh movement methods to this test case, we study how the error changes on a series of fixed meshes, varying between 160 mesh elements to only 8 mesh elements in the x-direction. The existence of experimental data for this migrating trench test case means we can calculate the total bedlevel error, which is equal to the sum of the model error (due to using a simplified model to approximate a real-world problem) and the discretisation error (arising from using a finite mesh to solve the model equations). Throughout this chapter, the error in the bedlevel is calculated at the end of the simulation. Furthermore, in this section, the error is calculated using a pointwise ℓ^2 error norm at the location of the data points in the experiment.

Figure 4.1a shows the total error between the simulated final model bedlevel for a series of fixed uniform meshes and the experimental final bedlevel data. In this figure, the points marked with a star are those where the number of mesh elements in the x-direction is equal to $[10, 20, 40, 80]$, those marked with a



(A) Total error against experimental data.

(B) Discretisation error against a high resolution solution with 320 mesh elements.

FIGURE 4.1. Errors in the final bedlevel for a series of fixed uniform meshes for the Trench test case. The marker shapes distinguish differences in the definition of the initial profile on the mesh: for the circle points, the initial trench profile is accurately defined on the mesh; for the star and square points, the initial trench profile is ill-defined but in different ways.

circle are those where the number of elements in the x-direction is equal to $[32, 64, 160]$ and finally, the point marked with a square has 8 elements in the x-direction. This distinction is needed because, for the circle points, the initial trench profile is correctly defined on the mesh, but for both the star and square points the initial profile is inaccurately defined on the mesh. Moreover, these inaccuracies are different for the star and square points: for example, the correct start location for the second slope is $x = 9.5\,\text{m}$, but for the star points it starts at $x = 9.6\,\text{m}$, whereas for the square point it starts at $10\,\text{m}$. These differences in the initial trench profile definition result in fluctuations in the error trend and the surprising result that the total error with 10 elements is less than that for finer meshes, likely due to a difference in the effective diffusivity at this coarser resolution. Despite these fluctuations, the general trend is that the total error begins to plateau at around 20 mesh elements, and by 32 mesh elements the error has clearly converged to a value of approximately $0.0244\,\text{m}$.

To better understand this error trend, we isolate the discretisation error, by considering the final bedlevel obtained on a fixed uniform mesh with 320 mesh elements in the x-direction to be a high resolution approximation to 'the truth'. Figure 4.1b shows that the discretisation error decreases with an order of approximately $(\Delta x)^2$ as the mesh becomes finer. This is the expected order of convergence because the truncation error of the finite element representation and the associated discretisation of the non-linear shallow water equations are known to be second-order (Comblen et al., 2010). Note that the visual separation of the number of mesh elements in the x-direction in Figure 4.1b is the same as that in Figure 4.1a and, like with the total error, the sets converge at slightly different rates. Comparing the two figures shows that, for up to 20 mesh elements, the total error is dominated by the discretisation error but, after this point, it begins to be dominated by the model error. The reason is that, on a fixed uniform mesh of over 20 mesh elements, the initial trench profile is either accurately defined or close to being accurately defined. Hence the discretisation error is small, because the model is quite insensitive to changes in Δx and Δt (see Section 2.4.1.2).

4.3.2 Mesh movement

To apply mesh movement to the test case, we must choose an appropriate monitor function. In the previous section, we have determined that the main source of discretisation error is the inaccurate definition of the initial trench profile on the mesh. Even when a mesh is chosen so that the profile is initially accurately defined, as soon as the simulation starts, the bed begins to move so the profile may quickly become ill-defined, which explains the differences in total error even for the circle points in Figure 4.1a. Furthermore, there are large sections of the trench profile that are flat and remain relatively unchanged during the simulation, especially at the inflow. Therefore, a good choice of monitor function for this test case is one that results in increased mesh resolution in regions where the bed gradient and/or curvature is high, and reduced mesh resolution where it is lower. In this section, we choose the following monitor function because it allows us to control the effect of the first- and second-order bedlevel derivatives on the mesh movement

$$m(x,y) = 1 + \max\left(\alpha \frac{\|\underline{\boldsymbol{H}}(z_b)\|_F}{\max\limits_{x,y} \|\underline{\boldsymbol{H}}(z_b)\|_F}, \beta \frac{\|\nabla z_b\|_2}{\max\limits_{x,y} \|\nabla z_b\|_2}\right), \tag{4.13}$$

where $\underline{\boldsymbol{H}}(z_b)$ represents the Hessian of the bedlevel and $||\cdot||_F$ the Frobenius norm. Because we want to capture all the regions where the bed gradient or the curvature is high, our monitor function depends on the maximum of the first- and second-order derivatives, meaning that a region where only one of the derivatives is high is not given a lower weighting than a region where both derivatives are high. This monitor function introduces two user-defined parameters, α and β, which control the effect of the underlying bed profile on mesh movement: α controls the effect of the second-order derivative (curvature), whilst β controls the effect of the first-order derivative (slope). In order to be able to compare these parameters across test cases, we choose to normalise the derivatives in the monitor function. Note that both the Frobenius norm and the 2-norm are taken on an element-by-element basis and then projected into the $P1$ space, rather than being calculated component-wise. This is because we found that computing the norms component-wise results in an insufficiently smooth monitor function, leading to model divergence. Using an element-wise formulation introduces additional numerical diffusion that helps to counteract this effect.

Figure 4.2 shows an example of how the mesh is moved using (4.13) for this test case. It clearly shows the mesh refining around the steep trench slopes in the x-direction, but hardly moving in the y-direction because the bedlevel is uniform in this direction. Moreover, Figure 4.3 shows that this mesh movement results in a final bedlevel that is much more accurate relative to the high resolution solution, than the fixed mesh solution. This figure also shows that in most regions, the mesh movement solution approximates the experimental data better than the fixed mesh solution, though we recall here that the error relative to the experimental data includes model error, which is unaffected by mesh movement.

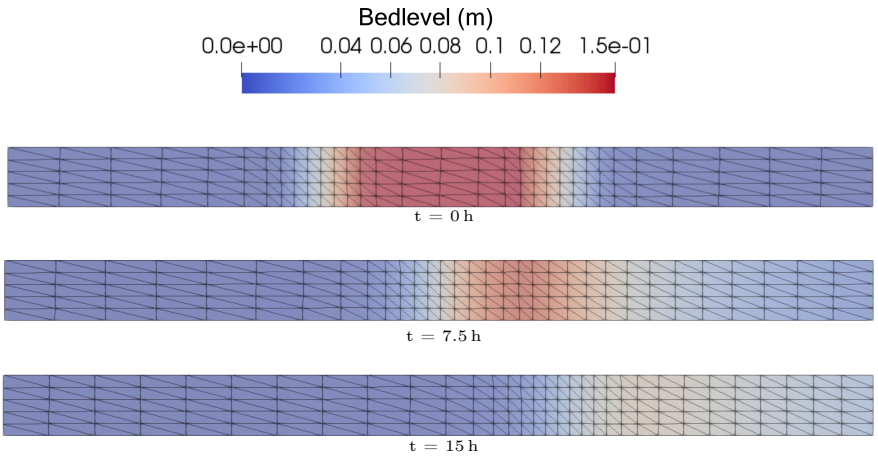


FIGURE 4.2. Snapshots of bedlevel and the underlying mesh for a moving mesh simulation of the Trench test case. The monitor function (4.13) with $\alpha = \beta = 3$ is applied to a mesh with 32 elements in the x-direction. Results are shown at three points in time demonstrating mesh movement to capture bed evolution to the right.

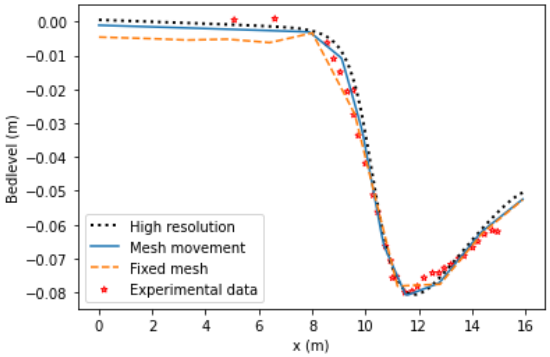


FIGURE 4.3. Comparison of final bedlevel resulting from fixed and moving mesh simulations of the Trench test case on a mesh with 32 elements in the x-direction. The moving mesh simulation applies the monitor function (4.13) with $\alpha = \beta = 3$ every 40 timesteps. Also shown are the experimental data from Van Rijn (1980) and the final bedlevel due to a high resolution fixed mesh simulation.

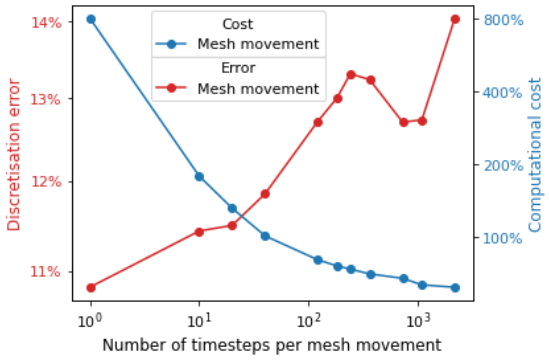


FIGURE 4.4. Trade-off between discretisation error and computational cost due to mesh movement frequency for the Trench test case. The monitor function (4.13) with $\alpha = \beta = 3$ is applied to a mesh with 32 elements in the x-direction. Note that, for all frequencies plotted, at least one mesh movement occurs. Errors and times are expressed as percentages relative to the fixed mesh case with the same resolution.

The mesh movement method used in this chapter allows us to choose how frequently the mesh is moved. If the modification to the mesh coordinates proposed by (4.8) is only minor, then solving this equation at every timestep of our hydro-morphodynamic model may prove an unnecessary expense. In fact, the slow moving nature of this test case leads us to infer that we can set a low mesh movement frequency without having an adverse effect on the results. More precisely, Figure 4.4 shows the effect of changing the frequency of the mesh movement on the discretisation error. The more timesteps between each mesh movement, the greater the discretisation error, but the lower the computational cost, as mesh movement methods have a non-negligible computational cost. Note that the total number of timesteps in this

simulation is 2160, hence in the lowest frequency case the mesh is only moved once at the beginning of the simulation, which is equivalent to using an appropriate initial fixed mesh. Significantly, the figure shows that if a large enough number of timesteps per mesh movement is chosen, we can reduce the computational cost below that of using a fixed uniform mesh and still be more accurate. This is because, although there is a cost associated with using the mesh movement algorithm, simulations using it require fewer model iterations to converge and thus it is possible to achieve a lower overall computational cost when the mesh movement frequency is low. This supports our argument, at least for this simple test case, that mesh movement can not only reduce error, but also reduce the computational cost of the simulation. In the remainder of this section, we choose to move the mesh after every 40 timesteps of the hydro-morphodynamic model (equivalent to moving the mesh 54 times during the simulation) because this provides a good balance between computational cost and accuracy.

Our mesh movement method also allows the user to set the values of the parameters α and β in (4.13), which control the mesh movement. In order to determine their optimum values, we conduct a small sensitivity study, but will seek a gradient-based approach in future work (see Section 4.6). We first study the effect of mesh movement on the total error and show the results in Figure 4.5 for a range of α values and numbers of elements. Note $\alpha = 0$ is equivalent to using a fixed uniform mesh. The total error is almost always minimized when α is large although CFL constraints relative to the hydrodynamic equations mean that it is not possible to test for larger values of α than those presented because the ratio between the maximum and minimum mesh spacing becomes too high. Figure 4.5 also shows that when there are fewer than 20 mesh elements, the total error converges to approximately 0.015 m, whereas for larger than 20 mesh elements, the total error converges to approximately 0.023 m.

As discussed previously, mesh movement methods can only affect the discretisation error component of the total error and should have no effect on the model error. Therefore, to better understand the best parameters to use in (4.13), we repeat the sensitivity study focusing solely on the discretisation error. Figure 4.6 shows that for all number of mesh elements, the discretisation error is minimised when α is close to 3. At $\alpha = 3$, the total error is approximately the same for all large number of elements and approximately the same as the value which the fixed mesh converges to (see Figure 4.5). This suggests that the reason the total error decreases as α gets larger in Figure 4.5 is due to a decrease in the model error. We have already shown that this problem is very sensitive to changes in diffusivity, and hence this model error change may be due to changes in the effective diffusivity in the simulation.

Additionally Figure 4.6 shows that, like with the total error, there is a distinction in the way the discretisation error varies with α and β above and below 20 mesh elements. For the smaller number of elements, the discretisation error is significantly smaller than that for the fixed mesh ($\alpha = \beta = 0$) and including the first-order derivative of the bedlevel in the monitor function is important (*i.e.* $\beta \neq 0$). By contrast, as the number of elements increases, the effect of mesh movement on the discretisation error decreases, and the first-order derivative becomes less important. This distinction is also present in Figure 4.1, and exists because below 20 mesh elements the trench profile is ill-defined on a fixed uniform mesh, hence the discretisation error dominates and mesh movement can have more of an impact. In contrast, above

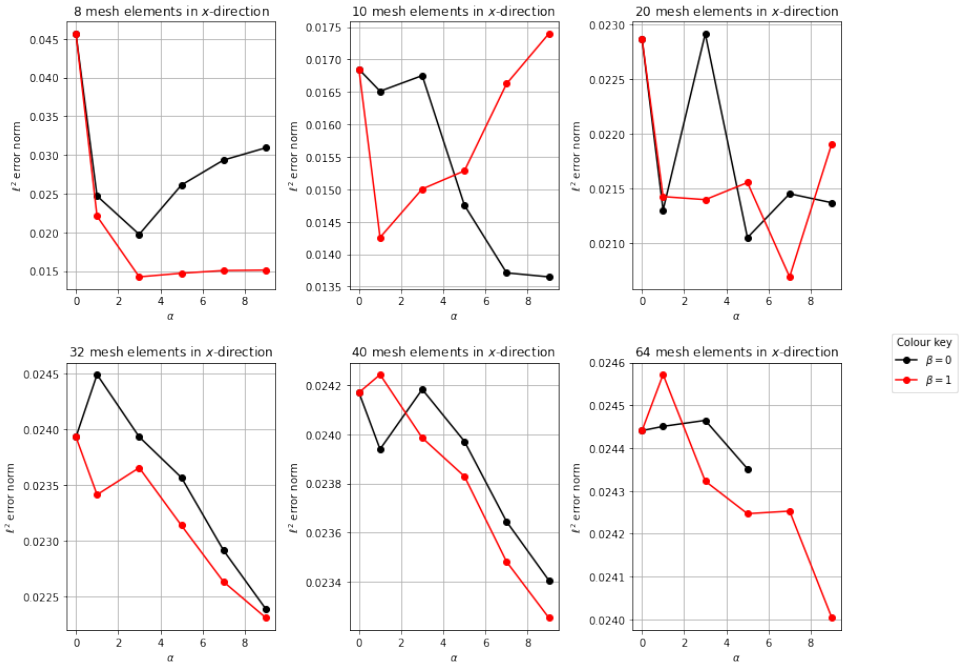


FIGURE 4.5. Total error curves for moving mesh simulations of the Trench test case under different values of α and β in the monitor function (4.13).

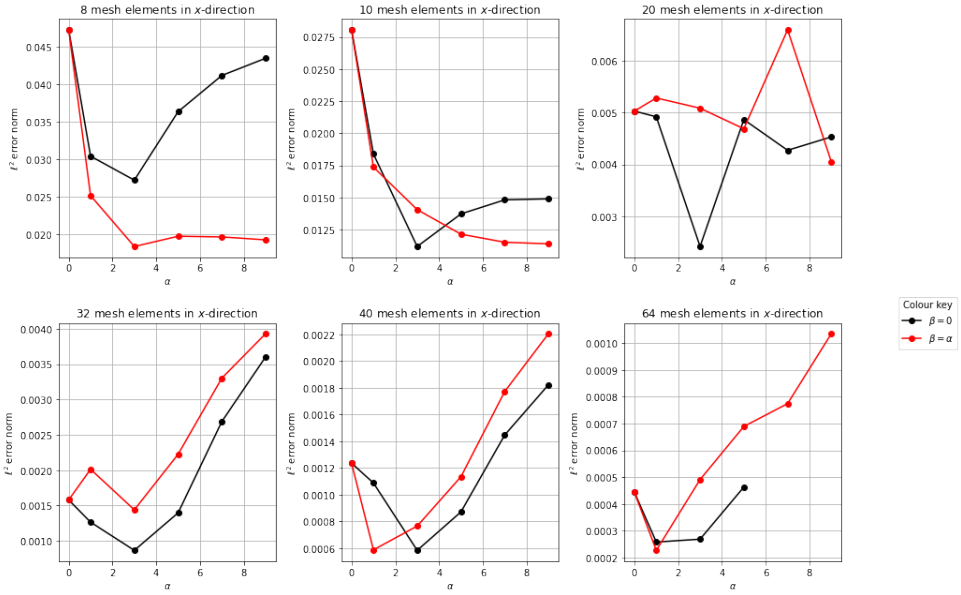


FIGURE 4.6. Discretisation error curves for moving mesh simulations of the Trench test case under different values of α and β in the monitor function (4.13).

20 mesh elements, the initial trench profile is either accurately defined or close to that, even on a fixed uniform mesh, and so the discretisation error is already very small (order of magnitude 10^{-3})).

Thus, we can conclude that for this test case a good general parameter choice for the monitor function (4.13) is $\alpha = 3$ and $\beta = 0$ for large numbers of mesh elements in the x-direction, and $\alpha = \beta = 3$ for small numbers of mesh elements in the x-direction. The greater dependence on the second-order derivative

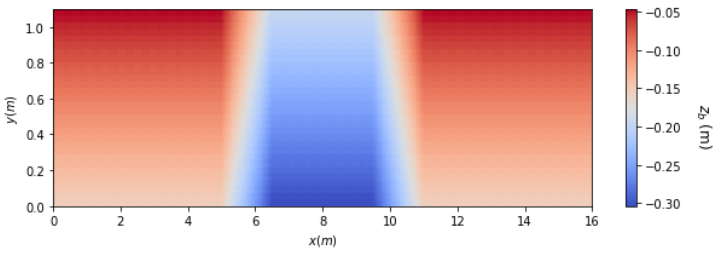


FIGURE 4.7. Initial bed profile for the Modified Trench test case.

than the first-order derivative is predictable because the regions of the trench that are most difficult to capture are the corners, where the absolute value of the second derivative is high.

4.3.3 Introducing a slope in the y-direction

The test case considered so far in this section is effectively 1D with little variation in the bedlevel or flow in the y-direction. However, most hydro-morphodynamic problems are 2D in nature and thus we modify the original migrating trench by introducing a slope of 0.1 in the y-direction. All setup parameters are kept the same and the initial profile is shown in Figure 4.7. The aim of this modified migrating trench test case is to show the generality of our moving mesh framework and that, if the monitor function parameters are known for a similar simple test case, then a full sensitivity analysis is not required to find the parameters for the more complex case.

4.3.3.1 Fixed uniform meshes

Before using mesh movement, we first consider a series of fixed uniform meshes with 8, 10, 20, 32, 40 and 64 mesh elements in the x-direction and an appropriate number in the y-direction to keep the elements in the fixed mesh approximately uniform. With the extra slope in the y-direction, we no longer have experimental data to compare against, but can still analyse the discretisation error by considering results obtained on a fixed uniform mesh of 320 mesh elements in the x-direction and 20 mesh elements in the y-direction to be a high resolution approximation of 'the truth'. Furthermore, as this is a 2D case, we can no longer consider the pointwise error at specific locations, but instead calculate the L^2 error norm over the whole domain. Figure 4.8 shows that as the fixed uniform mesh becomes finer, the error decreases approximately linearly. As discussed for the original trench test case, the truncation error of the finite element representation and the associated discretisation of the non-linear shallow water equations used here is known to be second-order. However, a reduction in the observed order of convergence is to be expected, because this modified test case is more complex.

4.3.3.2 Mesh movement

As with the original trench case, the main source of discretisation error is due to the initial trench profile being ill-defined on the mesh. Therefore, we use the same monitor function (4.13) here and Figure 4.9 shows an example of how the mesh is moved using this monitor function for this modified test case. The

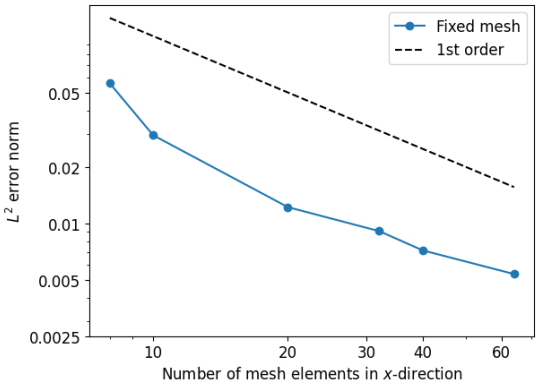


FIGURE 4.8. Discretisation error in the final bedlevel using a series of fixed uniform meshes in the Modified Trench test case. The reference solution here is a high resolution solution obtained on a fixed uniform mesh with 320 mesh elements in x-direction and 20 mesh elements in y-direction.

mesh moves in both the x and y-direction, with the element height in the y-direction being generally smaller at the top of the domain than at the bottom. This is especially visible between $x = 6.5\,\text{m}$ and $x = 9.5\,\text{m}$.

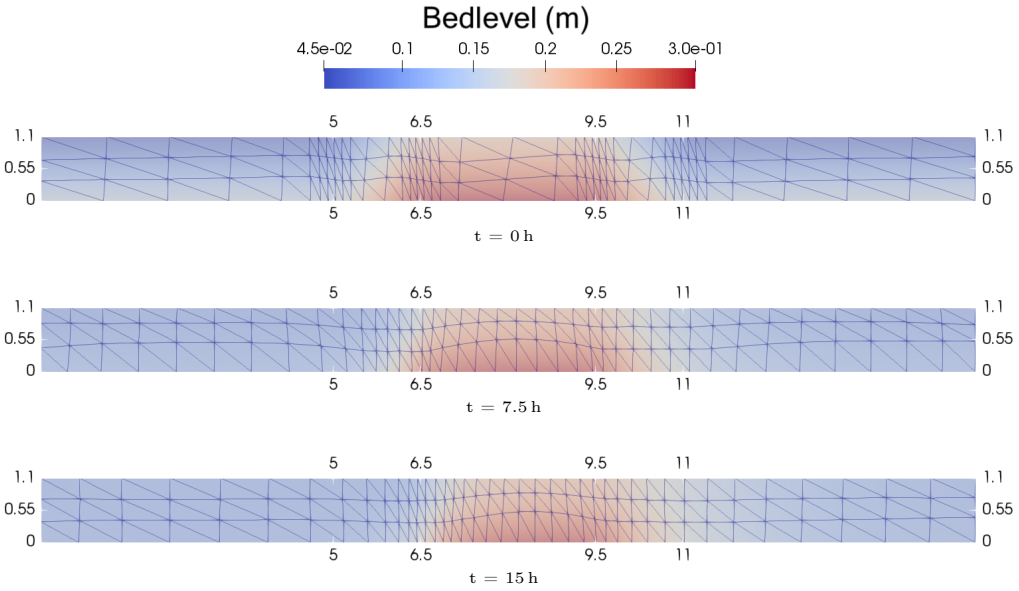


FIGURE 4.9. Snapshots of bedlevel and the underlying mesh for a moving mesh simulation of the Modified Trench test case. The monitor function (4.13) with $\alpha = 0$ and $\beta = 10$ is applied to a mesh with 40 elements in the x-direction and 3 elements in the y-direction. Results are shown at three points in time demonstrating mesh movement to capture bed evolution to the right.

Following our analysis from the original migrating trench test case, we move the mesh every 40 timesteps for this modified case and use monitor function parameters in (4.13) of a similar magnitude as the optimum ones found for that test case, instead of running a full sensitivity analysis. However, with the introduction of an extra slope, a different relationship between α and β may be optimal and thus we

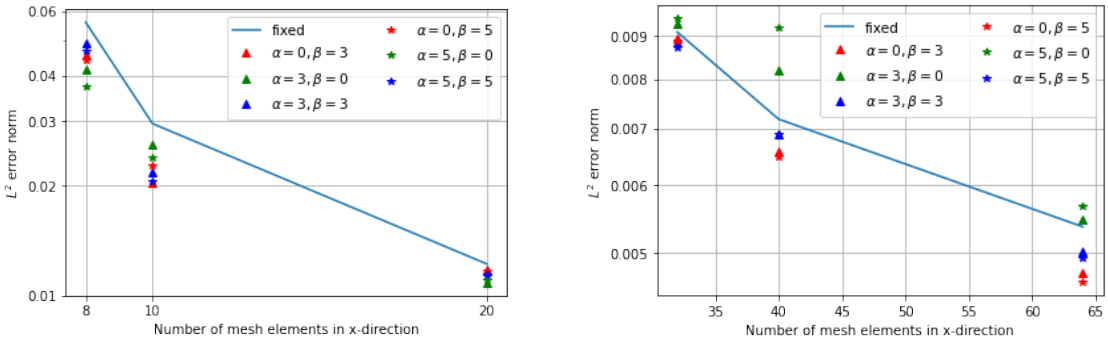
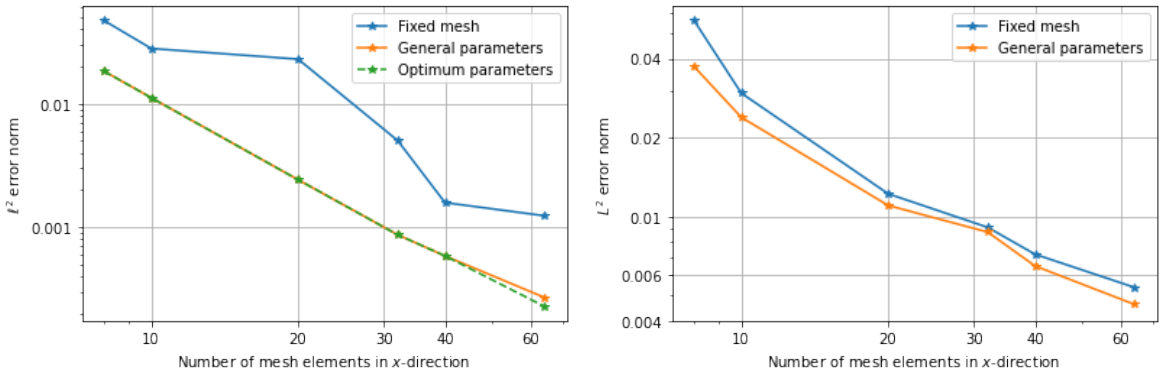


FIGURE 4.10. Discretisation error curves for moving mesh simulations of the Modified Trench test case under different values of α and β in (4.13).

conduct a small sensitivity study. Figure 4.10 shows that for small numbers of mesh elements in the x-direction, it is the second-order derivative of the bedlevel that is the most important, then for 32 mesh elements the first- and second-order derivatives are of equal importance, and finally for more than 32 mesh elements, the first-order derivative becomes the most important. These results are predictable from those of the original test case where for small numbers of mesh elements, mesh movement causes a large error reduction, but the effect lessens when the number of elements is large. We infer from this that, for the modified test case, the discretisation error is dominated by errors in the x-direction for small numbers of mesh elements, making the second-order derivative important (as in Section 4.3). As the number of mesh elements increases, the error in the y-direction starts to dominate, meaning the first-order derivative becomes the most important due to the linear slope in the y-direction.

Figure 4.10 also shows that using monitor parameters with a greater magnitude (a magnitude of 5) than that used in Section 4.3 (a magnitude of 3) results in a greater error reduction. This is due to the fact that for the modified case, the bedlevel is more complex and so more mesh movement is required. Therefore, whilst the magnitude of monitor function parameters obtained from simpler test cases results in error reduction in more complex problems, optimum mesh movement in this more complex case can be achieved by increasing the magnitude of the monitor function parameters.

Finally, Figures 4.11a and 4.11b present a summary of our discretisation error results for the original and modified cases respectively. Specifically, Figure 4.11a compares, for the original case, the optimum discretisation error achieved by mesh movement with that achieved using our generalised parameter set ($\alpha = 3, \beta = 3$ for small numbers of elements and $\alpha = 3, \beta = 0$ for large numbers) and Figure 4.11b shows, for the modified case, the discretisation error achieved by our generalised parameter set ($\alpha = 5, \beta = 0$ for small numbers of mesh elements; $\alpha = 5, \beta = 5$ for 32 mesh elements; and $\alpha = 0, \beta = 5$ for large numbers of mesh elements). Recall that as we did not conduct a full sensitivity analysis for the modified trench test case, we do not know what the optimum parameter set is. These figures show that moving mesh methods consistently result in a lower error than fixed mesh methods, and that our general parameter choice produces very similar error results to the optimum values. Thus, we have shown that mesh movement methods can be used to improve accuracy and reduce computational cost for quasi-1D and 2D cases with non-trivial bed profiles. We have also been able to use this test case to draw some conclusions about



(A) Original (Section 4.3.1 and 4.3.2). (B) Modified (Section 4.3.3).

FIGURE 4.11. Discretisation error comparison for fixed and moving mesh simulations of the original and modified trench test case. Both general and optimum parameters for the monitor function (4.13) are considered for the original case. Note that for the original case, the error shown is the pointwise ℓ^2 error norm and for the modified case the error shown is the L^2 error norm over the whole domain.

general good parameter choices for (4.13). In the next section, we test these general optimum parameter choices on a completely different 2D problem to see if they are again optimum or close to optimum choices.

4.4 Wet-dry interface test case: Beach Profile

Coastal problems often have a wet-dry interface, for example, as a wave or tide moves up a beach. These problems have been historically difficult to solve because of their computational expense, but mesh adaptation methods provide a way to retain accuracy, whilst improving efficiency, as in, for example, Zhou et al. (2013) where mesh movement methods are successfully applied to wet-dry test cases. However, to the best of our knowledge, these methods have not previously been used to solve coupled hydro-morphodynamic cases with wet-dry interfaces.

In this section, we consider the test case of a wave running up a beach slope, with the initial bed profile shown in Figure 4.12. The figure also shows an example of the simulated water surface due to the incoming wave, which is governed by

$$\mathbf{u} = 0.5\cos(0.05t) \quad \text{at } x = 0\,\text{m}. \tag{4.14}$$

The total simulated time is 60 000 s (approximately equivalent to 16 h) with a morphological acceleration factor of 250, meaning the full wave passes over the beach just over twice.

For this test case, we use a similar hydrodynamic setup to that used for the Balzano test case in Kärnä et al. (2011). Following that work, the Manning friction formula is used to determine the bed friction

$$C_h = g\frac{n_m^2}{H^{1/3}}, \tag{4.15}$$

where n_m is the Manning friction coefficient set to 0.01 $\mathrm{s\,m^{-1/3}}$. As we are using wetting-and-drying, we use the wet-dry model discussed in Section 2.2.2 and the simulation parameters are summarised in Table 4.1. Note that our model cannot currently simulate shoaling and breaking waves and thus, as we did for the test case in Section 2.5.1, we use a relatively high viscosity value of $\nu = 1\,\mathrm{m^2 s^{-1}}$ in the hydrodynamics to dissipate energy. This is standard practice as in, for example, Li and Huang (2013) who view viscosity as a model calibration parameter for energy dissipation, rather than a physical parameter.

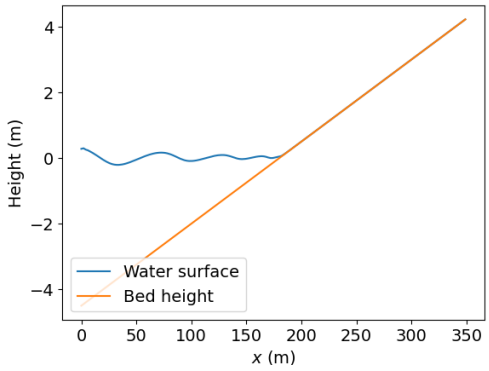


FIGURE 4.12. Initial bed profile of the Beach test case with an example of the simulated water surface.

Variable Name	Variable Value
Length in x-direction	350 m
Length in y-direction	10 m
Timestep, dt.	0.5 s
Bed slope gradient	1/40
Morphological simulation time	60 000 s
Morphological acceleration factor	250
Median particle size (d_{50})	1×10^{-4} m
Sediment density (ρ_s)	2650 $\mathrm{kg\,m^{-3}}$
Water density (ρ_f)	1000 $\mathrm{kg\,m^{-3}}$
Kinematic eddy viscosity (ν_0)	1 $\mathrm{m^2\,s^{-1}}$
Kinematic molecular viscosity (ν_m)	1×10^{-6} $\mathrm{m^2\,s^{-1}}$
Bed sediment porosity (p')	0.4
Diffusivity coefficient (ϵ_s)	1 $\mathrm{m^2\,s^{-1}}$
Manning friction coefficient (n_m)	0.01 $\mathrm{s\,m^{-1/3}}$
Wetting-and-drying parameter (δ)	0.05 m

TABLE 4.1. Parameter values for the Beach test case.

4.4.1 Fixed uniform meshes

We first consider a series of fixed meshes with uniform spacing varying from 700 mesh elements in the x-direction to 70 mesh elements, and choose an appropriate Δy to keep the elements in the fixed mesh approximately uniform. Note that we set $\Delta t = 0.5\,\mathrm{s}$ for all the simulations in this section to guarantee the wave is correctly defined. As this is a synthetic test case for which no experimental data exists, we use the model solution at 700 mesh elements in the x-direction and 20 mesh elements in the y-direction as a high resolution approximation of 'the truth' and are thus showing the discretisation error. Note that, for this section, the discretisation error is the pointwise ℓ^2 error norm at $y = 5\,\mathrm{m}$ at evenly spaced intervals. Figure 4.13 shows that the discretisation error decreases approximately linearly as the fixed uniform mesh becomes finer. For simple test cases, Kärnä et al. (2011) show that the wetting and drying scheme detailed in Section 2.2.2 results in an order of convergence of 1.5. However, our test case is more complex and includes coupling to morphology changes, and thus a further reduction in convergence order, as observed here, is to be expected.

4.4.2 Mesh movement

We apply mesh movement methods using the same monitor function (4.13) as that used in the previous test cases. Figure 4.14 shows an example of how the mesh is moved using this monitor function for this particular test case. Note that the mesh hardly moves in the y-direction because the bedlevel is

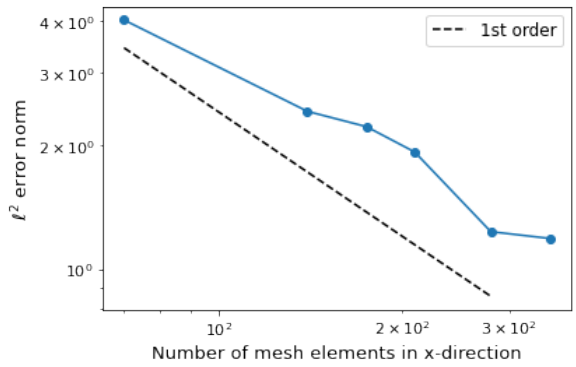


FIGURE 4.13. Discretisation error in the final bedlevel using a series of fixed uniform meshes in the Beach test case. The reference solution here is a high resolution solution simulated on a fixed uniform mesh with 700 elements in the x-direction. The discretisation error is the pointwise ℓ^2 error norm.

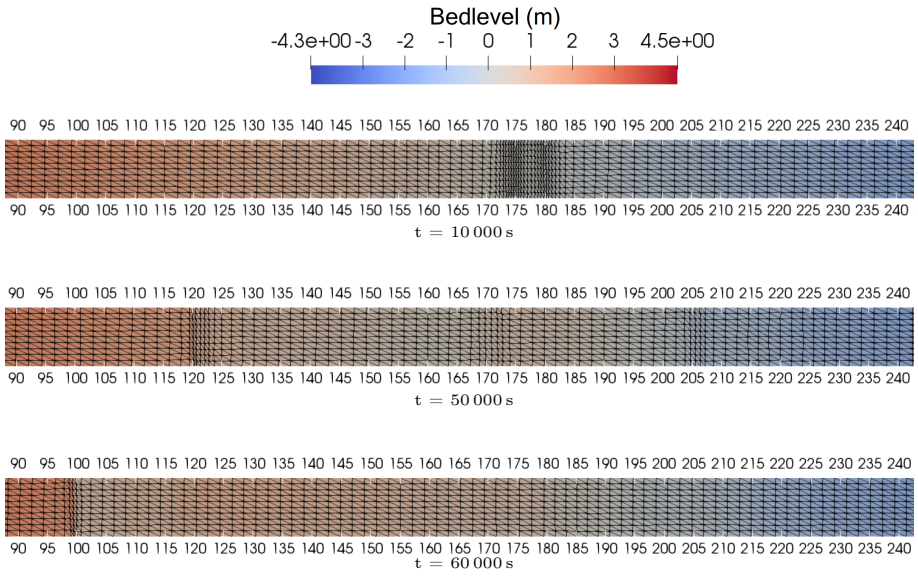


FIGURE 4.14. Snapshots of bedlevel and the underlying mesh for a moving mesh simulation of the Beach test case. The monitor function (4.13) with $\alpha = 5$ and $\beta = 0$ is applied to a mesh with 175 elements in the x-direction. Results are shown at three points in time demonstrating mesh movement to capture bed evolution to the right.

uniform in this direction. To understand the effect of this mesh movement, Figure 4.15 compares the final bedlevel obtained by the mesh movement shown in Figure 4.14 with the bedlevel obtained using a fixed uniform mesh with the same number of elements and the bedlevel from the high resolution solution (700 elements). This shows that using mesh movement allows us to capture the steep bed gradients more accurately than using a fixed uniform mesh of the same resolution.

As discussed in the previous sections, we can also choose the mesh movement frequency. This is particularly important in a test case with fast flowing hydrodynamics, like this one, and hence we conduct a sensitivity study considering multiple different mesh resolutions. Figure 4.16 shows that decreasing the

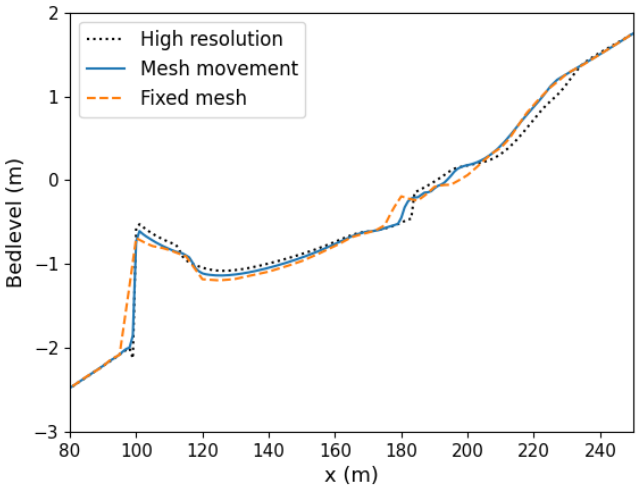
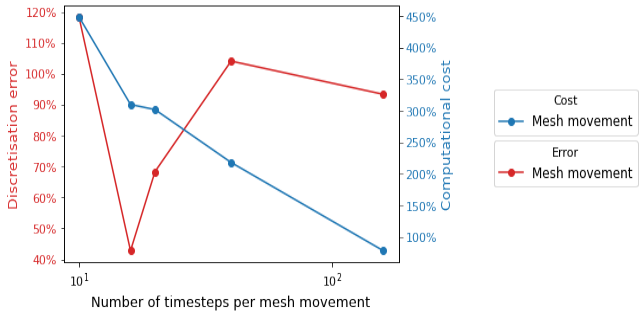


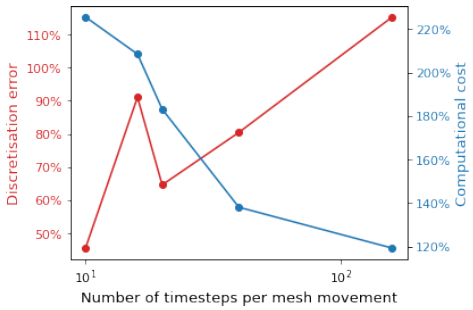
FIGURE 4.15. Comparison of final bedlevel resulting from fixed and moving mesh simulations of the Beach test case on a mesh with 175 elements in the x-direction. The moving mesh simulation applies the monitor function (4.13) with $\alpha = 5$ and $\beta = 0$. A high resolution solution (fixed uniform mesh with 700 elements in x-direction) is also shown.

number of timesteps per mesh movement almost always decreases the error and always increases the computational cost. Note that for all frequencies plotted, there is at least once mesh movement operation. For the smaller number of mesh elements considered (70 mesh elements in the x-direction), the error is minimised by using 16 timesteps per mesh movement, and increases when the number of timesteps is decreased beyond this point. We hypothesise that when there are fewer numbers of mesh elements, frequent mesh movement can result in numerical instabilities and, hence, larger errors. By contrast, for both larger numbers of mesh elements considered, the error is minimised by using 10 timesteps per mesh movement and is proportionally larger using 16 timesteps than at other similar frequencies. Thus, following our observations, in this test case a good compromise is to move the mesh every 10 timesteps for all mesh resolutions, apart from for 70 mesh elements when it is moved every 16 timesteps.

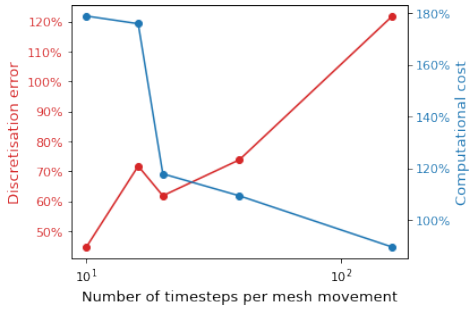
As in the previous test cases, we can also choose an α and β in (4.13) to minimise the discretisation error. We set $\beta = 0$ because the initial bed profile has a uniform slope (see Figure 4.12) and hence we expect that greater mesh refinement will be required in regions where the bed slope changes (*i.e.* where the second derivative is large). Given that in the previous sections we found that a parameter magnitude of either 3, for simple cases, or 5, for more complex ones, provides good general optimisation of the discretisation error, we conduct our sensitivity study using similar values. Figure 4.17 shows the results of this sensitivity study. Most significantly, the figure shows that mesh movement can more than half the error relative to the fixed uniform mesh error ($\alpha = 0$) for almost all mesh resolutions considered. Note that the exception is 280 elements, where error reduction is not as large because our model cannot produce a result for $\alpha > 3$ due to Courant number restrictions (recall from Section 2.2.3 that our model is not fully implicit). Additionally, Figure 4.17 shows that a general magnitude of 5 corresponds to either an optimum, or a very close to optimum, minimisation in the discretisation error for all mesh resolutions. This fits with our physical understanding of the problem because the presence of waves makes the problem



(A) 70 mesh elements in x-direction.



(B) 175 mesh elements in x-direction.



(C) 210 mesh elements in x-direction.

FIGURE 4.16. Trade-off between discretisation error and computational cost due to mesh movement frequency for the Beach test case. The monitor function (4.13) applied uses $\alpha = 5$ and $\beta = 0$ for all numbers of mesh elements. Note that for all frequencies plotted, at least one mesh movement occurs. Errors and times are expressed as percentages relative to the fixed mesh case with the same resolution.

more complex, thus requiring a strong mesh movement factor. Figure 4.18 emphasises the significant error reduction achieved by using mesh movement methods on this problem and, importantly, shows that for most mesh resolutions, the general parameter choice of $\alpha = 5$ and $\beta = 0$ results in similar errors to those for the optimum parameter values.

We have thus shown that, for this relatively complex wetting-and-drying test case, mesh movement methods can more than half the error at most mesh resolutions compared to a fixed uniform mesh, even when using general parameters that have not been tuned for this specific test case.

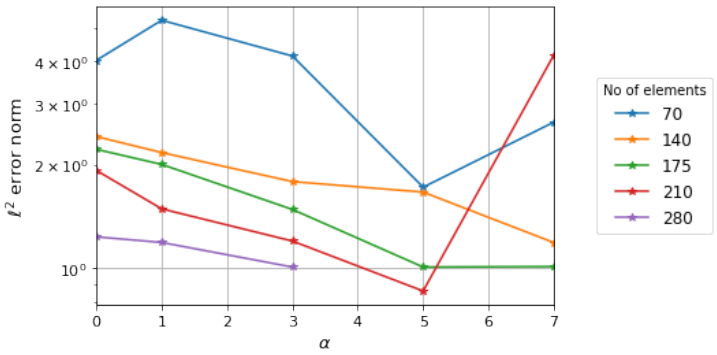


FIGURE 4.17. Discretisation error curves for moving mesh simulations of the Beach test case under different values of α in the monitor function (4.13).

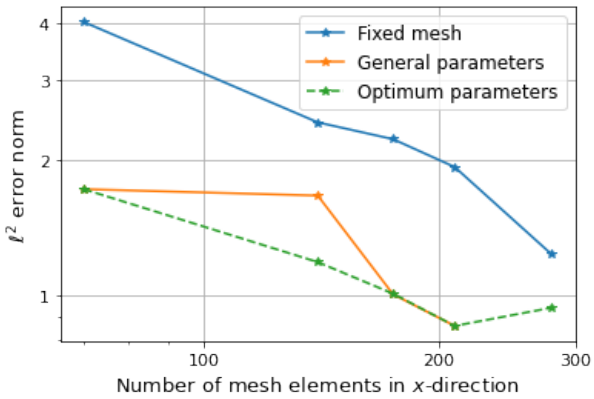


FIGURE 4.18. Discretisation error comparison for moving mesh and fixed mesh simulations of the Beach test case. Both general ($\alpha = 5, \beta = 0$) and optimum parameters for the monitor function (4.13) are considered.

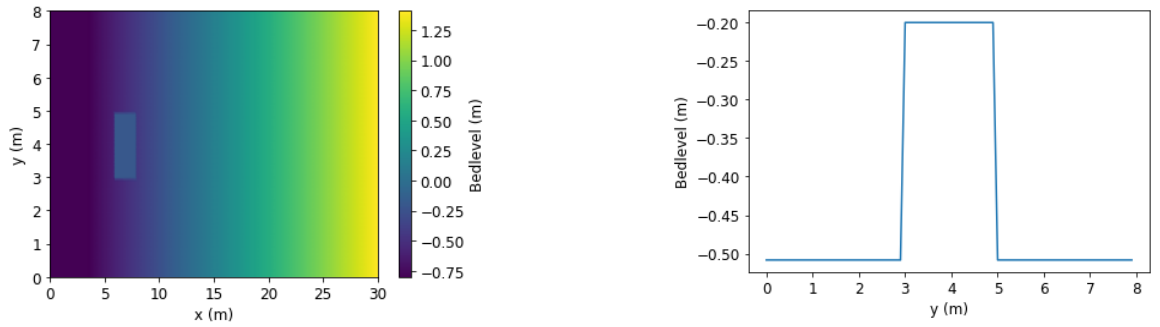
4.5 Complex bed profile with a wet-dry interface test case: Tsunami-like wave with an obstacle

As a final test of our moving mesh framework, we consider an example with both a wet-dry interface and a complex initial bed profile. So far in this chapter, we have mostly considered quasi-1D cases. Thus here we choose a more complex bed profile of a cube block obstacle on a sloping beach, inspired by the 2D test case in Hudson and Sweby (2005) and show the bed profile in Figure 4.19. To increase the complexity of this test case, we simulate a series of tsunami-like solitary waves breaking over this beach, inspired by the test case in Kobayashi and Lawrence (2004), which we validated for our model in Section 2.5.1. As discussed in that section, the hydrodynamic component of our model is not able to exactly represent the propagation of a solitary wave (Barthélemy, 2004; Kazhyken et al., 2021). However, our validation of the tsunami-like wave test case in that section suggests that using shallow water equations instead of a dispersive wave model does not result in substantial divergence from the experimental set-up. Moreover, the presence of the block obstacle means we are now considering a theoretical test case. Therefore, in this section, we are not comparing against experimental data but, instead, against internally consistent solutions obtained using the shallow water solver, meaning our mesh movement results are still valid.

In this section, we use the same parameters as in Section 2.5.1 (summarised in Table 2.8) and as before, simulate the incoming wave by imposing (2.94) as the free surface elevation boundary condition at the open boundary, which we recall here

$$\eta(t) = H_{\text{wave}} \operatorname{sech}^2\left(\sqrt{\frac{3H_{\text{wave}}}{4h}}\frac{\sqrt{g(H_{\text{wave}}+h)}}{h}(t - t_{\max})\right) + \eta_{\text{down}}. \tag{4.16}$$

Here H_{wave} is the average wave height, h the still water depth, $t_{\max}$ the arrival time of the wave crest and η_{down} the initial decrease of the elevation at the beginning of the simulation. Given we are not matching with experimental data and for reasons of time, we only run one wave in our simulation. However, following Section 2.5.1, we set a morphological acceleration factor of four, meaning that this one wave simulation is approximately equivalent to simulating four distinct waves. Recall also from the validation



(A) Planar view.

(B) Transect at $x = 7\,\mathrm{m}$.

FIGURE 4.19. Initial bed profile for the Tsunami with obstacle test case.

example in Section 2.5.1 that we only model suspended sediment transport in this test case because studies, such as Goto et al. (2011), show this is by far the most important type of sediment transport caused by tsunami-like waves.

4.5.1 Fixed uniform meshes

As with the other test cases, we begin by considering a series of fixed uniform meshes with 30, 60, 90, 120 and 150 mesh elements in the x-direction, corresponding to 8, 16, 24, 32, 40 mesh elements in the y-direction respectively, ensuring that the mesh elements are roughly uniform. Because we have combined two test cases to construct this test case, we no longer have experimental data available. Thus, we use the model solution at 600 mesh elements in the x-direction and 160 mesh elements in the y-direction as a high resolution approximation of 'the truth'. The discretisation error in this section is the L^2 error over the whole domain.

When we run our model on these fixed uniform meshes, we find that with only 30 mesh elements in the x-direction, our model crashes no matter how small a timestep is used. This is because the model cannot accurately simulate the movement of the tsunami-like wave along the slope at such a coarse resolution and, instead, unphysical shocks form that cannot be properly resolved. For the other fixed uniform meshes, Figure 4.20 shows that the discretisation error decreases approximately linearly as the number of mesh elements in the fixed uniform mesh increases, the same convergence order as for the wet-dry test case considered in Section 4.4.

4.5.2 Mesh movement

This test case is more complex than the ones considered previously because there are two regions with a potentially complex bed – the first is the block obstacle in the wave-approach and the second is where the wave breaks at the wet-dry interface. Thus, we add a component that tracks the wet-dry interface

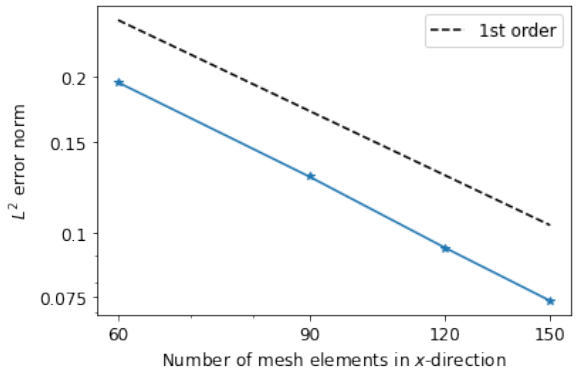


FIGURE 4.20. Discretisation error in the final bedlevel using a series of fixed uniform meshes in the Tsunami with obstacle test case. The reference solution here is a high resolution solution simulated on a fixed uniform mesh with 600 elements in the x-direction.

to the monitor function (4.13) used in previous test cases. This new monitor function is

$$m(x,y) = 1 + \mu\left(\max\left(\alpha\frac{\|\underline{\boldsymbol{H}}(z_b)\|_F}{\max\limits_{x,y}\|\underline{\boldsymbol{H}}(z_b)\|_F}, \beta\frac{\|\nabla z_b\|_2}{\max\limits_{x,y}\|\nabla z_b\|_2}\right) + \frac{\lambda}{\cosh(b_\lambda(\eta - z_b))^2}\right), \tag{4.17}$$

where η is the elevation, z_b is the bedlevel, b_λ controls the width of the wet-dry interface tracker and μ, α, β and λ are all user-defined parameters. Note that we set b_λ equal to 1 for all numbers of mesh elements except for 30 mesh elements, the smallest number considered, where we set b_λ equal to 5 to ensure that the wet-dry tracker still has an effect in this case. We choose to keep the wet-dry tracker outside of the maximum expression for the bedlevel derivatives because the bedlevel gradient and wet-dry interface are largely unrelated, hence their contributions in the monitor function should be separate

Figure 4.21 shows how a mesh with 120 mesh elements in the x-direction moves when the monitor function (4.17) is applied to it. The figure shows the mesh and the 3D view of the bedlevel separately, for clarity. The first- and second-order bedlevel derivatives in (4.17) cause the mesh to deform around the edges of the block obstacle in the wave-approach, whilst the wet-dry interface monitor tracks the movement of the wave up and down the slope. To emphasise this tracking movement, the wet-dry interface is plotted as a thick black line on Figure 4.21.

As with the previous test cases, we can choose how frequently the mesh is moved. Figure 4.22 shows that increasing the frequency of mesh movement decreases the discretisation error and increases the computational cost. We emphasise that for all frequencies plotted, there is at least one mesh movement operation. Whilst the cost of using mesh movement is always greater than that of using a fixed uniform mesh, mesh movement methods always result in an error that is less than half the size of that from the fixed uniform mesh. The accuracy improvement seems to plateau at around 20 timesteps per mesh movement and, therefore, this is the frequency with which we move the mesh in the remainder of this section. As the simulation time is 20 s and the timestep is $dt = 0.025$ s, this is equivalent to moving the mesh 40 times during the simulation. Figure 4.23 compares the final bedlevel obtained by moving the mesh at this frequency to the bedlevel obtained using a fixed uniform mesh with the same number of elements and the bedlevel from the high resolution solution (600 elements). It shows the mesh movement

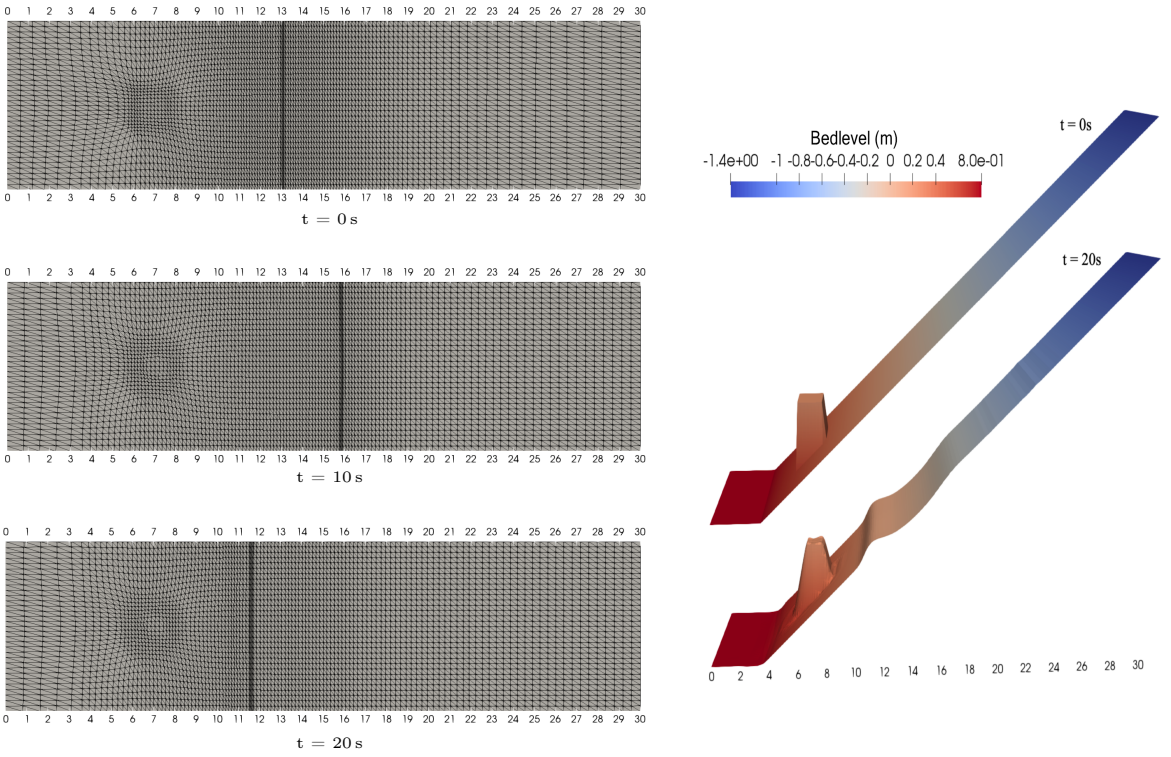


FIGURE 4.21. Snapshots of bedlevel and the underlying mesh for a moving mesh simulation for the Tsunami with obstacle test case. The monitor function (4.17) with $\mu = 15$ and $\alpha = \beta = \lambda = 1$ is applied to a mesh with 120 mesh elements in the x-direction. Mesh movement here captures bedlevel evolution to the right. The mesh is shown at three points in time with the wet-dry interface shown as a thick black line (LEFT) and the 3D form of the bed is shown at two points in time (RIGHT).

solution is much more accurate than the fixed mesh solution and, thus, that moving the mesh at this frequency is appropriate for this new more complex monitor function (4.17).

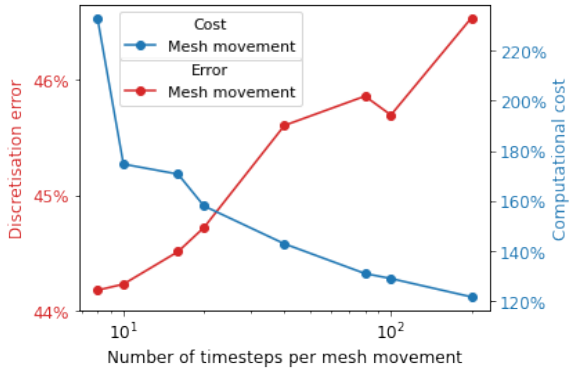


FIGURE 4.22. Trade-off between discretisation error and computational cost due to mesh movement frequency for the Tsunami with obstacle test case. The monitor function (4.17) with $\mu = 7$, $\alpha = 0$ and $\beta = \lambda = 1$ is applied to a mesh with 60 elements in the x-direction in the case. Note that for all frequencies plotted, at least one mesh movement occurs. Errors and times are expressed as percentages relative to the fixed mesh case with the same resolution.

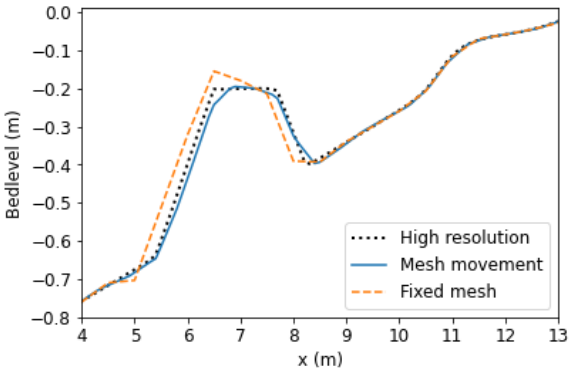


FIGURE 4.23. Comparison of final bedlevel resulting from fixed and moving mesh simulations of the Tsunami with obstacle test case on a mesh with 60 elements in the x-direction. The moving mesh simulation applies the monitor function (4.17) with $\mu = 7$, $\alpha = 0$, $\beta = \lambda = 1$ every 40 timesteps. A high resolution solution simulated on a fixed uniform mesh with 600 elements in the x-direction is also shown.

In this test case, we can also choose μ, α, β and λ to minimise the discretisation error. In previous test cases we found that a value equivalent to $\mu \approx 5$ provides a good general optimisation of the discretisation error and thus we conduct our sensitivity study using similar values. Figure 4.24 shows that again a magnitude of 5 provides a good general optimisation of the discretisation error. In all cases, the error is reduced by using mesh movement methods compared to the fixed uniform mesh ($\mu = 0$). Note that the fixed uniform mesh model crashes for 30 mesh elements in the x-direction, and hence no error is plotted on Figure 4.24 for $\mu = 0$ for this number of elements.

Figure 4.24 also shows the effect on the discretisation error of different relationships between α, β and λ in the monitor function (4.17). The largest errors almost always occur when the monitor function only includes the wet-dry interface tracker ($\lambda = 1, \alpha = \beta = 0$), which is understandable because the block obstacle in the wave-approach is not well-captured with this monitor function. However, in almost all cases, the inclusion of the wet-dry interface tracker with some combination of the first- and second-order derivative of the bedlevel results in a decrease in the discretisation error relative to the tracker not being present. In fact, it is only with the inclusion of the wet-dry interface tracker that our model can properly resolve the wave movement on the coarse mesh with only 30 elements in the x-direction. This highlights that appropriate mesh movement can not only decrease computational cost and improve accuracy, but also improve model stability and justifies the use of the more complex monitor function (4.17) in this test case. A good general choice for the relationship between the parameters is $\alpha = \beta = \lambda = 1$, *i.e.* an equal weighting is given to the first- and second-order bedlevel derivatives and the wet-dry interface tracker. This makes physical sense because the bedlevel derivatives are necessary to capture the obstacle correctly, whilst the interface tracker is necessary to capture the erosion and deposition caused by the

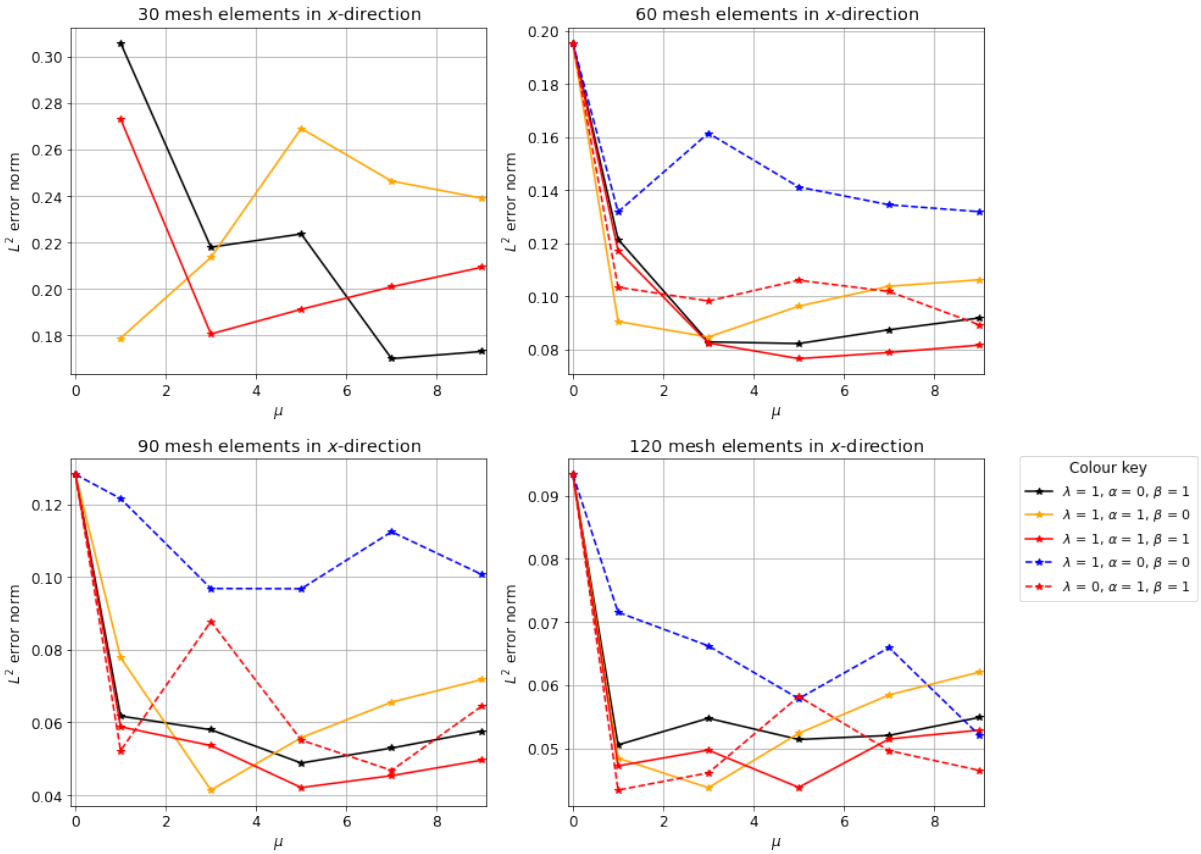


FIGURE 4.24. Discretisation error curves for moving mesh simulations of the Tsunami with obstacle test case under different values of μ, α, β and λ in the monitor function (4.17).

incoming wave. Using this general parameter choice, we can more than half our model error for the same number of mesh elements when compared to a fixed uniform mesh, which is a notable result.

We have thus shown that a good general parameter choice for this test case is $\mu = 5$ and $\alpha = \beta = \lambda = 1$, which is the same magnitude value as the complex test cases considered in Section 4.3.3 and 4.4, and the same order of magnitude as the simple test case in Section 4.3.2. Using these general parameters, in Figure 4.25 we compare simulation accuracy versus computational cost. The figure shows that using mesh movement methods results in both a significant improvement in accuracy and a significant reduction in computational cost, even when general parameters are used rather than optimum parameters. Note the optimum parameters plotted in this figure are the parameters that provide the smallest error, but not necessarily the fastest simulations, justifying why the general parameters perform better in the cost-to-accuracy ratio than the 'optimum' parameters. In many cases, it is possible to halve the discretisation error for the same computational cost, a notable improvement that is a particularly good result if we wish to run this test case more than once for calibration purposes. In addition, using mesh movement methods reduces the number of mesh elements required to achieve a good level of accuracy, thus reducing the memory costs of running the test case if the adjoint framework were applied to it (see Chapter 3).

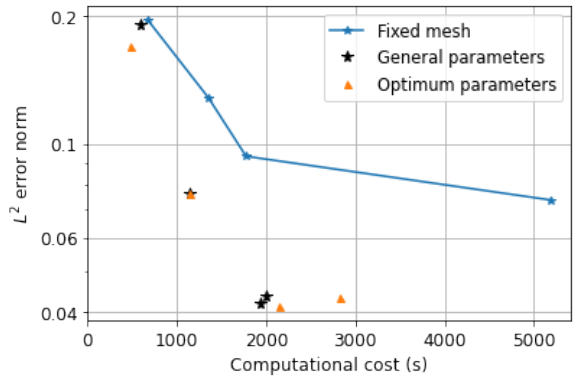


FIGURE 4.25. Computational cost vs discretisation error for both fixed mesh simulations and mesh movement simulations with general and optimum parameters, for the Tsunami with obstacle test case. Note that the different points correspond to different numbers of mesh elements.

Therefore, we have shown that for test cases with relatively complex bed profiles and a wet-dry interface, we can not only more than half the error for the same number of mesh elements, but also reduce computational cost and improve model stability, even when using general parameters that have not been tuned for this specific test case, a noteworthy result.

4.6 Conclusion

In this chapter we have implemented a mesh movement scheme as part of a hydro-morphodynamic model for the first time. We have shown that these mesh movement methods can be used to improve accuracy and decrease the computational cost of hydro-morphodynamic test cases with complex bed profiles and/or moving wet-dry interfaces. Moreover, we have demonstrated that mesh movement can also improve model stability, in certain cases. A highlight of this chapter is that these improvements are particularly significant for test cases with wet-dry interfaces, which are present in many coastal zone applications. For both the coastal zone test cases considered in this chapter, using mesh movement methods results in an error that is less than half the size of the error from the fixed uniform mesh with the same number of elements. This in turn leads to a reduction in computational and memory cost and, in future work, will mean more complex wet-dry test cases can be simulated accurately and efficiently, and also that model calibration can be more readily conducted, as more simulations can be performed for an equivalent cost.

For the mesh movement method considered, we present a monitor function for which the scaling factor that optimises the error reduction is fairly predictable from the physical characteristics of the test case. In particular, we have found that for 2D cases, and 1D cases with steep gradients, optimum mesh parameters always have a magnitude of approximately 5. These findings mean that this monitor function can be used on further problems without the need for a computationally expensive sensitivity study.

Code availability

The relevant code for the moving mesh hydro-morphodynamic model presented in this chapter is stored at https://doi.org/10.5281/zenodo.5155816.

CHAPTER 5

Applying the multilevel Monte Carlo method to assess erosion and flood risk in the coastal zone

Highlights

- First application of the *multilevel* Monte Carlo method (MLMC) to a hydro-morphodynamic model;
- MLMC leads to a large drop in computational cost for the same statistical accuracy compared to standard Monte Carlo methods;
- Informed decision making aided by using MLMC to estimate the cumulative distribution of output variables.

Summary

Hydro-morphodynamic models can be difficult to use in practice because they are subject to a large degree of uncertainty and are also computationally expensive. In this chapter, we present the novel application of the *multilevel* Monte Carlo method (MLMC) to a hydro-morphodynamic coastal ocean model, XBeach, to quantify uncertainty by computing statistics of key output variables. For both theoretical and real-world coastal zone case studies, we show that MLMC can significantly reduce computational cost, resulting in speed-up factors of 40 or greater compared to a standard Monte Carlo approach, whilst keeping the same statistical accuracy. Furthermore, a sophisticated ensemble generating technique is used to estimate the cumulative distribution functions of variables from the MLMC output. These functions are of significant value to decision makers assessing risk in the coastal zone.

Publications and Disclaimer

The work in this chapter has been submitted for publication to *Coastal Engineering*, where it is currently under review. The preprint of this paper is available here:

Clare, M. C., Piggott, M., & Cotter, C. J. (2021). 'Assessing erosion and flood risk in the coastal zone through the application of the multilevel Monte Carlo method.' https://eartharxiv.org/repository/view/1956/.

5.1 Introduction

Hydro-morphodynamic models are known to have a large degree of uncertainty associated with them, both due to incomplete knowledge and natural variability in the system. This uncertainty can be quantified by combining numerical hydro-morphodynamic models with statistical frameworks.

In this chapter, we explore statistical uncertainty quantification techniques using the depth-averaged finite volume based hydro-morphodynamic coastal ocean model XBeach (Roelvink et al., 2009), described in Section 2.3.2. We choose this model, rather than our own hydro-morphodynamic model developed in Chapters 2-4, because of XBeach's ability to parameterise short wavelength wind-driven wave propagation, which is required in order to simulate the complex wind-wave spectra in the real-world studies considered here. XBeach has been successfully applied to simulate complex flows, sediment transport, and morphodynamic changes (see for example McCall et al., 2010; Riesenkamp, 2011; Callaghan et al., 2013; Roelvink et al., 2018; de Beer et al., 2020). However, like all complex hydro-morphodynamic models, XBeach is computationally relatively expensive and, thus, does not lend itself well to Monte Carlo simulations and hazard assessments, which require a large number of individual model runs. For example, in Harris et al. (2018), a Monte Carlo simulation with 240,000 individual runs of the 1D version of XBeach is required to perform the desired study. This computational cost has limited the scenarios that Monte Carlo based studies can be used to address, and has meant that, for more complex test cases, researchers have had to use simplified models. For example, Callaghan et al. (2013) use the simpler semi-empirical model SBeach (Larson and Kraus, 1989) in 1D for their Monte Carlo simulation which takes 40 days to run, and they estimate that even using the 1D version of XBeach would take four and a half millennia (on a single processor).

Rather than reverting to simpler, less costly models, we apply the relatively novel *multilevel* Monte Carlo method (MLMC), first presented in Giles (2008), to XBeach. MLMC seeks to reduce computational cost by accelerating the Monte Carlo approach through the use of a hierarchy of model configurations, each with a different level of resolution. Like the standard Monte Carlo method, MLMC quantifies uncertainty by computing estimators of the expected value of key output variables with respect to uncertain input parameters. We emphasise here that the purpose of our work is to use MLMC to simulate statistics on a fine resolution grid in XBeach more efficiently than the standard Monte Carlo method. The difference in the final Monte Carlo and MLMC estimators is the statistical (sampling) error – they will have approximately the same model and discretisation errors because in our comparisons the Monte Carlo method resolution is the same as the finest MLMC resolution. As we are mostly considering cases that have already been validated in XBeach and set the finest level to be the resolution considered in the evaluation, we assume that the model and discretisation errors are acceptable for practitioners.

In this chapter, we present an exploration of how MLMC can be applied to the hydro-morphodynamic model XBeach to investigate the impact of a variety of uncertain input parameters in both theoretical and real-world test cases. To the best of our knowledge, this is the first application of MLMC with complex hydro-morphodynamic models. It has been used in areas as diverse as mathematical finance (Giles, 2008),

quantum mechanics (Jansen et al., 2020) and atmospheric dispersion modelling (Katsiolides et al., 2018). The most comparable application of MLMC to that in this chapter is in Sánchez-Linares et al. (2016), where MLMC is successfully applied to estimate uncertainty in landslide generated tsunamis using a simple shallow-water like model.

A potential issue when applying MLMC successfully to coastal problems is that decision makers are not only interested in the expected value of a variable, but also in the probability/risk of a variable exceeding a certain value. This probability can be expressed as the expectation, $\mathbb{E}[\mathbb{1}_{X\geq x}]$, where $\mathbb{1}$ is the indicator function and, for example, X represents the maximum horizontal inundation distance and x the location of a physical structure. However as discussed in Giles (2015), MLMC struggles with binary output variables like $\mathbb{1}_{X\geq x}$, because a large number of samples is required to ensure accurate variance estimates. A number of different methods have been proposed to deal with this issue, including smoothing (Giles et al., 2015) and selective refinement (Elfverson et al., 2016). In this chapter, the ensemble generating method from Gregory and Cotter (2017) is used (see Section 5.4). Unlike other approaches, this method generates the entire cumulative distribution function for the output variable and does so in a simple and computationally efficient manner.

The remainder of this chapter is structured as follows: in Section 5.2 we outline the relevant MLMC theory; in Section 5.3, we integrate MLMC with XBeach and run a series of theoretical and real-world test cases; in Section 5.4, we estimate the cumulative distribution function of the output variables using MLMC; and finally, in Section 5.5, we present conclusions from this chapter.

5.2 Multilevel Monte Carlo Method (MLMC)

In this section, following Giles (2008) and Giles (2015), we discuss the *multilevel* Monte Carlo method (MLMC) and its use in computing estimators for the expectation of discretised random variables given uncertain input parameters. Whereas the Monte Carlo method considers the model at just one resolution, MLMC accelerates the Monte Carlo method by using a hierarchy of models at different levels of resolution. The fundamental idea underlying MLMC comes from the linearity of expectations: for two variables X_l and X_{l-1},

$$\mathbb{E}[X_l] = \mathbb{E}[X_{l-1}] + \mathbb{E}[X_l - X_{l-1}], \tag{5.1}$$

where $\mathbb{E}[\cdot]$ denotes the expectation and extending this idea

$$\mathbb{E}[X_L] = \mathbb{E}[X_{l_\mu}] + \sum_{l=l_\mu+1}^{L} \mathbb{E}[X_l - X_{l-1}]. \tag{5.2}$$

In the MLMC framework, X_l denotes the numerical approximation to X on level l of the multilevel environment and, thus, X_{l_μ} and X_L denote the approximation on the coarsest and finest level of resolution, respectively. Within this framework, each level l is required to have the numeric mesh element size

$$h_l = M^{-l}T, \tag{5.3}$$

where T is the total length of the domain and M is the integer factor the mesh element size is refined by at each level (following standard practice, we use $M = 2$ throughout). Thus, the mesh becomes more refined as l increases. Note that the domain can be multi-dimensional, in which case there is an h_l for every dimension of T.

The expectation estimator is then defined as

$$\hat{Y} = \sum_{l=l_\mu}^{L} \hat{Y}_l, \tag{5.4}$$

where, throughout this thesis, we use the standard notation, $\hat{\cdot}$, to denote an estimator. Here $\hat{Y}_l$ is the estimator for $\mathbb{E}[X_l - X_{l-1}]$ defined as

$$\hat{Y}_l = \begin{cases} N_{l_\mu}^{-1} \sum_{i=1}^{N_{l_\mu}} X_{l_\mu}^{(i)} & l = l_\mu, \\ N_l^{-1} \sum_{i=1}^{N_l} \left(X_l^{(i)} - X_{l-1}^{(i)} \right) & l > l_\mu, \end{cases} \tag{5.5}$$

where N_l is the number of samples at each level pair $(l,\ l-1)$ and N_{l_μ} is the number of samples at the coarsest level l_μ. Here, X_l and X_{l-1} are on the same Brownian path, *i.e.* we use the same random numbers to construct the variables X_l and X_{l-1} to ensure strong convergence ($\mathbb{E}[|X_l - X_{l-1}|]$ as the grid is refined). To ensure independence between the estimators at each level, different independent samples are used at each level. Thus, $\mathrm{Cov}(\hat{Y}_i, \hat{Y}_j) = 0$ if $i \neq j$ and the variance formula can be simplified to

$$\mathrm{Var}[\hat{Y}] = \mathrm{Var}\left(\sum_{l=l_\mu}^{L} \hat{Y}_l \right) = \sum_{l=l_\mu}^{L} N_l^{-1} V_l, \tag{5.6}$$

where $\mathrm{Var}[\cdot]$ denotes the variance and V_l is the variance of $X_l^{(i)} - X_{l-1}^{(i)}$. The overall cost of calculating the expectation estimator (5.4) is given by

$$C = \sum_{l=l_\mu}^{L} N_l C_l, \tag{5.7}$$

where C_l is the cost of a model run at level l. Finally, the root mean square error (RMSE) of this MLMC estimator (5.4) is

$$\mathrm{RMSE} = \sqrt{\mathbb{E}[(\hat{Y} - \mathbb{E}[X])^2]}, \tag{5.8}$$

which can be simplified by noting that $\hat{Y}$ is an unbiased estimator of $\mathbb{E}[X_L]$ (see Giles, 2015), and so

$$\mathrm{RMSE} = \sqrt{\mathbb{E}[(\hat{Y} - \mathbb{E}[X_L])^2] + (\mathbb{E}[X_L] - \mathbb{E}[X])^2}. \tag{5.9}$$

The first term in (5.9) is the variance of the estimator $\hat{Y}$ and is the Monte Carlo statistical sampling error; and the second term is the square of the bias error and is the numerical discretisation error.

With these definitions, we can now outline the main complexity theorem for MLMC given in Giles (2015):

Theorem 1. *Let X denote a random variable and X_l be the corresponding approximation at level l. If there exist independent estimators $\hat{Y}_l$ based on N_l samples, $\alpha \geq \frac{1}{2}\min(\beta,\gamma)$, and β, γ, c_1, c_2, c_3 are positive constants such that*

(i) $|\mathbb{E}[X_l - X]| \leq c_1 2^{-\alpha l}$,

(ii) $\mathbb{E}[\hat{Y}_l] = \begin{cases} \mathbb{E}[X_{l_\mu}], & l = l_\mu, \\ \mathbb{E}[X_l - X_{l-1}], & l > l_\mu, \end{cases}$

(iii) $Var[X_l - X_{l-1}] \leq c_2 2^{-\beta l}$,

(iv) $C_l \leq c_3 2^{\gamma l}$,

then $\exists\ c_4 > 0$ such that for any $\epsilon < e^{-1}$, there are L and N_l such that $\hat{Y} = \sum_{l=l_\mu}^{L} \hat{Y}_l$ has RMSE such that $RMSE \leq \epsilon$ and overall computational complexity

$$C \leq \begin{cases} c_4 \epsilon^{-2}, & \beta > \gamma, \\ c_4 \epsilon^{-2} (\log \epsilon)^2, & \beta = \gamma, \\ c_4 \epsilon^{-2-(\gamma-\beta)/\alpha}, & 0 < \beta < \gamma. \end{cases} \tag{5.10}$$

Recall that C here is the overall computational cost given by (5.7). The proof of this theorem is given briefly in Giles (2015) and is similar to one given in detail in Cliffe et al. (2011). We note that a key criterion for the proof of the theorem is that for the maximum level L,

$$L = \lceil \alpha^{-1} \log_2(\sqrt{2} c_1 \epsilon^{-1} \rceil. \tag{5.11}$$

Note that this is enforced implicitly by the convergence condition in step 6 of Algorithm 1 but can be reinforced by choosing $L_{\max} = L$.

In the theorem, (i), (iii) and (iv) represent a bound on the bias, variance and cost, respectively, at each level, and that (ii) follows from the definition in (5.5). As the mesh is refined, β controls the decay of the variance and γ the growth of the cost. Thus, for $\beta > \gamma$, most of the computational cost is at the coarser levels; whereas for $\beta < \gamma$, most of the computational cost is at the finest levels; and finally for $\beta = \gamma$, the computational cost is evenly split between the levels (Cliffe et al., 2011). In this chapter, we find the values of α, β and γ by estimating the gradient of the lines of best fit for $\hat{Y}_l$ (the estimator for $\mathbb{E}[X_l - X_{l-1}]$) versus l for α, $\mathrm{Var}[X_l - X_{l-1}]$ versus l for β and C_l (timed cost) versus l for γ.

A key part of MLMC is determining N_l, the optimum number of samples required at each level l to achieve the convergence stated in Theorem 1. Following Giles (2008), the optimum value of N_l is determined by minimising the overall variance (*i.e.* the statistical sampling error) with respect to the fixed overall computational cost $\sum_{l=l_\mu}^{L} N_l C_l = c$. In mathematical terms, this is equivalent to finding the stationary

points of

$$S(N_l, \lambda) = \sum_{l=l_\mu}^{L} \frac{V_l}{N_l} + \lambda \left(\sum_{l=l_\mu}^{L} N_l C_l - c \right), \tag{5.12}$$

with respect to N_l, where λ is a Lagrange multiplier and recall $V_l = \mathrm{Var}[X_l - X_{l-1}]$. Differentiating and setting this expression to zero, we find

$$N_l = \frac{\sqrt{V_l}}{\lambda \sqrt{C_l}}. \tag{5.13}$$

To find the number of samples required to achieve a maximum statistical error of $\epsilon^2/2$, we set the expression for the variance (5.6) equal to this tolerance and substitute in (5.13) to find

$$\frac{1}{\lambda} = 2\epsilon^{-2} \sum_{l=l_\mu}^{L} \sqrt{V_l C_l}. \tag{5.14}$$

Hence the number of samples required is

$$N_l = \left\lceil 2\epsilon^{-2} \sqrt{\frac{V_l}{C_l}} \left(\sum_{l=l_\mu}^{L} \sqrt{V_l C_l} \right) \right\rceil, \tag{5.15}$$

where $\lceil \cdot \rceil$ is the ceiling operator which returns the smallest integer greater than or equal to the quantity inside it.

The formula (5.15) is dependent on V_l and C_l, but it can be expensive to obtain good estimates for these values at finer levels and often requires oversampling. Therefore, following the proof in Cliffe et al. (2011), we instead use the conditions in Theorem 1 to create an upper bound on N_l using, for example, that (iii) and (iv) mean $V_l C_l < c_2 c_3 2^{(\gamma-\beta)l}$. Thus

- if $\beta > \gamma$:

$$N_l = \left\lceil 2\epsilon^{-2} c_2 \left(1 - 2^{-(\beta-\gamma)/2}\right)^{-1} 2^{-(\beta+\gamma)l/2} \right\rceil, \tag{5.16}$$

- if $\beta < \gamma$:

$$N_l = \left\lceil 2\epsilon^{-2} c_2 2^{(\gamma-\beta)L/2} \left(1 - 2^{-(\gamma-\beta)/2}\right)^{-1} 2^{-(\beta+\gamma)l/2} \right\rceil, \tag{5.17}$$

- if $\beta = \gamma$:

$$N_l = \left\lceil 2\epsilon^{-2} (L+1) c_2 2^{-\beta L} \right\rceil, \tag{5.18}$$

are the expressions that we use for the optimum number of samples at each level in MLMC.

5.2.1 MLMC verification checks

As shown above, MLMC is complex and therefore we need the following checks, detailed in Giles (2015), to ensure that the method implementation is both correct and appropriate:

- **Convergence check:** The convergence condition (i) in Theorem 1 states that $\mathbb{E}[X_l - X_{l-1}] \propto 2^{-\alpha l}$ and, thus, the bias $\mathbb{E}[X - X_L] = \frac{\mathbb{E}[X_L - X_{L-1}]}{2^\alpha - 1}$. We can therefore use the following as a convergence test

$$\frac{|\mathbb{E}[X_L - X_{L-1}]|}{2^\alpha - 1} < \frac{\epsilon}{\sqrt{2}}. \tag{5.19}$$

  If this check fails, this means MLMC has not converged to within the specified tolerance, either because of a mathematical or programming error, or simply because the value of ϵ chosen is too small for the number of levels in the MLMC algorithm;

- **Consistency check:** If $\mathbb{E}[\hat{Y}_l]$ is consistent, then $\mathbb{E}[\hat{Y}_l] - \mathbb{E}[X_l] + \mathbb{E}[X_{l-1}] = 0$ is true, with a small allowance for error due to Monte Carlo sampling. Furthermore, the variances should obey $\sqrt{\mathrm{Var}[X_{l-1} - X_l + \hat{Y}_l]} \leq \sqrt{\mathrm{Var}[X_{l-1}]} + \sqrt{\mathrm{Var}[X_l]} + \sqrt{\mathrm{Var}[\hat{Y}_l]}$. Giles (2015) show that if these two statements are true, the following ratio

$$\frac{|\mathbb{E}[\hat{Y}_l] - \mathbb{E}[X_l] + \mathbb{E}[X_{l-1}]|}{3\left(\sqrt{\mathrm{Var}[\hat{Y}_l]} + \sqrt{\mathrm{Var}[X_l]} + \sqrt{\mathrm{Var}[X_{l-1}]}\right)}, \tag{5.20}$$

  has a probability of 0.3% of being greater than 1. Therefore, if this ratio is greater than 1, the check fails indicating either a mathematical or programming error;

- **Kurtosis check:** The kurtosis, κ, of $X_l - X_{l-1}$ determines the number of samples required for a good variance estimate. If κ is very large, this indicates that the variance estimate is poor and that either the number of samples used is insufficient or the method chosen for sampling is incorrect. Thus, following standard practice, if the kurtosis is greater than 100, an error is raised.

We use these checks to evaluate the coding and mathematical implementation of the test cases in this chapter and to identify any inherent problems with applying MLMC to these test cases.

5.2.2 MLMC algorithm

We conclude this section by stating Algorithm 1, the full MLMC algorithm used in this chapter.

Algorithm 1: Multilevel Monte Carlo Method.

1. Estimate the variance and expectation at user-specified levels (and hence α, β and γ) using an initial estimate for the number of samples
2. Start with $L = l_\mu$
3. Define optimal N_l for using (5.16) if $\beta > \gamma$, (5.17) if $\beta < \gamma$ and (5.18) if $\beta = \gamma$
4. If the optimal N_l is greater than the number of samples already evaluated, evaluate the extra samples needed
5. If $L \geq 2$, test for convergence, *i.e.* check if (5.19) is satisfied
6. If the algorithm has not converged and $L < L_{\max}$, set $L := L + 1$ and return to Step 3
7. If algorithm converged or $L \geq L_{\max}$, STOP

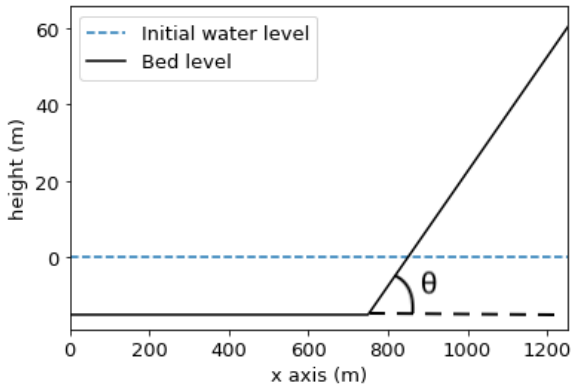


FIGURE 5.1. Schematic of bed with uncertain bed slope angle θ for 1D bed slope test case.

5.3 Applying MLMC to XBeach

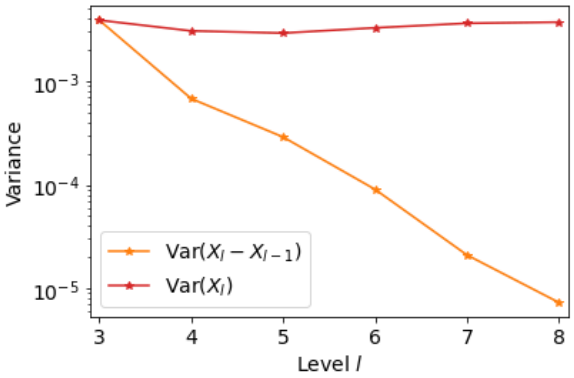
Now that we have outlined the theory behind MLMC, we can implement the algorithm with the coastal ocean model XBeach in order to assess uncertainty in both theoretical and real-world test cases. Recall that a description of XBeach is given in Section 2.3.2. To implement MLMC, we construct our own Python MLMC wrapper around the XBeach model. Our wrapper is written in such a way that it can easily be shared on distributed computational cores of an HPC cluster, thus increasing the efficiency of the algorithm. Hence, throughout this section, when we refer to time, we mean the total time the simulation takes to run multiplied by the number of cores used. Note that our MLMC wrapper is fully flexible meaning it can be easily applied to other models in further work.

5.3.1 Uncertainty in bed slope angle

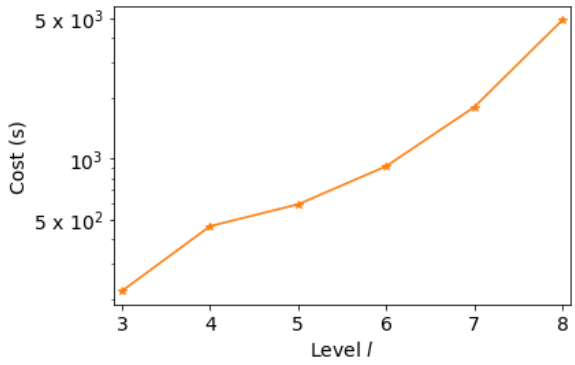
For our first set of test cases for the MLMC-XBeach framework, we evaluate the uncertainty associated with the beach bed profile by considering the bed slope angle θ to be uncertain. We choose this field because beach profiles represent a significant source of uncertainty, as discussed in Unguendoli (2018).

5.3.1.1 One-dimensional (1D) bed slope test case

We first consider the simple 1D hydrodynamics-only problem of a stationary wave approaching a beach with a wave period of 10 s and a maximum wave amplitude of 2.5 m, for which an example bed set-up is depicted in Figure 5.1. Each simulation is run for 200 s and is simulated using the stationary mode in XBeach (see Section 2.3.2). The distribution of the bed slope angle θ is set to be $\theta \sim \mathcal{U}(\arctan(0.035), \arctan(0.5))$, where the lower bound is chosen to ensure that there is always an area of the bed which is above the initial water elevation height. During the simulation, we seek the expected value of the maximum horizontal inundation distance. In order to satisfy the constraints on ϵ required to achieve the convergence criteria stated in Theorem 1, the inundation distance is normalised by dividing it by the total x-length of the domain (1250 m).



(A) Variance of difference, $X_l - X_{l-1}$, compared to variance of single variable, X_l at each level, l.



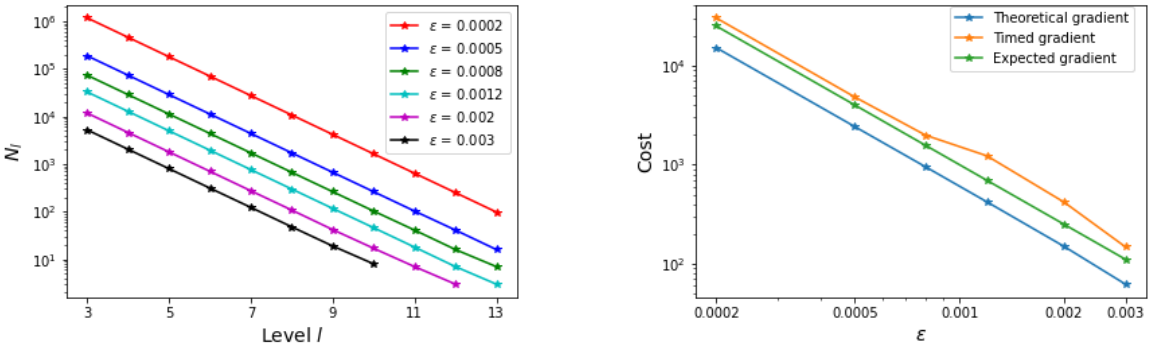
(B) Increasing computational cost (s) as the mesh is refined (*i.e.* l increases).

FIGURE 5.2. Preliminary MLMC results for the 1D bed slope test case showing the use of MLMC is justified.

Before running the full MLMC algorithm (Algorithm 1), we first run Step 1, in order to check that the MLMC approach is appropriate for this test case. We use six levels ($l = 3$ to 8) with a grid-size of $\Delta x = 1250/2^l$. In Figure 5.2a, the log plot of $\text{Var}[X_l - X_{l-1}]$ versus level shows that the variance decreases as the level number l increases, *i.e.* as the mesh gets finer. This observation is important because, recalling (5.6), this means fewer samples are needed on the finer levels. The figure also shows that $\text{Var}[X_l - X_{l-1}]$ is lower than $\text{Var}[X_l]$ for all l, another notable result because otherwise a lower variance could be achieved by using the same number of samples with a standard Monte Carlo simulation. Thus, we expect the MLMC simulation to be less computationally expensive than a Monte Carlo simulation, meaning using MLMC for this test case is justified.

Figure 5.2b shows that the cost (calculated by timing the code) increases as the level number l increases. This is expected because a fixed number of sample points is being used for the preliminary MLMC run and, at finer levels, XBeach is more computationally expensive. Using this figure and Figure 5.2a, as well as the log relationship between $\hat{Y}_l$ and level l, convergence tests estimate the parameter values defined in Theorem 1 as $\alpha = 1.29; \beta = 1.89; \gamma = 1.10$ for this test case. Thus, the numerical discretisation is approximately first-order accurate (based upon the value of α) and, as $\beta > \gamma$, most of the computational cost in the MLMC algorithm comes from the coarser levels. From Theorem 1, this means that for a fixed finest resolution, the overall cost C is proportional to ϵ^{-2}, which is the same as for Monte Carlo at a fixed resolution, but we still expect MLMC to be computationally faster due to the improvements in the variance Cliffe et al. (2011).

These observations from the preliminary study, as well as the fact that the consistency and kurtosis checks from Section 5.2.1 passed, means we can conclude that MLMC can be used successfully with XBeach and we can run the full algorithm. For this test case, the chosen range of tolerance values are $\epsilon = [0.0002, 0.0005, 0.0008, 0.0012, 0.002, 0.003]$, recalling that ϵ controls the bound on the convergence test (5.19). As $\beta > \gamma$, we use these ϵ values in the formula (5.16) to derive the optimum number of samples required at each level, N_l. We also set a maximum limit on the finest level that l can reach of



(A) Optimum number of samples required at each level l for given tolerance value ϵ.

(B) Verification that both theoretical cost (5.21) and timed cost vary with ϵ at the expected rate.

FIGURE 5.3. Effect of tolerance value ϵ on cost and sample size for the 1D bed slope test case.

$L_{\max} = 13$ using (5.11) (see Step 7 of Algorithm 1). Figure 5.3a shows the optimal number of samples N_l needed and, most importantly, that as the level number increases (*i.e.* as the resolution becomes finer), fewer samples are needed until at the finest level only approximately 100 samples are required. Thus, MLMC is more efficient than a standard Monte Carlo method which later in this section we show requires at least 700,000 samples at the finest level.

From these N_l, we use the formula given in Giles (2008)

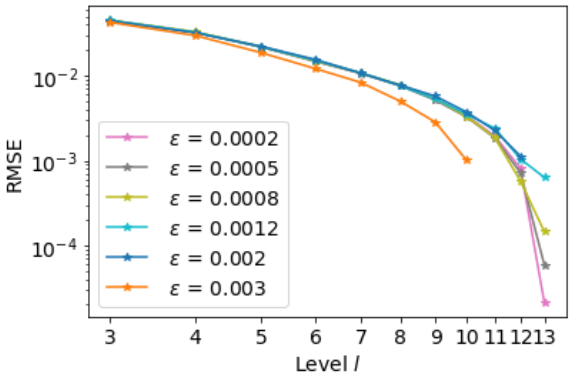
$$C = \left(1 + \frac{1}{2}\right) \sum_{l=l_\mu}^{L} N_l \, 2^{\gamma(l+1)}, \tag{5.21}$$

to calculate the theoretical cost C of the simulation for each ϵ. Figure 5.3b shows that both the gradient of the theoretical cost and the timed cost agree with the expected gradient $C \propto \epsilon^{-2}$, hence the code has been correctly applied.

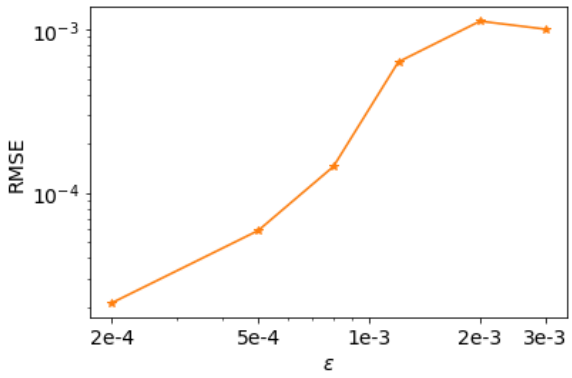
To test the accuracy of the MLMC estimate, we also estimate the 'true' solution using a Monte Carlo simulation at the finest mesh considered by the MLMC simulation ($\Delta x = 0.153\,\text{m}$). According to that Monte Carlo simulation, the expected maximum horizontal inundation distance divided by the total domain length is 0.6785. As the total domain length is 1250 m, this means the water inundates a horizontal distance of 848.1 m. We calculate the root mean square error of the MLMC simulation compared to the Monte Carlo simulation using

$$\text{RMSE of MLMC} = \sqrt{\left(\hat{Y} - \hat{X}_{MC}\right)^2}, \tag{5.22}$$

where $\hat{X}_{MC}$ denotes the Monte Carlo estimator for X and recall that $\hat{Y}$ is the MLMC estimator, defined in (5.4). This RMSE is a measure of the difference in the statistical error, as the finest mesh considered by MLMC is the mesh used for the Monte Carlo estimator meaning that the discretisation and model error of the two estimators are approximately the same. Figure 5.4a shows that as the mesh becomes finer, the RMSE decreases for all ϵ values considered. In addition, Figure 5.4b shows that, as ϵ decreases, the final expected value generally becomes closer to the 'true' solution, showing that the convergence



(A) RMSE of MLMC at each level l for given tolerance value ϵ.



(B) RMSE of MLMC once it has converged for each tolerance value ϵ.

FIGURE 5.4. RMSE of MLMC (5.22) for the 1D bed slope test case for varying tolerance value, ϵ.

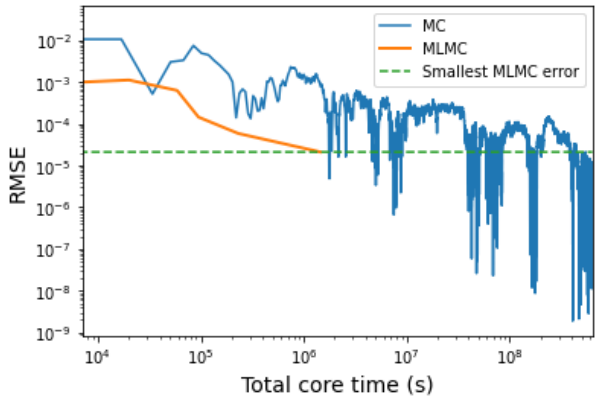


FIGURE 5.5. Comparison of total cost and statistical accuracy when using MLMC and Monte Carlo (MC) for the 1D bed slope test case.

condition (5.19) is working as expected. Note that the only exception to this trend is between $\epsilon = 0.002$ and $\epsilon = 0.003$. This is likely because, at these values of ϵ, the optimal number of samples is quite low (see Figure 5.3a) leading to less accurate variance approximations.

Finally, Figure 5.5 compares the total cost and statistical accuracy for MLMC versus Monte Carlo. Most significantly, as summarised in Table 5.1, MLMC achieves the same accuracy as Monte Carlo for 1/377 (0.26%) of the computational cost, a significant reduction. Figure 5.5 also shows that MLMC convergence is much more uniform than Monte Carlo convergence due to the use of the convergence test in Section 5.2.1 and the optimum number of samples calculation. For Monte Carlo, we stop the simulation once the Monte Carlo RMSE is below the MLMC RMSE for over 150,000 samples, but for MLMC we do not need to conduct further samples to check whether the algorithm has really converged. This decreases the real overall computational cost further.

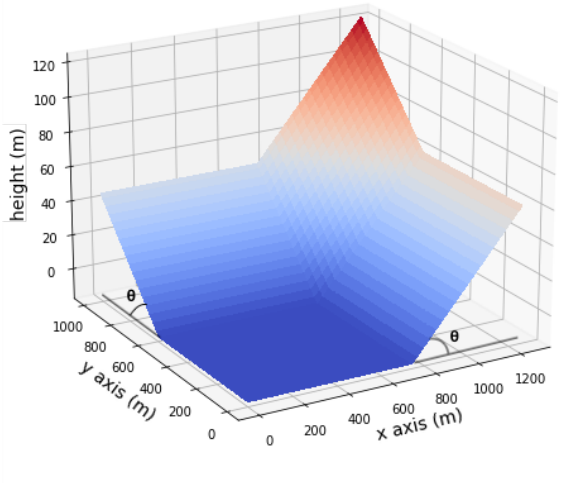


FIGURE 5.6. Schematic of 2D bed with uncertain bed slope angles θ for the 2D bed slope test case.

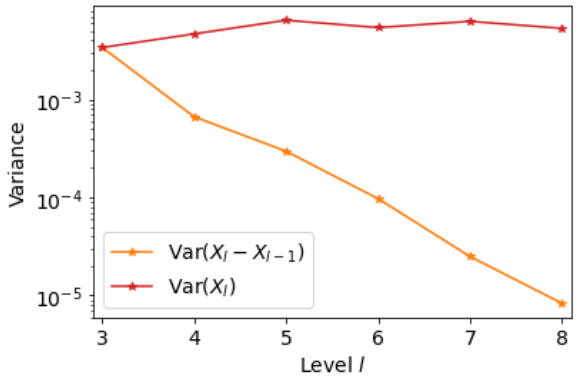


FIGURE 5.7. Variance of difference, $X_l - X_{l-1}$, compared to variance of single variable, X_l, at each level l for the 2D bed slope test case, showing the use of MLMC is justified.

	MLMC	**Monte Carlo**
RMSE	2.12×10^{-5}	2.12×10^{-5}
Time to achieve RMSE (hours)	402	151,736
MLMC speed-up factor	**377**	

TABLE 5.1. Computational cost improvement from using MLMC instead of Monte Carlo to achieve the same RMSE for the 1D slope test case.

We have thus shown that for a simple 1D test case in XBeach with an unknown slope, MLMC can significantly reduce computational cost compared to a traditional Monte Carlo approach.

5.3.1.2 Two-dimensional (2D) bed slope test case

In order to show the preceding promising results are not restricted to the simple 1D case, we consider the more complex test case of a 2D beach with a domain size of 1250 m by 1000 m. We use the same parameters as in Section 5.3.1.1 and modify the bed slope angle, θ, in both the x and y direction, as shown in Figure 5.6. We again choose a distribution of $\theta \sim \mathcal{U}(\arctan(0.035), \arctan(0.5))$ and seek to find the expected value of the maximum horizontal inundation distance during the simulation for all values of y. In order to satisfy the constraints on ϵ required to achieve the convergence stated in Theorem 1, the inundation distance is normalised by dividing it by the total x-length of the domain (1250 m).

Before running the full MLMC algorithm (Algorithm 1), as in the previous test case we run Step 1 first in order to check that the MLMC approach is appropriate. We use six levels ($l = 3$ to 8) with a grid-size of $\Delta x = 1250/2^l$ and $\Delta y = 1000/2^l$. Figure 5.7 shows that the variance decreases as the level number l increases, *i.e.* as the mesh gets finer, which means fewer samples will be needed on the finer meshes. The figure also shows that $\mathrm{Var}[X_l - X_{l-1}]$ is lower than $\mathrm{Var}[X_l]$ for all l. As discussed before, this is

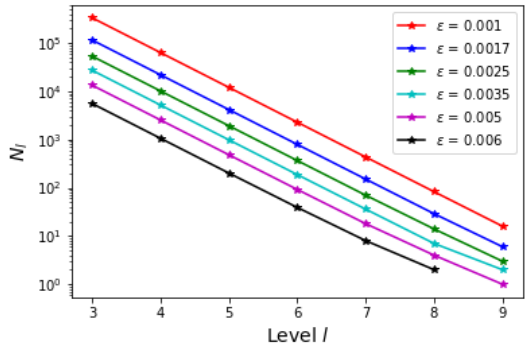
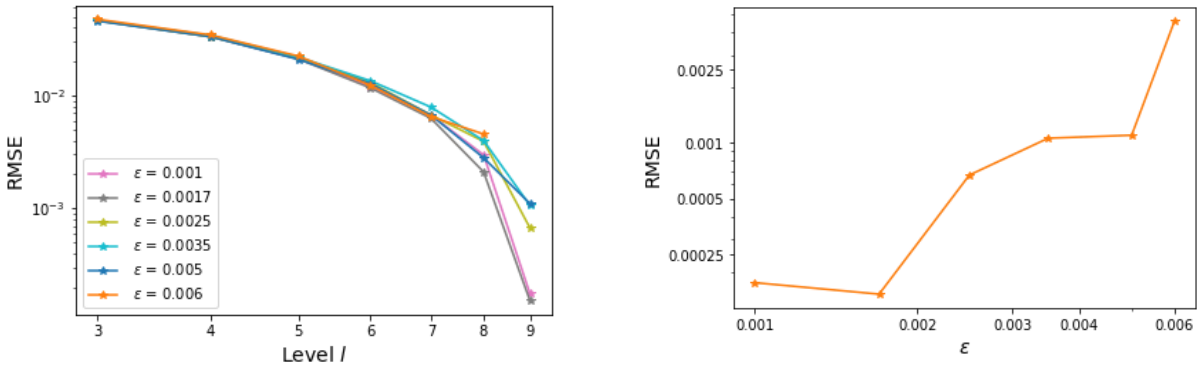


FIGURE 5.8. Optimum number of samples required at each level l for given tolerance value, ϵ, for the 2D bed slope test case.

important because otherwise a lower variance could be achieved by using the same number of samples with a standard Monte Carlo simulation. Thus, using the MLMC approach is again justified.

Using Figure 5.7 together with the log relationships between cost and level l and between $\hat{Y}_l$ and level l, convergence tests estimate the values in Theorem 1 to be $\alpha = 1.18; \beta = 1.69; \gamma = 2.93$ for this test case. Thus, the numerical discretisation is approximately first-order accurate (based upon the value of α). As $\gamma > \beta$ most of the computational cost in the MLMC algorithm comes from the finer levels and we expect the order of convergence to be higher than for the standard Monte Carlo method. These observations from the preliminary study, as well as the fact that the consistency and kurtosis checks from Section 5.2.1 passed, means we can conclude that MLMC can be used successfully with XBeach and we can run the full algorithm. For this test case, the chosen range of tolerance values are $\epsilon = [0.001, 0.0017, 0.0025, 0.0035, 0.005, 0.006]$, and we set a maximum limit on the finest level that l can reach of $L_{\max} = 9$ using (5.11) (see Step 7 of Algorithm 1). Note that we choose larger values of ϵ here than in Section 5.3.1.1 so that we do not need to consider as fine a mesh here as in that section. Finer meshes are not an issue for MLMC, but they would make the computational cost of the Monte Carlo simulation later in this section unmanageable. As $\gamma > \beta$, the optimum N_l are calculated using (5.17) and are shown in Figure 5.8. Most importantly, as the level becomes finer (*i.e.* as the level number increases), fewer samples are needed, which makes MLMC more efficient than a standard Monte Carlo method.

To test the statistical accuracy of the MLMC estimate, we also estimate the 'true' solution using a Monte Carlo simulation with $\Delta x = 1250/2^9 = 2.44\,\text{m}$ and $\Delta y = 1000/2^9 = 1.95\,\text{m}$, which is the same grid-size as the finest mesh in the MLMC simulation. This gives that the maximum expected horizontal inundation distance along the x-axis over all time and all y divided by the total x-length of the domain is equal to 0.6934. As the total domain length is 1250 m, this means the water inundates a horizontal distance of 866.8 m. We calculate the RMSE of the MLMC simulation compared to the Monte Carlo simulation using (5.22) as before (which is approximately the statistical error between the two simulations). Figure 5.9a shows that the RMSE decreases for all ϵ values considered as the mesh becomes finer. In addition, Figure 5.9b shows that as ϵ decreases, the final expected value generally becomes closer to the 'true' solution.



(A) RMSE of MLMC at each level l for given tolerance value ϵ.

(B) RMSE of MLMC once it has converged for each tolerance value ϵ.

FIGURE 5.9. RMSE of MLMC (5.22) for the 2D bed slope test case for varying tolerance value ϵ.

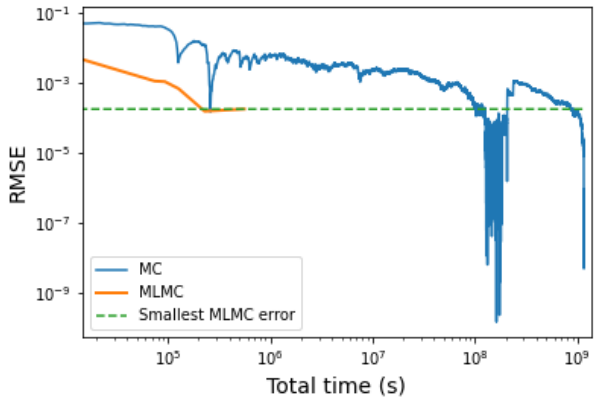


FIGURE 5.10. Comparison of total cost and statistical accuracy when using MLMC and Monte Carlo (MC) for the 2D bed slope test case.

Finally, Figure 5.10 compares the total cost and statistical accuracy of MLMC and Monte Carlo. Most significantly, as summarised in Table 5.2, MLMC achieves the same statistical accuracy as Monte Carlo for 1/1569 (0.06%) of the computational cost, which is a notable reduction. The reduction in computational time achieved by using MLMC rather than Monte Carlo is even greater here than in Section 5.3.1.1. This is a promising result as it suggests that the efficiency improvement from using MLMC increases as the number of dimensions increases.

	MLMC	**Monte Carlo**
RMSE	0.000175	0.000175
Time to achieve RMSE (hours)	160	250,973
MLMC speed-up factor	**1569**	

TABLE 5.2. Computational cost improvement from using MLMC instead of Monte Carlo to achieve the same RMSE for the 2D slope test case.

Although the test cases considered so far are somewhat abstract, they demonstrate that MLMC can be used with XBeach for a wave with known properties approaching a 1D or 2D beach with unknown slope. We have shown that the key advantage of MLMC is that it can achieve accurate results considerably

faster than a standard Monte Carlo method. This means we can confidently move to more realistic and complicated test cases.

5.3.2 Uncertainty in wave height

In the previous section, we considered the uncertainty associated with a beach bed profile. Another significant source of uncertainty is the structure of the waves approaching the beach. Whilst the previous test cases are purely theoretical, we consider here a lab test case and a real-world test case, with the uncertain parameter being the wave height in the JONSWAP wave spectrum, h (see Section 2.3.2 for more details on this spectrum). A key condition of MLMC theory is that, for each X_l^i, X_{l-1}^i pair calculation, the only difference in the model must be the mesh element size. Thus, when calculating X_l^i, we use the XBeach option `random = 0` which means that the same random seed is used to initialise the wave boundary conditions for both X_{l-1}^i and X_l^i. Both test cases below pass the consistency check in Section 5.2.1, indicating that the formulation is both mathematically correct and implemented correctly.

5.3.2.1 Morphology test case

Coastal areas are very vulnerable to erosion and in this section, we consider an example of this risk. For our test case, we consider the quasi-1D standard DelflandStorm case, available as part of the XBeach documentation (Roelvink et al., 2015) and set the distribution of the wave height to be $h \sim \mathcal{U}(0m, 5m)$. We then seek the expected total volume change over the simulation. This is calculated by using the trapezium rule to approximate the integral of the difference between the initial bed profile and the final bed profile, as shown in Figure 5.11. Thus, if the total volume change is positive, the erosion volume is greater than the accretion volume, and the beach has been net eroded, whilst if the inverse is true, the beach has accumulated material and grown. We normalise this volume change using the total initial volume of the beach, *i.e.* the volume underneath the slope using the lowest point of the bed as the bottom vertical limit. Note that these volumes are, in fact, areas because this test case only has one horizontal dimension. However, to avoid confusion with surface areas, we shall continue referring to them as volumes.

To check whether MLMC can be used for this test case, we run Step 1 of Algorithm 1. We use five levels ($l = 7$ to 11) with a grid-size of $\Delta x = 1451/2^l$ and an unaltered grid-size in the y-direction because this is a quasi-1D test case. Figure 5.12 shows that $\mathrm{Var}(X_l - X_{l-1})$ decreases as the mesh becomes finer and that for all l it is lower than $\mathrm{Var}(X_l)$. Using convergence tests, the values in Theorem 1 are $\alpha = 0.921, \beta = 1.97, \gamma = 2.40$. Thus, the numerical discretisation is approximately first-order accurate and, as $\gamma > \beta$, we expect the order of convergence to be higher than for the standard Monte Carlo method. In addition, the test case also passes the standard checks in Section 5.2.1.

Given the success of the preliminary study, we run the full MLMC algorithm using tolerance values of $\epsilon = [1.5 \times 10^{-5}, 2 \times 10^{-5}, 5 \times 10^{-5}, 10^{-4}]$ with a maximum limit for the finest level of $L_{\max} = 12$ (calculated using (5.11)). As $\gamma > \beta$, the optimum N_l are calculated using (5.17) and are shown in Figure

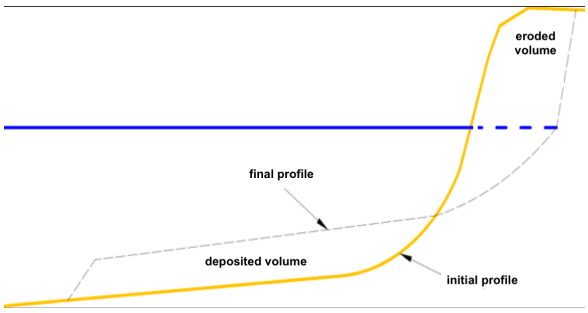


FIGURE 5.11. Schematic showing the initial and final bed profiles after a simulation. If the volume change in the Morphology test case is positive, this means that the erosion volume is greater than the accretion volume and vice versa. Adapted from Van Gent et al. (2007).

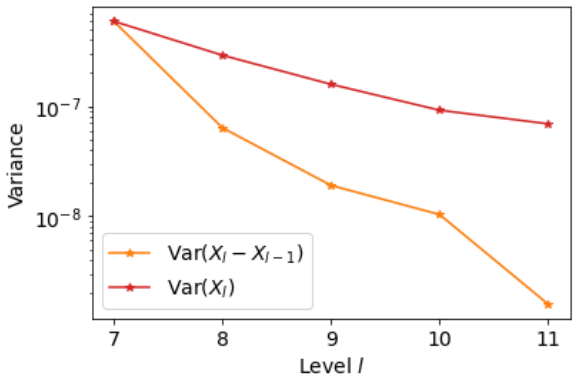


FIGURE 5.12. Variance of difference, $X_l - X_{l-1}$, compared to variance of single variable, X_l, at each level l for the Morphology test case showing the use of MLMC is justified.

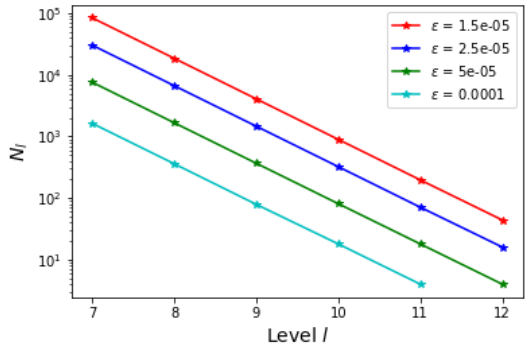
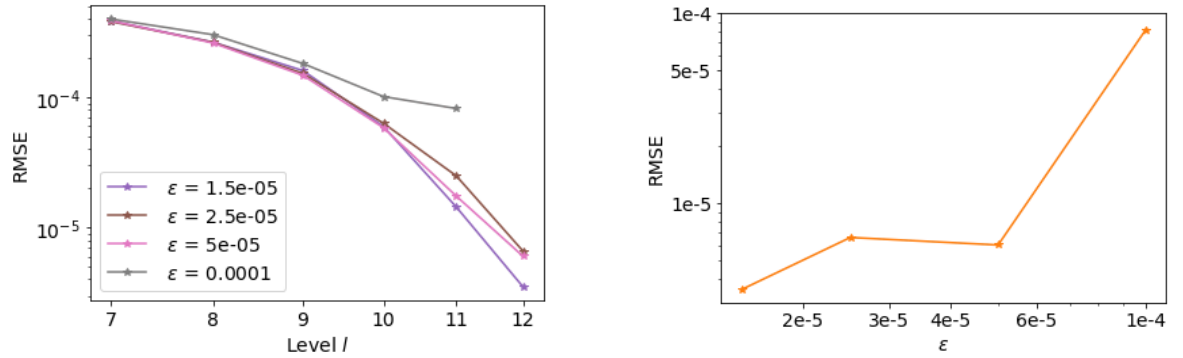


FIGURE 5.13. Optimum number of samples required at each level l for given tolerance value ϵ for the Morphology test case.

5.13. Importantly, the number of samples required decreases as the mesh becomes finer. As before, we calculate the RMSE of MLMC for this test case, by estimating the 'true' solution through a Monte Carlo simulation using $\Delta x = 1421.562/2^{12} = 0.347\,\mathrm{m}$, which is the same as the finest mesh utilised in the MLMC simulation. Therefore this RMSE is a measure of the difference in the statistical error. The Monte Carlo simulation gives an expected volume change as a proportion of the total initial volume of 0.0006150, which is positive and thus means the beach is net eroded during the simulation. As the total volume is $16\,333\,\mathrm{m}^2$, this means that $10\,\mathrm{m}^2$ of beach material is eroded during the simulation (recall here that the test case is in one horizontal dimension so the volume is, in fact, an area). Figures 5.14a and 5.14b confirm that the RMSE decreases as the level number increases and as ϵ decreases, respectively, meaning MLMC is working as expected. Unlike with the other test cases considered, Figure 5.14b shows that the relationship between the RMSE and ϵ is not uniform and there is a clear elbow in the plot at $\epsilon = 5 \times 10^{-5}$. This is a discrete effect caused by the tolerance $\epsilon = 10^{-4}$, passing the convergence test (5.19) without requiring any samples on the finest level ($l = 12$) (see Figure 5.13).



(A) RMSE of MLMC at each level l for given tolerance value ϵ.

(B) RMSE of MLMC once it has converged for each tolerance value ϵ.

FIGURE 5.14. RMSE of MLMC (5.22) for the Morphology test case for varying tolerance value ϵ.

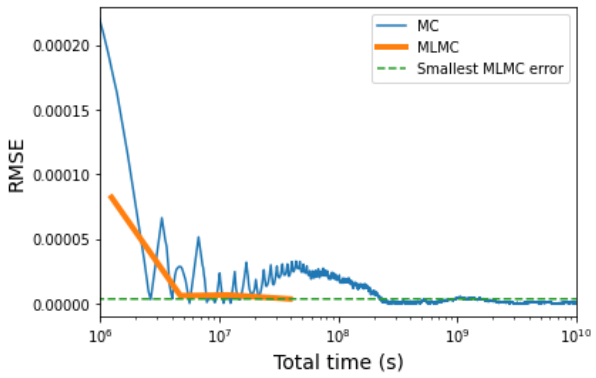


FIGURE 5.15. Comparison of total cost and statistical accuracy when using MLMC and Monte Carlo (MC) for the Morphology test case.

Finally, Figure 5.15 shows that MLMC with $\epsilon = 1.5 \times 10^{-5}$ achieves the same statistical accuracy as Monte Carlo for 1/43 (2%) of the computational cost, as summarised in Table 5.3. In addition, the figure shows that the convergence of the Monte Carlo simulation is notably less uniform than that from MLMC, meaning the Monte Carlo simulation must be run for longer than strictly necessary to ensure convergence – we stop the Monte Carlo algorithm once the Monte Carlo RMSE is below the MLMC RMSE for more than 20,000 samples. Thus, we have shown that MLMC can significantly decrease computational costs without compromising on accuracy, as well as improve result stability, for a full hydro-morphodynamic model with uncertainty in the wave height.

	MLMC	**Monte Carlo**
RMSE	3.51×10^{-6}	3.51×10^{-6}
Time to achieve RMSE (hours)	10,911	465,310
MLMC speed-up factor	**43**	

TABLE 5.3. Computational cost improvement from using MLMC instead of Monte Carlo to achieve the same RMSE for the Morphology test case.

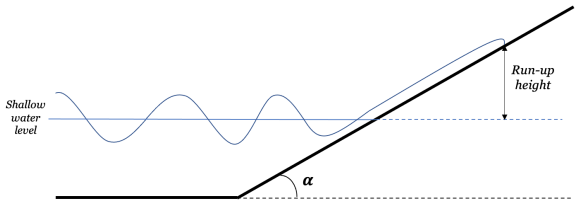


FIGURE 5.16. Schematic showing wave run-up height at a beach (the quantity of interest in the Boscombe Beach test case) and key quantities needed to calculate it.

5.3.2.2 Boscombe Beach test case

For the final test case of this chapter, we consider the real-world case of Boscombe Beach in Dorset, UK, which is a standard 2D case provided in the XBeach documentation (Roelvink et al., 2015). The distribution of the wave height is set to be $h \sim \mathcal{U}(0m, 3m)$. To simplify, the morphological and tidal component present in the standard test case are switched off, but all other values are unchanged.

When assessing flood risk, an important output variable of interest not yet considered in this chapter is the wave run-up height. Hunt Jr (1959) conclude that the wave run-up height is dependent on the structure of the wave and the angle of the beach bed slope. Traditionally, the value calculated is the wave run-up height exceeded by 2% of incoming waves, $R_{u2\%}$, shown in Figure 5.16, which is also known as the extreme run-up. This value is first mentioned in Asbeck et al. (1953), but has since been used often (see for example Stockdon et al., 2006; Suanez et al., 2015; Cohn and Ruggiero, 2016). According to Allsop et al. (2007), it is not entirely clear why the choice of 2% was made, although one possible explanation is that soft engineering measures, such as clay and grass, are considered sufficient to withstand the 2% of waves that exceed that height.

Although $R_{u2\%}$ is by definition a probabilistic measure, there exist a variety of standard empirical formulae that can be used to estimate this value deterministically for an individual simulation (see Melby et al., 2012a). Furthermore, run-up height is difficult to simulate due to changing infiltration rates and thus empirical formulae are often preferable than recording the run-up height from the model. Using field data from natural beaches with slopes between 0.07 and 0.2, Holman (1986) derives the following empirical run-up formula

$$R_{u2\%} = 0.83\xi_0 H_{m0} + 0.2H_{m0}, \tag{5.23}$$

where H_{m0} is the significant wave height at the beach-toe taken from the model simulation (*i.e.* where the water elevation (z_s) and the bed (z_b) are equal) and ξ_0 the Iribarren parameter defined by

$$\xi_0 = \frac{\tan\alpha}{\sqrt{H_{m0}/L_0}}, \tag{5.24}$$

where α is the beach slope angle at the beach-toe, L_0 is the wavelength calculated by

$$L_0 = \frac{gT_p^2}{2\pi}, \tag{5.25}$$

and T_p the peak wave period. Although relatively simple, (5.23) is a standard formula for calculating extreme wave run-up on natural beaches (see for example Ruggiero et al., 2001; Stockdon et al., 2006; Melby et al., 2012b; Díaz-Sánchez et al., 2014; Suanez et al., 2015; Park and Cox, 2016). Furthermore, this formula is appropriate for the shallow slope of Boscombe Beach, and thus we use it to find the expected value of the maximum run-up height $R_{u2\%}$ attained during the simulation over all values of y.

Convolution filter

The Boscombe Beach profile is defined using the bed data provided in the XBeach documentation (Roelvink et al., 2015). Clearly, in real-world environments, the beach approach is not smooth like it is with the previous test cases in this chapter. Instead, there may be sandbars or other similar physical features present, either natural or anthropogenic in nature. This is a potential issue, especially if a feature suddenly appears in finer meshes, although it is not as severe an issue as it could be because MLMC only requires that each mesh l is a 'good enough' approximation to the next finest mesh $l+1$.

In order to mitigate the issue of features suddenly appearing, we use a convolution matrix filter which has the general expression

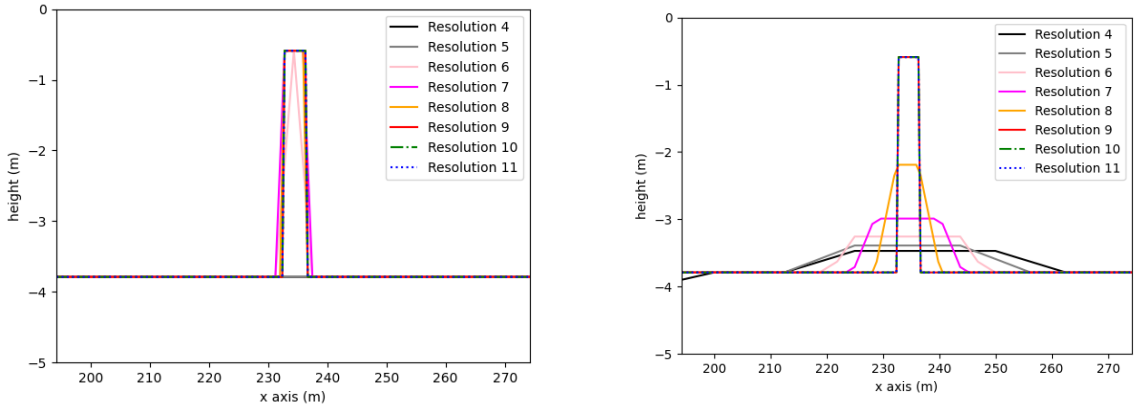
$$(f * g)(q) = \sum_{k=-\lambda}^{\lambda} g(k) f(q-k), \tag{5.26}$$

where $\lambda = \lfloor \frac{m}{2} \rfloor$ and m is the length of g. An appropriate filter kernel to use in this case is the normalised arithmetic mean

$$g(j) = \frac{1}{\sum_{q=1}^{m} g(q)}. \tag{5.27}$$

At the edges, we use the 'nearest' method, meaning that the edge value is repeated outwards as many times as necessary.

To understand the effect of the convolution filter, we consider a bump at different resolutions i (which corresponds to having 2^i elements in the x-direction). Figure 5.18a shows the result when no convolution filter is applied and Figure 5.18b the result when a convolution filter of length $m_i = \max((I_{\max} - i), 1)$ is applied at each resolution i, where $I_{\max}$ is the finest resolution considered. Note that when $m_i = 1$, the convolution filter is the identity filter and, thus, at that resolution the convolved bed corresponds to the original bed. In Figure 5.18a, the bump suddenly appears between $i = 5$ and $i = 6$, meaning the bed at $i = 5$ is unlikely to be a good approximation to that at $i = 6$, and therefore MLMC will not work well. By contrast, with the convolution filter, Figure 5.18b shows the feature 'grows' uniformly with no sudden jumps as the mesh becomes finer, slowly converging to the original bed, and meaning that there are no associated errors with using a convolution filter.



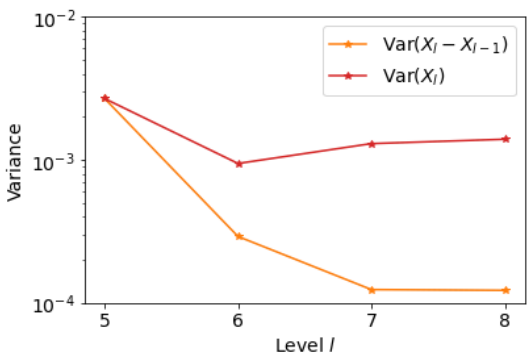
(A) With no convolution filter.

(B) With convolution filter (5.26).

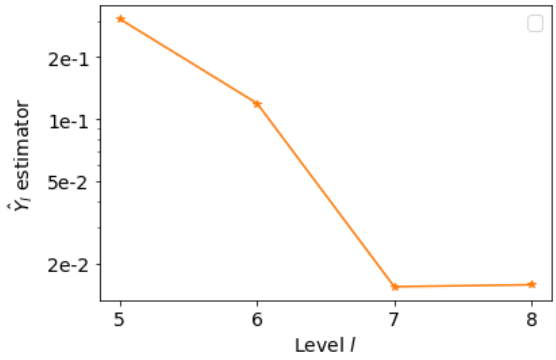
FIGURE 5.18. Comparing a bump at different mesh resolutions (number of elements $= 2^i$ for resolution i) with and without a convolution filter.

We therefore apply the convolution filter to the Boscombe Beach test case in the direction perpendicular to the coastline, with f being the underlying bed and the convolution filter length at level l, m_l, equalling $\max((L_{\max} - l), 1)$. To check whether MLMC is valid for this test case, as before, we run Step 1 of Algorithm 1. We use four levels ($l = 5$ to 8) with a grid-size of $\Delta x = 1123/2^l$ and $\Delta y = 1604/2^l$. Figure 5.19a shows that $\mathrm{Var}\,(X_l - X_{l-1})$ decreases as the mesh gets finer and, for all l, is lower than $\mathrm{Var}\,(X_l)$. Between the last two levels, there is only a slight decrease in the variance and this is also the case for the expectation as seen in Figure 5.19b. This plateauing is probably because of the loss of accuracy in the bathymetry due to interpolation being required on the finest mesh (the bathymetry data used is defined on a non-uniform mesh with an average grid size between $\Delta x = 1123/2^7$ and $\Delta x = 1123/2^8$). Using convergence tests, the values in Theorem 1 are $\alpha = 1.58$, $\beta = 1.46$ and $\gamma = 2.97$. As $\gamma > \beta$, we expect the order of convergence to be better than the standard Monte Carlo method. However the plateauing effect in the variance means that as $\epsilon \to 0$, Theorem 1 will no longer hold. Despite this, Figure 5.20 shows that for the values of ϵ considered, MLMC still approximately follows the expected cost relation. Finally, Figure 5.19b shows that the numerical discretisation is approximately first-order accurate, as with the test case in Section 5.3.2.1. In addition, the simulation passes the consistency and kurtosis checks from Section 5.2.1, and we can be confident that the MLMC algorithm is appropriate.

We then consider tolerance values of $\epsilon = [0.001, 0.0025, 0.005, 0.0075]$ and set a maximum limit of the finest mesh of $L_{\max} = 8$ (see Step 7 of Algorithm 1), as this corresponds to the mesh that the Boscombe Beach bed data is defined on. As $\gamma > \beta$, the optimum N_l are calculated using (5.17). Importantly, Figure 5.21 shows that the number of samples required decreases as the mesh becomes finer. As before, we also calculate the RMSE of the MLMC simulation by estimating the 'true' solution through a Monte Carlo simulation with a mesh of $\Delta x = 1123/2^8 = 4.39\,\mathrm{m}$ and $\Delta y = 1604/2^8 = 6.27\,\mathrm{m}$, which is the same as the finest MLMC mesh. This gives an expected maximum value of $R_{u2\%}$ equal to 0.1568 m. Using this value, Figures 5.22a and 5.22b confirm that the RMSE (5.22) decreases as the level becomes finer and



(A) Variance of difference, $X_l - X_{l-1}$, compared to variance of single variable, X_l, at each level, l.



(B) $\hat{Y}_l$ (the estimator for $\mathbb{E}[X_l - X_{l-1}]$) at each level, l.

FIGURE 5.19. Preliminary MLMC results for Boscombe Beach test case showing the use of MLMC is justified.

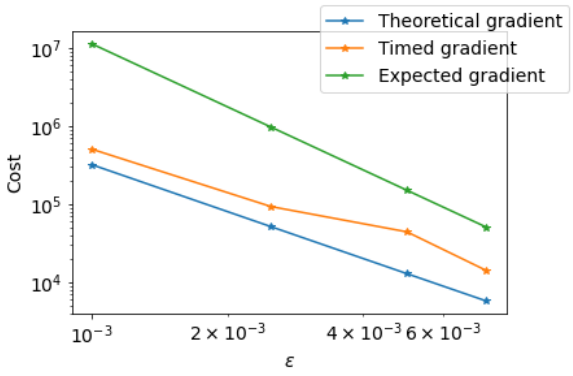


FIGURE 5.20. Verification that both theoretical cost (5.21) and timed cost vary with ϵ at the expected rate for the Boscombe Beach test case.

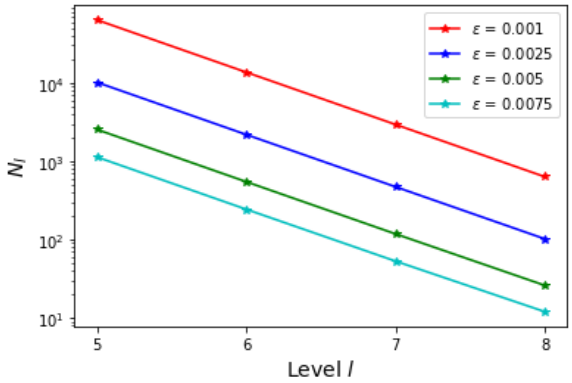
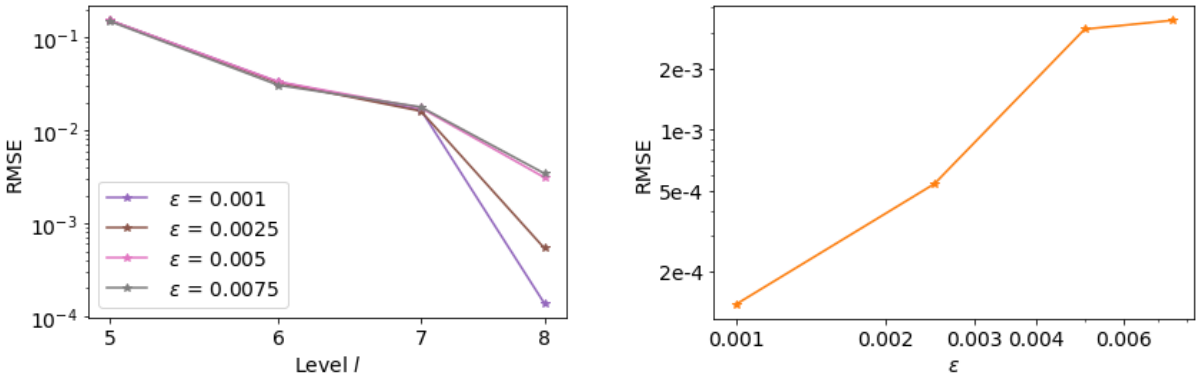


FIGURE 5.21. Optimum number of samples required at each level l for a given tolerance value ϵ for the Boscombe Beach test case.

ϵ decreases, respectively, meaning the MLMC algorithm is working as expected. Interestingly, Figure 5.22a shows that ϵ only has a notable effect on the error when the finest level is included, likely because previous levels are using the convolved version of the bed rather than the real data.

Finally, Figure 5.23 compares the total cost and statistical accuracy of MLMC versus Monte Carlo. Notably, MLMC achieves the same statistical accuracy as the Monte Carlo simulation for 1/61 (1.6%) of the computational cost, as summarised in Table 5.4. Figure 5.23 also shows the Monte Carlo simulation must be run for longer than strictly necessary to ensure convergence, as is the case for all the test cases considered in this chapter, and in contrast with MLMC. Thus, we have shown that for a real-world test case with uncertain wave height, MLMC can significantly decrease computational cost and increase result stability without compromising on statistical accuracy.



(A) RMSE of MLMC at each level l for given tolerance value ϵ.

(B) RMSE of MLMC once it has converged for each tolerance value ϵ.

FIGURE 5.22. RMSE of MLMC (5.22) for the Boscombe Beach test case for varying tolerance value ϵ.

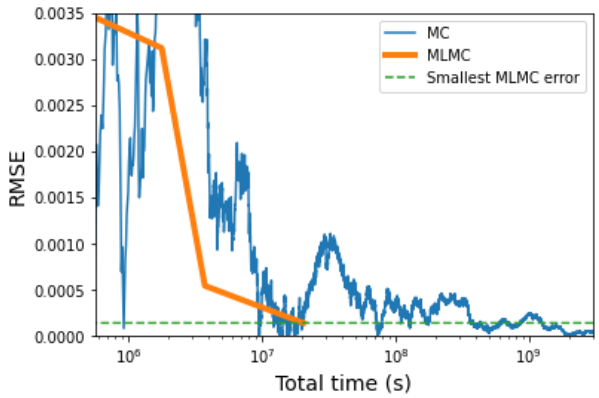


FIGURE 5.23. Comparison of total cost and statistical accuracy when using MLMC and Monte Carlo (MC) for the Boscombe Beach test case.

	MLMC	**Monte Carlo**
RMSE	1.38×10^{-4}	1.38×10^{-4}
Time to achieve RMSE (hours)	5662	347294
MLMC speed-up factor	**61**	

TABLE 5.4. Computational cost improvement from using MLMC instead of Monte Carlo to achieve the same RMSE for the Boscombe Beach test case.

5.4 Cumulative distribution functions

In an MLMC framework, the objective is normally to find the expectation of an output variable. However, when analysing risk in coastal problems, the probability of an output variable exceeding a certain value is also of interest. As discussed in Section 5.1, this probability is complicated to compute because MLMC provides very few values at the same resolution from which to build a distribution.

To resolve this issue, we follow Gregory and Cotter (2017) and use the inverse transform sampling method to evaluate the inverse cumulative distribution function, $F^{-1}(u)$ where $F(x) = \mathbb{P}(X \leq x)$. If F is strictly

increasing and absolutely continuous, then $x \equiv F^{-1}(u)$ is unique. A simple consistent estimate for x can be found by sorting the samples such that $X^1 < X^2 < ... < X^N$ and then calculating

$$\hat{F}^{-1}(u) = X^{\lceil N \times u \rceil}, \tag{5.28}$$

where $u \sim \mathcal{U}[0,1]$. This estimate is consistent as it converges in probability to x as $N \to \infty$ (Gregory and Cotter, 2017) *i.e.* the probability of $X^{\lceil N \times u \rceil}$ being more than a small distance ϵ from x tends to zero as $N \to \infty$. For an MLMC approximation, the inverse cumulative distribution function is then approximated by

$$F_L^{-1}(u) \simeq X_F^i = R(X)_{l_\mu}^{\lceil N_{l_\mu} \times u \rceil} + \sum_{l=l_\mu+1}^{L} \left(R(X)_l^{\lceil N_l \times u \rceil} - R(X)_{l-1}^{\lceil N_{l-1} \times u \rceil} \right), \tag{5.29}$$

where $R(X)_l^i$ represent the i^{th} order statistic of X_l on each level l. Note that, unlike with (5.2), there is no exact cancellation because the approximations at each level are not unbiased. Thus, (5.29) cannot generate samples from X_L but instead generate an ensemble of X_F^i, which are consistent approximations to $F_L^{-1}(u)$. As $N_l \to \infty$, these converge in probability to $F_L^{-1}(u)$. The cumulative distribution function (CDF) of the MLMC approximation is then given by

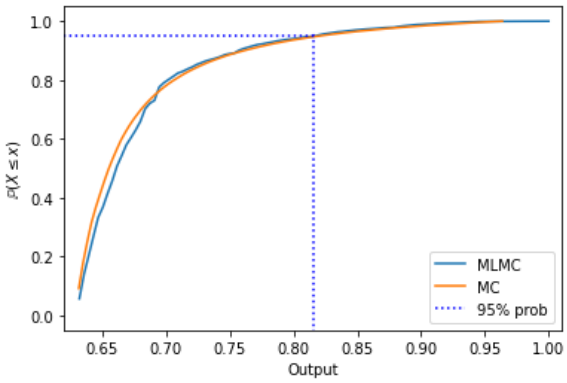
$$F(x) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}_{X_F^i \leq x}. \tag{5.30}$$

We refer the reader to Gregory and Cotter (2017) for more details on this method.

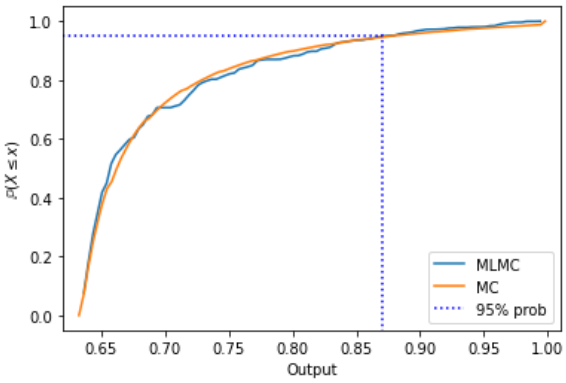
5.4.1 Applying inverse transform sampling to MLMC outputs

We thus apply the inverse transform sampling method to each of the four test cases discussed in Section 5.3. This method provides a CDF from which the probability that the output variable exceeds a certain value can be predicted. This is particularly useful for predicting the probability of extreme flooding/erosion events occurring, a valuable capability in coastal risk assessment. This method is applied by using the MLMC outputs generated with the lowest ϵ value considered in each test case. The outputs are ordered at each level separately, and then (5.29) is used to create pseudo-samples on the finest level.

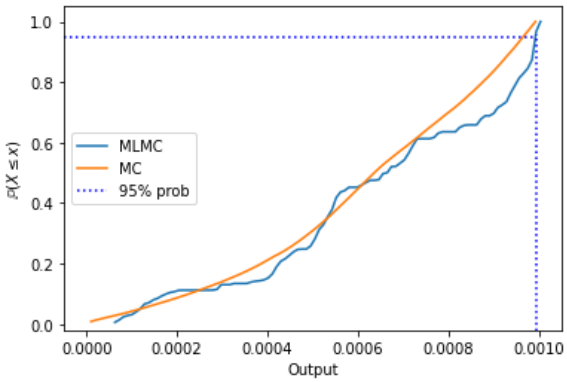
The CDFs for all four test cases are shown in Figure 5.24 along with dotted lines on each subfigure denoting $\mathbb{P}(X \leq x) = 0.95$ or, equivalently, $\mathbb{P}(X > x) = 0.05$. We can use these figures to readily assess risk. For example, Figure 5.24b shows that the distance the maximum inundation exceeds 5% of the time is equal to 1090 m (*i.e.* 0.87×1250 m as the original output is normalised). From a town planner's perspective, allowing a permanent structure to be built in this area would be inadvisable. Furthermore, Figure 5.24 also compares the CDFs generated from MLMC outputs with those generated from Monte Carlo outputs and shows that qualitatively there is good agreement between them for all test cases. In Table 5.5, we quantify this good agreement by showing that the L^2 error norm between the MLMC CDFs and Monte Carlo CDFs is always small. Note that we evaluate the CDFs at 100 equally spaced points and therefore the implementation of this L^2 error norm is equivalent to calculating the RMSE between the two CDFs.



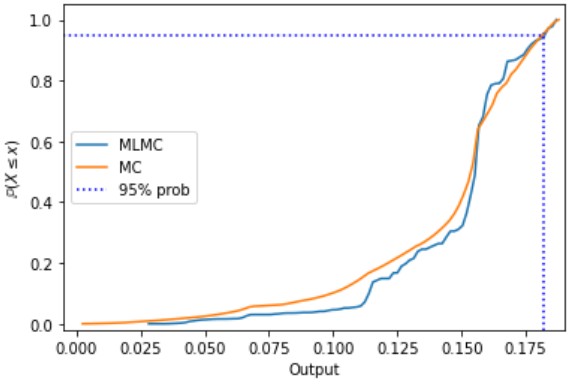
(A) 1D bed slope test case (Section 5.3.1.1).



(B) 2D bed slope test case (Section 5.3.1.2).



(C) Morphology test case (Section 5.3.2.1).



(D) Boscombe Beach test case (Section 5.3.2.2).

FIGURE 5.24. Comparing CDFs generated from MLMC outputs using the inverse transform sampling method with those generated using Monte Carlo (MC) outputs.

TABLE 5.5. L^2 error norm between the MLMC and Monte Carlo CDFs for each test case.

Test Case	L^2 **Error Norm**
1D bed slope	2.18×10^{-2}
2D bed slope	3.04×10^{-2}
Morphology	4.94×10^{-2}
Boscombe Beach	4.03×10^{-2}

Thus, we can conclude that MLMC can be used to not only find the expected value of output variables, but also to determine the probability of extreme flooding and erosion events occurring, and assess whether certain areas are at risk.

5.5 Conclusion

This chapter has shown the first successful application of MLMC in the coastal engineering field and the significant improvement in computational efficiency it provides when estimating the expected value and

cumulative distribution function of key variables of interest. Another significant advantage is that the MLMC result is stable, unlike the Monte Carlo one – in all test cases considered decreasing ϵ (*i.e.* the accuracy tolerance) in MLMC leads to a result with a smaller statistical error. Therefore, there is no need to run the simulation for longer to ensure convergence, like there is with Monte Carlo, thus providing another computational cost advantage.

These efficiency advantages mean that MLMC enables uncertainty analysis to be performed on previously unfeasible cases. For example, in Section 5.1, we discuss the test case in Callaghan et al. (2013), which is similar to both the morphology and Boscombe Beach test cases in this chapter. We show that using MLMC for these two test cases results in a computational cost speed-up of 43 and 61 times, respectively. This suggests that using MLMC would cut the 4.5 millennia the authors of Callaghan et al. (2013) estimate it would take to run using XBeach with Monte Carlo on a single processor to 'only' 90,000 years on the same processor. Whilst this is still a large cost, this is no longer outside the realm of certain supercomputers, since the methodology and our framework can be shared across distributed cores.

Finally, the flexibility of MLMC means that it has no difficulties in dealing with the complex nature of the XBeach model, particularly because it only requires the result from one mesh to be a good approximation to that of the previous coarser mesh. The main hindrance to this approximation is that, in real-world cases, small bed features can appear as the mesh becomes finer, but we were able to resolve this issue by implementing convolution filters. This is an encouraging sign for further work, when more complex examples will be considered, and the effect of uncertainty in more than one parameter on more than one output variable will be investigated.

Code availability

The relevant code for the MLMC framework presented in this chapter is stored at `https://github.com/mc4117/MLMC_XBeach`.

CHAPTER 6

A multi-model approach to assessing risk through the application of the multilevel multifidelity Monte Carlo method

Highlights

- First application of the *multilevel multifidelity* Monte Carlo method (MLMF) to assess coastal flood risk hazards;
- MLMF provides notable improvements in computational cost for the same or higher statistical accuracy compared to both MLMC and the Monte Carlo method;
- In-depth statistical analysis of a complex real-world test case which would not be feasible using the Monte Carlo method;
- First use of MLMF to successfully estimate the cumulative distribution of output variables, helping with informed decision making.

Summary

When choosing an appropriate hydro-morphodynamic model, there is always a compromise between accuracy and computational cost, with high fidelity models generally being more expensive than low fidelity ones. However, when assessing uncertainty, we can use a multifidelity approach to take advantage of the accuracy of high fidelity models and the computational efficiency of low fidelity models. Here, we apply the *multilevel multifidelity* Monte Carlo method (MLMF) to quantify uncertainty in a computationally efficient manner by computing the expected values of key output variables with respect to uncertain parameters, using the high fidelity hydro-morphodynamic model XBeach and the low fidelity coastal flooding solver SFINCS. This work represents the first application of MLMF in the coastal zone and one of its first applications in any field. For both theoretical and real-world test cases, MLMF can significantly reduce computational cost for the same or higher statistical accuracy compared to both MLMC and the standard Monte Carlo method. In particular, we demonstrate using the case of Myrtle Beach (USA) that this improvement in computational efficiency allows uncertainty analysis to be conducted in the case of real-world coastal environments – a task that would have been practically unfeasible with standard Monte Carlo methods. Moreover, for the first time, we show how an inverse transform sampling technique can be used to accurately estimate the cumulative distribution function (CDF) of variables from MLMF outputs. MLMF's estimates of the expectations and the CDFs of the variables of interests are of significant value to decision makers when assessing risk.

6.1 Introduction

In hydro-morphodynamic modelling, like in most fields, there is no single standard model that is always appropriate and, instead, a number of different models exist. Some models will be high fidelity but computationally expensive, whilst others will compromise on fidelity (*i.e.* be low fidelity) but have a low computational cost. The 'holy grail' of a high fidelity computationally cheap hydro-morphodynamic model is yet to be found. However, when assessing uncertainty, we can use multifidelity approaches which take advantage of the accuracy of high fidelity models and the computational efficiency of low fidelity ones in order to produce accurate yet computationally cheap uncertainty analyses (Peherstorfer et al., 2018). More recently, the *multilevel multifidelity* Monte Carlo (MLMF) method, developed in Geraci et al. (2015), combines the hierarchy of model resolutions approach of MLMC (see Section 5.2) with the multiple models used in multifidelity approaches. Research into these methods is still in its infancy, but they have been successfully applied in aerospace research (Geraci et al., 2017) and cardiology (Fleeter et al., 2020).

In this chapter, we present the first application of MLMF in the coastal zone and show that it results in a significant decrease in the computational cost of assessing risk and uncertainty for the same statistical accuracy, even when compared to the sophisticated MLMC approach that itself offers substantial cost savings over standard Monte Carlo methods. MLMF quantifies uncertainty by computing estimators of the expected value of key output variables with respect to uncertain input parameters. We emphasise here that the purpose of our work is to use MLMF to simulate statistics on a fine resolution grid more efficiently than the standard Monte Carlo method or MLMC. The difference in the final Monte Carlo, MLMC and MLMF estimators is the statistical (sampling) error – they will have approximately the same model and discretisation errors because in our comparisons the Monte Carlo method resolution is the same as the finest resolution in MLMC and MLMF. Of course, it is possible or even likely that the model error dominates the statistical error, but to manage this, we choose XBeach (see Section 2.3.2) as our high fidelity model because it has been successfully validated and verified numerous times in the coastal zone including, for example, in Roelvink et al. (2018) and de Beer et al. (2020). Furthermore, we have already shown in Chapter 5 that the MLMC approach used in combination with XBeach is considerably more efficient than standard Monte Carlo and, thus, it is interesting to investigate whether further improvements can be made through combining XBeach with an MLMF framework. For our low fidelity model, we choose SFINCS (see Section 2.3.3) because it is a relatively cheap model based on reduced physics. Also, SFINCS has already been favourably compared to XBeach in numerous test cases (see Leijnse, 2018; Leijnse et al., 2021), demonstrating that it is computationally cheaper than XBeach while not being extremely less accurate. Together these points indicate that SFINCS is a good choice for a low fidelity model.

We test our MLMF implementation through the consideration of a variety of theoretical and real-world test cases, some of which would have been impractical and unrealistic to run using standard Monte Carlo methods. Note that we study coastal flooding problems in this chapter using the hydrodynamic

version of XBeach because SFINCS does not currently possess a morphodynamic component. However, we make the problem more complex by calculating the expected value of output variables at multiple locations simultaneously, resulting in a spatial uncertainty analysis of the coastal flooding. Furthermore, we modify the inverse transform sampling method from Gregory and Cotter (2017) so that MLMF can be used to estimate the cumulative distribution function at these locations. This provides information allowing practitioners to determine the probability/risk of a variable exceeding a certain value, which can be of more interest than the expected value.

The remainder of this chapter is structured as follows: in Section 6.2, we outline the relevant MLMF theory; in Section 6.3, we apply MLMF with XBeach and SFINCS to theoretical and real-world test cases; in Section 6.4, we estimate the cumulative distribution function for the considered output variables; and, finally, in Section 6.5, we conclude this chapter.

6.2 Multilevel multifidelity Monte Carlo method (MLMF)

The *multilevel multifidelity* Monte Carlo method (MLMF) was first introduced in Geraci et al. (2015), and combines MLMC with multifidelity estimators. Whilst we have already described MLMC in Section 5.2, we have not yet described multifidelity estimators and, thus, we begin this section by briefly outlining them.

6.2.1 Multifidelity estimators

Generally, a multifidelity approach uses a low fidelity model to generate surrogate approximations for the outputs of a high fidelity model. If applied correctly, the resulting multifidelity estimator is then as accurate as the equivalent high fidelity one. There exist a number of different multifidelity approaches (see Peherstorfer et al., 2018) and MLMF uses the control variate approach which we outline here, following Geraci et al. (2015) throughout.

The general form of the control variate is

$$Q_M^{HF,CV} = Q_M^{HF} + \alpha_{\mathrm{F}} \left(Q_M^{LF} - \mathbb{E}[Q_M^{LF}] \right), \tag{6.1}$$

where M indicates the fixed discretisation level, α_{F} is a scalar determined below in (6.3), and HF and LF are used throughout this chapter to denote the high and low fidelity models, respectively. The multifidelity estimator is unbiased and given by

$$\hat{Q}_{M,N}^{HF,CV} = \hat{Q}_{M,N}^{HF} + \alpha_{\mathrm{F}} \left(\hat{Q}_{M,N}^{LF} - \mathbb{E}[Q_M^{LF}] \right), \tag{6.2}$$

where α_{F} is a scalar and N is the number of samples used. Recall that, throughout this thesis, we use the standard notation, $\hat{\cdot}$, to denote an estimator. The value of α_{F} is then determined by minimising the

variance of $\hat{Q}_{M,N}^{HF,CV}$ and is given by

$$\alpha_{\mathrm{F}} = -\rho\sqrt{\frac{\mathrm{Var}(\hat{Q}_{M,N}^{HF})}{\mathrm{Var}(\hat{Q}_{M,N}^{LF})}}, \tag{6.3}$$

where ρ is the Pearson's correlation coefficient for the HF and LF estimators.

Equation (6.2) assumes that $\mathbb{E}\left[Q_M^{LF}\right]$ is known, but this is almost never true. Therefore, extra realisations of the LF model must be conducted in order to estimate this quantity. To avoid confusion, the number of realisations to estimate both $\hat{Q}_{M,N}^{HF}$ and $\hat{Q}_{M,N}^{LF}$ is denoted as N_{HF}, and the number of extra realisations is denoted by Δ^{LF}. By definition $\Delta^{LF} = rN_{HF}$, where the optimum value of r is determined later. Thus, the overall computational cost of the multifidelity estimator is

$$C = C^{HF} + C^{LF}(1+r), \tag{6.4}$$

and its variance is

$$\mathrm{Var}\left[\hat{Q}_{M,N}^{HF,CV}\right] = \mathrm{Var}\left[\hat{Q}_{M,N}^{HF}\right]\left(1 - \frac{r}{1+r}\rho^2\right). \tag{6.5}$$

Note that ρ^2 is less than one by definition, so r greater than zero means that the variance of the estimator is reduced by using this method.

6.2.2 Multilevel multifidelity Monte Carlo method (MLMF)

Although both MLMC and multifidelity approaches can improve computational efficiency compared to the standard Monte Carlo method, a greater improvement can be achieved by combining these two approaches using MLMF. To derive the MLMF estimator, $\hat{Q}_{M_L}^{HF,CV}$, we follow Geraci et al. (2017) and combine the MLMC estimator (5.4) with the control variate (6.2) to obtain

$$\begin{aligned}\mathbb{E}\left[Q_{M_L}^{HF,CV}\right] &= \mathbb{E}\left[Q_{M_{l_\mu}}^{HF,CV}\right] + \sum_{l=l_\mu+1}^{L}\mathbb{E}\left[Q_{M_l}^{HF,CV} - Q_{M_{l-1}}^{HF,CV}\right] \\ &\simeq \hat{Q}_{M_L}^{HF,CV} \\ &\simeq \sum_{l=l_\mu}^{L}\left(\hat{Y}_{M_l}^{HF} + \alpha_l\left(\hat{Y}_{M_l}^{LF} - \hat{E}\left[Y_{M_l}^{LF}\right]\right)\right),\end{aligned} \tag{6.6}$$

where

$$\alpha_l = -\rho_l\sqrt{\frac{\mathrm{Var}(\hat{Y}_{M_l}^{HF})}{\mathrm{Var}(\hat{Y}_{M_l}^{LF})}}, \tag{6.7}$$

and recall l_μ is the coarsest level considered. Note that $\hat{E}[\cdot]$ denotes an estimator for the expectation throughout this chapter. The estimators $\hat{Y}_{M_l}^{LF}$ and $\hat{Y}_{M_l}^{HF}$ are the MLMC estimators defined using (5.5)

which we state below for HF

$$\hat{Y}_{M_l}^{HF} = \begin{cases} \frac{1}{N_{l_\mu}^{HF}} \sum_{i=1}^{N_{l_\mu}^{HF}} X_{l_\mu}^{(i)} & l = l_\mu, \\ \frac{1}{N_l^{HF}} \sum_{i=1}^{N_l^{HF}} \left(X_l^{(i)} - X_{l-1}^{(i)} \right) & l > l_\mu. \end{cases} \tag{6.8}$$

Recall also from Section 5.2 that X_l and X_{l-1} are on the same Brownian path, *i.e.* we use the same random numbers to construct the variables X_l and X_{l-1}. Additionally, in MLMF, the estimators $\hat{Y}_{M_l}^{HF}$ and $\hat{Y}_{M_l}^{LF}$ are also on the same Brownian path, *i.e.* the same random numbers are used to construct both the estimators.

For reasons of independence, different independent samples are used at each level (as with MLMC) and, hence, the variance of the MLMF estimator is

$$\text{Var}\left[\hat{Q}_{M_L}^{HF,CV}\right] = \sum_{l=l_\mu}^{L} \left(N_l^{HF}\right)^{-1} \text{Var}\left[\hat{Y}_l^{HF}\right] \left(1 - \frac{r_l}{1+r_l}\rho_l^2\right), \tag{6.9}$$

where N_l^{HF} is the number of HF samples required to compute $\hat{Y}_{M_l}^{HF}$ which is also the number of LF samples required to compute $\hat{Y}_{M_l}^{LF}$, and r_l is the factor of extra LF samples required to compute $\hat{E}\left[Y_{M_l}^{LF}\right]$. Note that, throughout this chapter and for simplicity, we follow Geraci et al. (2015) and refer to N_l^{HF} as the number of HF samples required, because the total number of LF samples required is the combined quantity $(1+r_l)N_l^{HF}$, and not just N_l^{HF}.

Using the cost of the multifidelity approach (6.4) and the cost of the MLMC approach (5.7), the overall cost of the MLMF algorithm is

$$C = \sum_{l=l_\mu}^{L} N_l^{HF} \left(C_l^{HF} + C_l^{LF}(1+r_l)\right). \tag{6.10}$$

In order to obtain the optimum values for N_l^{HF} and r_l in (6.9), we minimise this cost with respect to the statistical error constraint $\text{Var}\left[\hat{Q}_{M_L}^{HF,CV}\right] < \epsilon^2/2$. In mathematical terms, this is equivalent to finding the stationary point of

$$f(N_l^{HF}, r_l, \lambda) = \sum_{l=l_\mu}^{L} N_l^{HF} \left(C_l^{HF} + C_l^{LF}(1+r_l)\right) + \lambda \left(\sum_{l=l_\mu}^{L} \left(N_l^{HF}\right)^{-1} \text{Var}\left[\hat{Y}_l^{HF}\right] \left(1 - \frac{r_l}{1+r_l}\rho_l^2\right) - \frac{\epsilon^2}{2} \right), \tag{6.11}$$

with respect to N_l^{HF} and r_l, where λ is the Lagrange multiplier. This minimisation results in the following optimum formula for the factor of extra LF samples

$$r_l = -1 + \sqrt{\frac{\rho_l^2}{1-\rho_l^2}\omega_l}, \tag{6.12}$$

where $\omega_l = C_l^{HF}/C_l^{LF}$ is the cost ratio between the HF and LF models, and the following optimum formula for the number of HF samples

$$N_l^{HF} = \frac{2}{\epsilon^2} \left[\sum_{k=l_\mu}^{L} \left(\frac{\text{Var}\left[\hat{Y}_k^{HF}\right] C_k^{HF}}{1-\rho_l^2} \right)^{1/2} \Lambda_k(r_k) \right] \sqrt{(1-\rho_l^2) \frac{\text{Var}\left[\hat{Y}_l^{HF}\right]}{C_l^{HF}}}, \tag{6.13}$$

where

$$\Lambda_k(r_k) = 1 - \frac{r_k}{1+r_k}\rho_k^2. \tag{6.14}$$

6.2.2.1 Increasing the correlation between samples

The variance formula (6.9) and the optimum number of samples formulae (6.12) and (6.13) all depend heavily on the correlation ρ between the HF and LF models. The greatest efficiency improvements are achieved when the correlation between the two models is high and we thus seek to maximise it. Geraci et al. (2017) show that, because the multifidelity control variate is unbiased, correlation can be artificially increased by modifying the estimator $\hat{Y}_l^{LF}$ using

$$\mathring{Y}_l^{LF} = \gamma_l \hat{X}_l^{LF} - \hat{X}_{l-1}^{LF}, \tag{6.15}$$

where the modification factor γ_l adds an extra degree of freedom to maximise the correlation. Therefore, (6.6) becomes

$$\hat{Q}_{M_L}^{HF,CV} = \sum_{l=l_\mu}^{L} \left(\hat{Y}_{M_l}^{HF} + \alpha_l \left(\mathring{Y}_l^{LF} - \hat{E}\left[\mathring{Y}_l^{LF} \right] \right) \right), \tag{6.16}$$

and the new correlation coefficient $\mathring{\rho}_l^2$ is dependent on γ_l and is equal to

$$\mathring{\rho}_l^2 = \rho_l^2 \frac{\mathrm{Cov}^2\left(\hat{Y}_l^{HF}, \mathring{Y}_l^{LF} \right)}{\mathrm{Cov}^2\left(\hat{Y}_l^{HF}, \hat{Y}_l^{LF} \right)} \frac{\mathrm{Var}\left[\hat{Y}_l^{LF} \right]}{\mathrm{Var}\left[\mathring{Y}_l^{LF} \right]}, \tag{6.17}$$

where we clarify a typographical error in the formula given in Geraci et al. (2017). By differentiating (6.17) with respect to γ_l, we find the correlation is maximised when

$$\gamma_l = \frac{\mathrm{Cov}\left(\hat{Y}_l^{HF}, X_{l-1}^{LF} \right) \mathrm{Cov}\left(X_l^{LF}, X_{l-1}^{LF} \right) - \mathrm{Var}\left[X_{l-1}^{LF} \right] \mathrm{Cov}\left(\hat{Y}_l^{HF}, X_l^{LF} \right)}{\mathrm{Var}\left[X_l^{LF} \right] \mathrm{Cov}\left(\hat{Y}_l^{HF}, X_{l-1}^{LF} \right) - \mathrm{Cov}\left(\hat{Y}_l^{HF}, X_l^{LF} \right) \mathrm{Cov}\left(X_l^{LF}, X_{l-1}^{LF} \right)}. \tag{6.18}$$

We adopt this modification in all test cases considered in this chapter and thus replace $\hat{Y}_l^{LF}$ and ρ_l with $\mathring{Y}_l^{LF}$ and $\mathring{\rho}_l$, respectively, in all formulae.

6.2.2.2 MLMF algorithm

We conclude this section by stating the MLMF algorithm used in this chapter in Algorithm 2.

Algorithm 2: Multilevel Multifidelity Monte Carlo method.

1 Estimate the variance and cost of the MLMF estimator, as well as the correlation and cost ratio between the HF and LF models at user-specified levels using an initial estimate for the number of samples. The same set of random numbers must be used for the HF and LF models

2 Start with $L = l_\mu$

3 Define optimal N_l^{HF} using (6.13) and r_l using (6.12) with increased correlation factor (6.17) when required

4 If the optimal N_l^{HF} is greater than the number of samples of the HF and LF models from Step 1, evaluate the extra samples required

5 If the optimal $r_l N_l^{HF}$ is greater than the number of samples of the LF model after Step 4, evaluate the extra samples of LF required

6 If the algorithm has not converged and $L < L_{\max}$, set $L := L + 1$ and return to Step 3

7 If algorithm converged, or $L \geq L_{\max}$, STOP

This algorithm is implemented using our own Python MLMF wrapper constructed around both XBeach and SFINCS, which can be shared on distributed cores of an HPC cluster to increase efficiency. In this wrapper, the models are run and post-processed separately, meaning there is no issue with different input or output formats. Therefore, our MLMF wrapper, like that of MLMC in Chapter 5, is model-independent, meaning it can be easily applied to other models in further work. Note that, given the use of distributed cores, any times quoted in this chapter are the total simulation times multiplied by the number of cores used.

6.3 Applying MLMF to coastal zone test cases

We can now apply the outlined MLMF algorithm to both theoretical and real-world coastal flooding test cases to calculate the expectation of an output variable at multiple locations based on uncertain input data.

6.3.1 Non-breaking wave test case

For our first test case, we consider the 1D case of a non-breaking wave propagating over a horizontal plane from Hunter et al. (2005) and Bates et al. (2010), which has already been simulated using XBeach and SFINCS in Leijnse (2018). The domain is initially dry and the wave is generated by imposing a rising water elevation boundary condition and a constant velocity boundary condition ($u(x = 0, t) = 1\,\mathrm{ms}^{-1}$) at the inlet. Note that this test case can be interpreted as a propagating wet-dry interface but, as in Hunter et al. (2005) and Bates et al. (2010), we refer to it as a wave. In this test case, we evaluate the uncertainty associated with the Manning friction coefficient, n_m, and set this parameter to have the normal distribution $n_m \sim \mathcal{N}(0.03, 0.01)$ $\mathrm{s\,m}^{-1/3}$. Note that, as Manning coefficients must be non-negative, any sampled values below 0 are discarded. The remaining parameters are the same as those in Leijnse (2018) and, in particular, we keep the simulated time at 1 h and the length in the x-direction

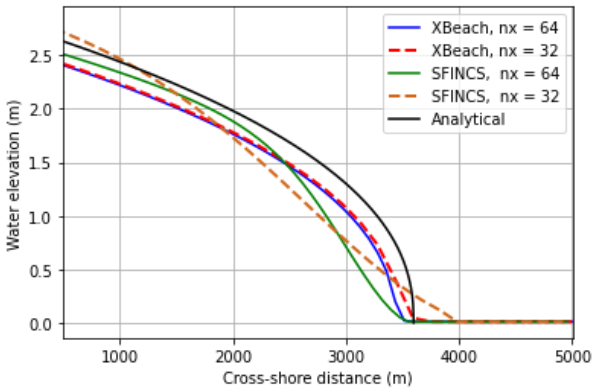


FIGURE 6.1. Comparing the final water elevation from using SFINCS and XBeach at different resolutions with the analytical result for the non-breaking wave test case. A Manning friction coefficient of 0.0364 $\mathrm{s\,m^{-1/3}}$ is used in all simulations.

equal to 5000 m. The quantity of interest is the expected value of the water elevation at the end of the simulation at $x = 1000\,\mathrm{m}$, $x = 1500\,\mathrm{m}$, $x = 2000\,\mathrm{m}$ and $x = 2500\,\mathrm{m}$.

The advantage of this test case is that, due to the flat slope and the constant velocity condition at the inlet, the inviscid shallow water equations can be solved analytically with the following result

$$h(x,t) = \left(-\frac{7}{3} n_m^2 u^2 (x - ut)\right)^{3/7}. \tag{6.19}$$

This means we can get a good estimate of the expected value of the true solution, although we cannot find the 'true' expected value because of the uncertainty in n_m. The full derivation of this result can be found in Hunter et al. (2005), although in (6.19) we clarify a typographical error in that work. Note also that following Hunter et al. (2005) and Leijnse (2018), u is set equal to $1\,\mathrm{m\,s^{-1}}$.

Before running any of the MLMF algorithm, we run a small test using a constant Manning friction coefficient of 0.0364 $\mathrm{s\,m^{-1/3}}$ to compare the final water elevations from SFINCS and XBeach with the analytical result obtained from (6.19), and check they approximately agree. We also check how the output variable varies with grid-size for both models. Figure 6.1 shows that the XBeach results agree more closely with the analytical result than SFINCS', which is to be expected as XBeach is the high fidelity model. Nevertheless, the SFINCS results are not very different from the analytical result, showing that it represents a good choice for the low fidelity model. The figure also shows that both models have better agreement with the analytical result at the coarser level, which is not what we would expect. This is likely due to the error being made up of both model error, the error due to using a simplified model to approximate a real-world problem, and discretisation error, the error arising from using a finite mesh to solve the model equations (see discussion in Section 4.3.1). These two different types of error may have opposite signs and, to some extent, cancel each other out for this particular Manning value. Then, as the level number increases, the discretisation error decreases whilst the model error stays the same, leading to an apparent increase in the total error.

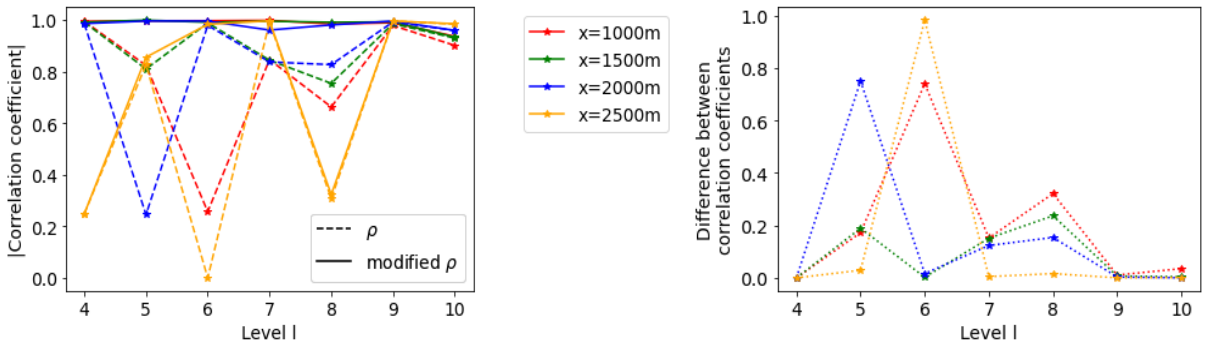
For our MLMF simulation, we use grids with 2^l mesh cells in both SFINCS and XBeach, where the coarsest grid-size is $l = 4$ and the finest is $l = 10$. We first run Step 1 of the MLMF algorithm (Algorithm 2) to

determine key MLMF parameter values at each location. Figure 6.2a shows that SFINCS and XBeach are not well correlated for every output location at every level, but using the modified correlation in (6.17) greatly improves this and, in some cases, results in almost perfect correlation. This large improvement is due to the large variation from unity in the values of the modification factor γ_l shown in Figure 6.2b. The trend in the modified correlation coefficient is very similar to that in the optimum factor of extra LF samples required (see Figure 6.2d) because these extra samples only improve accuracy when the models are highly correlated. Recall that the number of extra LF samples required is $r_l N_l^{HF}$ and, thus, even though r_l is large at finer levels, the general decreasing trend in N_l^{HF} (see Figure 6.6 later) means that, overall, the required number of extra LF samples decreases as the level becomes finer.

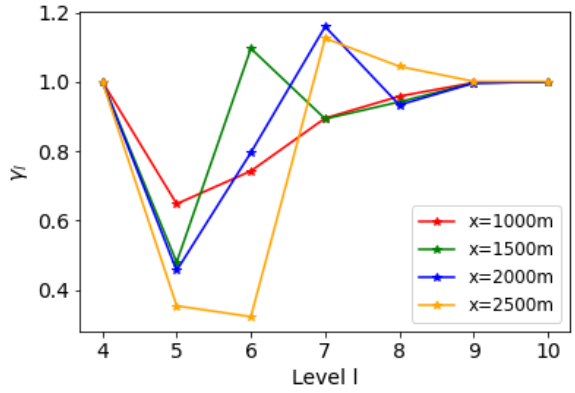
Another key MLMF parameter is the cost ratio between SFINCS and XBeach. Figure 6.2c shows that, as the level number increases, the cost efficiency improvement from using SFINCS over XBeach per model run becomes greater, which is a promising result. Finally, Figure 6.2e shows that the absolute value of α_l is larger at both the coarsest and finest levels. Therefore, at these levels, $\hat{Y}_l^{LF} - \hat{E}\left[Y_l^{LF}\right]$ in (6.6) has a higher weight although, as it is a difference between estimators, this higher weight does not necessarily imply MLMF results will be either closer to those of SFINCS MLMC or further from those of XBeach MLMC.

With these key MLMF parameter values, we can run the next steps of the MLMF algorithm and compare our MLMF results to the analytical result. We also compare our MLMF results with those obtained using the MLMC approach with SFINCS and XBeach separately. Initially, we use a tolerance of $\epsilon = 1 \times 10^{-3}$ in (6.13) and (5.15) to calculate the optimum number of samples to use for MLMF and MLMC, respectively. We highlight here that, throughout this chapter, we use (5.15) to determine the optimum number of MLMC samples, instead of the approximate formulae used in Chapter 5, so that we can fairly compare the optimum number of samples required by both algorithms. Figure 6.3 shows that, in general, the error with respect to the analytical result decreases as the level number increases for the MLMF approach and for both single model MLMC approaches. Furthermore, Figure 6.3a shows that the errors from MLMC with XBeach and MLMF are similar. In contrast, Figure 6.3b shows that MLMC with SFINCS is significantly less accurate than either MLMF or MLMC with XBeach, again justifying our choices of high and low fidelity models.

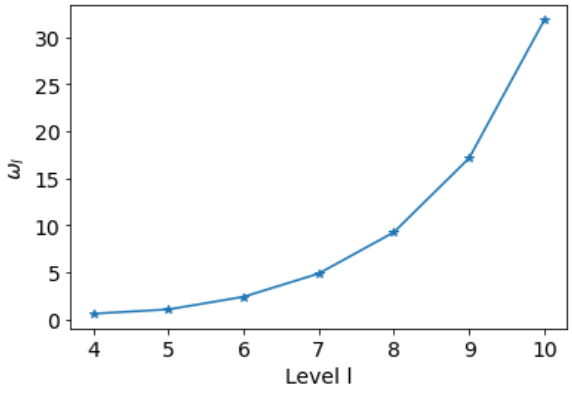
The error to the analytical result shown in Figure 6.3 includes statistical error, model error and discretisation error, but MLMC and MLMF have no effect on model error. Thus, we remove model error by comparing the expected values of MLMF and MLMC to the expected values from using the standard Monte Carlo method with 500,000 simulations of XBeach at the finest resolution considered (1024 mesh cells in the x-direction). Figure 6.4a shows that, as the level number increases, the error to the Monte Carlo result decreases much more uniformly for both MLMF and MLMC with XBeach and that, again, their errors are similar. In contrast, Figure 6.4b shows that this error does not decrease uniformly for MLMC with SFINCS, most likely because model error still exists as the Monte Carlo result is produced using XBeach.



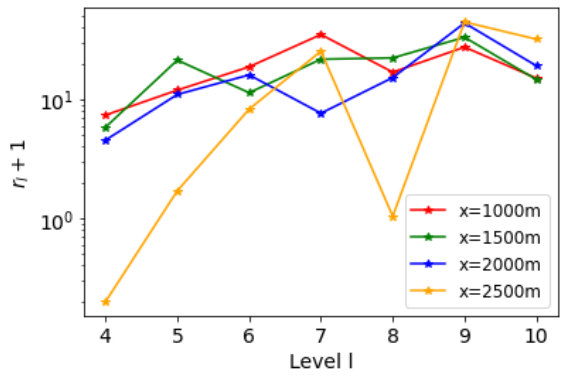
(A) Comparing the real and modified correlation values between SFINCS and XBeach. Left: Absolute value of real correlation, ρ compared to absolute value of modified correlation (6.17). Right: Difference between absolute value of modified and real correlation.



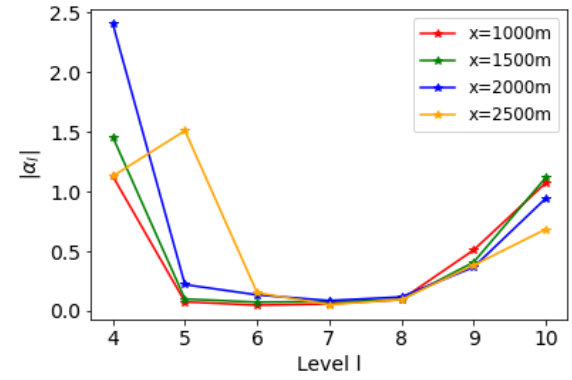
(B) Modification factor, γ_l, (defined by (6.18)) to increase correlation between models.



(C) Cost ratio, ω_l between cost of XBeach (HF) and SFINCS (LF).



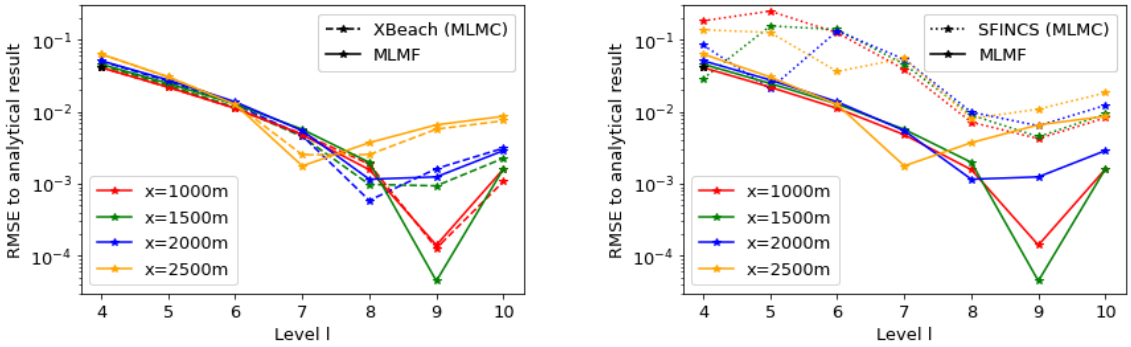
(D) Factor of total LF samples ($r_l + 1$) compared to number of HF samples, where r_l is (6.12).



(E) Value of α_l defined by (6.7) and dependent on the correlation and the variance ratio between HF and LF.

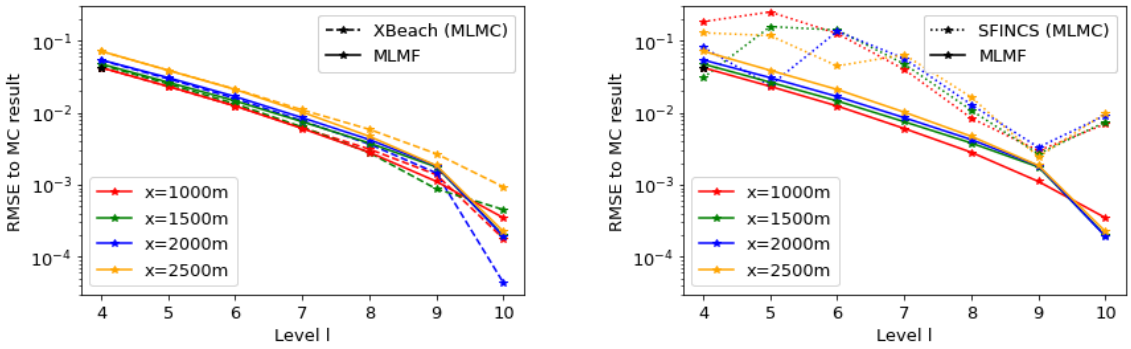
FIGURE 6.2. Behaviour of key MLMF parameters at different levels to determine the final water elevation at specific locations in the non-breaking wave test case. In particular, using the modified correlation (6.17) leads to good correlation at all levels.

All the test case results shown so far in this section use the same tolerance value ($\epsilon = 1 \times 10^{-3}$) and, hence, we re-run the test case using a range of tolerance values. Figure 6.5 shows that, as expected, as ϵ decreases, both the error in the MLMF result and the error in the XBeach MLMC result decrease. In contrast, the error in the SFINCS MLMC result is relatively unaffected by ϵ, again likely due to there being a component of model error included here. More importantly, the MLMF error is almost



(A) Comparing MLMF with MLMC (XBeach). (B) Comparing MLMF with MLMC (SFINCS).

FIGURE 6.3. Error with respect to the analytical result for the final water elevation at the locations of interest as the level number increases (*i.e.* as the resolution becomes finer) in the non-breaking wave test case. Both the error from using MLMF and the error from using MLMC with a single model are shown. Here a tolerance of $\epsilon = 1 \times 10^{-3}$ is used in (6.13) and (5.15) for MLMF and MLMC respectively.

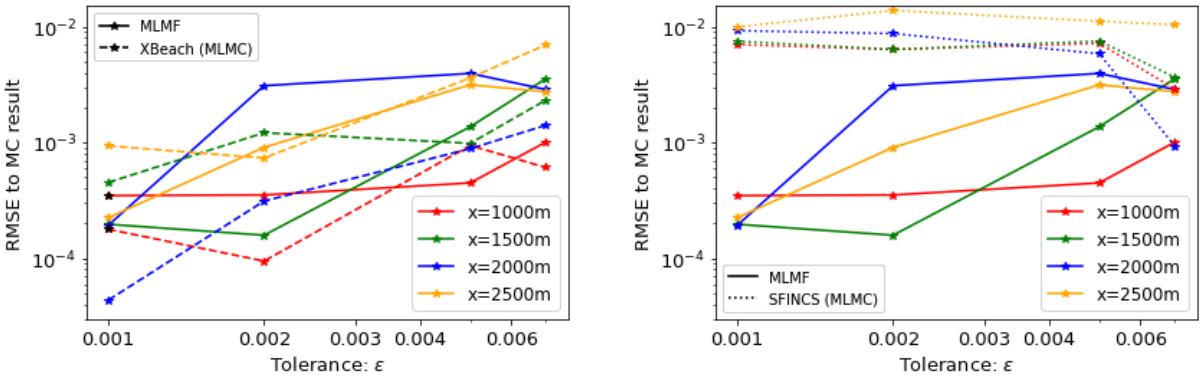


(A) Comparing MLMF with MLMC (XBeach). (B) Comparing MLMF with MLMC (SFINCS).

FIGURE 6.4. Error between the MLMF result and the Monte Carlo (MC) result for the final water elevation at the locations of interest as the level number increases (*i.e.* as the resolution becomes finer) for the non-breaking wave test case. The error from using MLMC with a single model is also shown. Here a tolerance of $\epsilon = 1 \times 10^{-3}$ is used in (6.13) and (5.15) for MLMF and MLMC, respectively.

always lower or of the same order of magnitude as the XBeach MLMC error and both are much lower than the SFINCS MLMC error. Furthermore, Figure 6.6 shows that MLMF achieves this accuracy using significantly fewer HF samples. This figure also shows that, unlike with the test cases in Chapter 5, for large ϵ the optimum number of XBeach samples required for both MLMC and MLMF does not decrease uniformly, but instead increases at level 10 relative to the coarser level 9. However, the number of samples is so small (less than 10 and also less than the total number of processing cores used) that this does not make a significant difference to the computational cost. Thus, overall, Figure 6.6 suggests that notable computational cost savings can be made in this test case by using MLMF and MLMC.

Finally, throughout this section, we have assumed that either we know the 'true' value of the expectations or we can generate a close approximation to it by simulating large numbers of Monte Carlo samples. However, if MLMF is to be of use, we need to apply it to cases where the 'true' value is not known.



(A) Comparing MLMF with MLMC (XBeach).

(B) Comparing MLMF with MLMC (SFINCS).

FIGURE 6.5. Error between the MLMF result and the Monte Carlo (MC) result as the tolerance value ϵ in (6.13) is varied in the non-breaking wave test case. This is compared to the error when varying ϵ in (5.15) for MLMC with a single model.

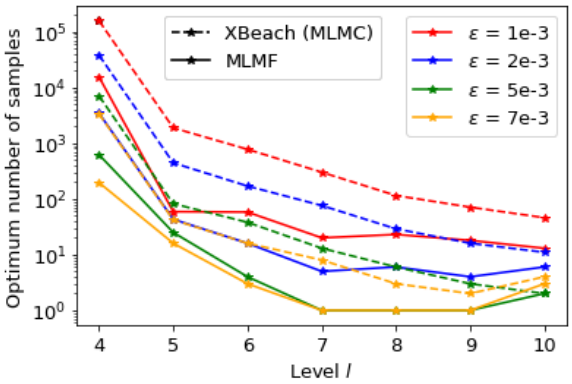


FIGURE 6.6. Optimum number of XBeach (HF) samples required by MLMF (6.13) and MLMC (5.15) for the non-breaking wave test case. The number required by MLMF is always substantially fewer than that required by MLMC.

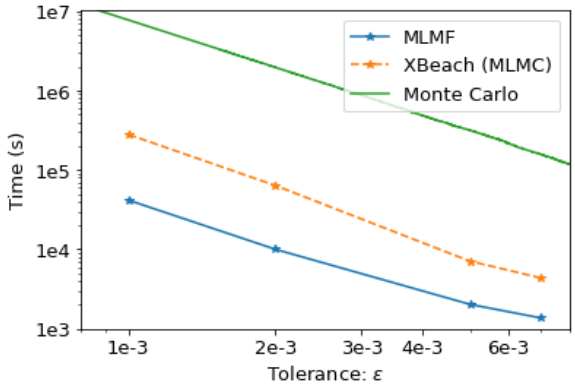


FIGURE 6.7. Comparing the computational cost required to achieve tolerance ϵ using MLMF, XBeach with MLMC and the Monte Carlo method for the non-breaking wave test case.

The main goal of this chapter is to reduce the statistical (sampling) error in a computationally efficient manner and if the true value is unknown, the only parameter the practitioner can use to do this is the tolerance value ϵ in the constraint $\text{Var}\left[\hat{Q}_{M_L}^{HF,CV}\right] < \epsilon^2/2$. Figure 6.7 compares the computational cost required by MLMF, MLMC and the Monte Carlo method to satisfy this constraint. MLMC and MLMF ensure this through the formulae for the optimum number of samples and thus, for these methods, ϵ is plotted against the computational cost for those optimum number of samples. For the Monte Carlo method, the value of ϵ is equal to the square root of twice the variance calculated after each sample is taken, and is plotted against the time taken to run that number of samples. Figure 6.7 shows that, even for this simple test case, MLMF is more than a hundred times faster than Monte Carlo, and on average, five times faster than MLMC with XBeach. This is a very promising result for such a simple 1D test case and suggests MLMF is a good method for improving computational efficiency, whilst still achieving accurate results.

6.3.2 Carrier-Greenspan test case

For our second test case, we consider the 1D Carrier-Greenspan test case, first introduced in Carrier and Greenspan (1958), where a harmonic, non-breaking infragravity wave travels over a plane sloping frictionless beach. This test case is more complex than our first case because it requires the simulation of run-up and run-down, but Leijnse (2018) shows it can be successfully simulated using both XBeach and SFINCS. Following that work, we generate the wave using a varying elevation boundary condition at the inlet, which results in a wave of period 48 s.

In this section, we evaluate the uncertainty associated with the bedslope and assume it has the normal distribution, slope $\sim \mathcal{N}(0.04, 0.02)$. Note that any samples below 0.005 are discarded because otherwise the domain is completely wet. The remaining parameters are the same as those used in Leijnse (2018), in particular, the length in the x-direction is 150 m and the simulated period is 384 s. An advantage of the Carrier-Greenspan test case is that there exists an analytical result. A full derivation of this result is given in Carrier and Greenspan (1958) and is based on solving the dimensionless inviscid shallow water equations where friction is ignored. The analytical result is written in terms of the independent variables

$$\lambda = 2(v + t) \qquad \text{and} \qquad \sigma = 4c, \tag{6.20}$$

where v is the dimensionless velocity, t is dimensionless time and c is dimensionless wave propagation speed. Then, the dimensionless quantities are

$$\begin{aligned} v &= \sigma^{-1}\phi_\sigma(\sigma, \lambda), \\ x &= \frac{\phi_\lambda}{4} - \frac{\sigma^2}{16} - \frac{v^2}{2}, \\ \eta &= \frac{\phi_\lambda}{4} - \frac{v^2}{2}, \\ t &= \frac{\lambda}{2} - v, \end{aligned} \tag{6.21}$$

where $\phi = AJ_0(\sigma)\cos(\lambda)$ and J_0 is the first Bessel function. In order to compare the dimensionless analytical result with the numerical model results, we re-dimensionalise it using the same values as those in the numerical model (bed slope (random sample), initial water depth of 5 m, wave period of 32 s and wave amplitude of 0.5 m).

When a wave runs up a slope, often the quantity of most interest is not the water depth at a particular location in time but, instead, the maximum run-up height. Thus, for this test case, our quantity of interest is the maximum run-up height over the whole 384 s of the simulated period. Unlike in Section 5.3.2.2 which is a complex real-world 2D test case, here the run-up height is simply taken to be the water elevation in the last wet cell in the domain. We first test how the maximum elevation height over the whole domain depends on the resolution and model used in the simulation. Figure 6.8 shows that SFINCS underpredicts the run-up height compared to XBeach and that the coarser resolution also results in an underprediction. It further shows that both models at both resolutions underpredict the run-up height relative to the analytical result and that, whilst the XBeach high resolution result is the most accurate

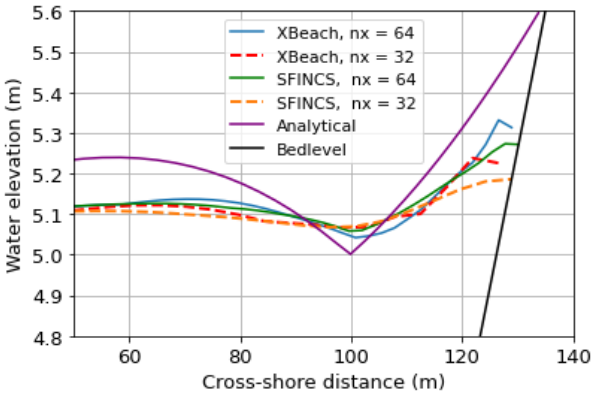
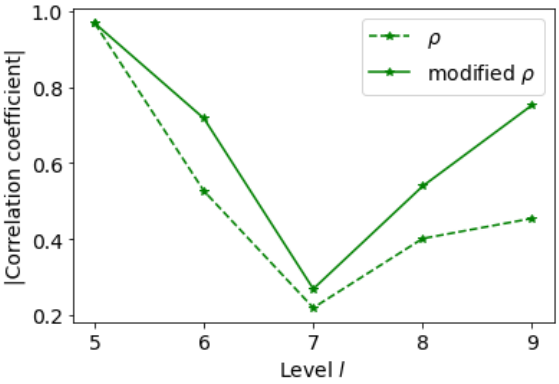


FIGURE 6.8. Comparing the maximum elevation height achieved at every point in the domain over the entire simulation when using SFINCS and XBeach at different resolutions with the analytical result for the Carrier-Greenspan test case. A slope of 0.0682 is used in all simulations. Note that the water elevation height does not always meet the bedlevel because the bedlevel is slightly differently defined at the different resolutions.

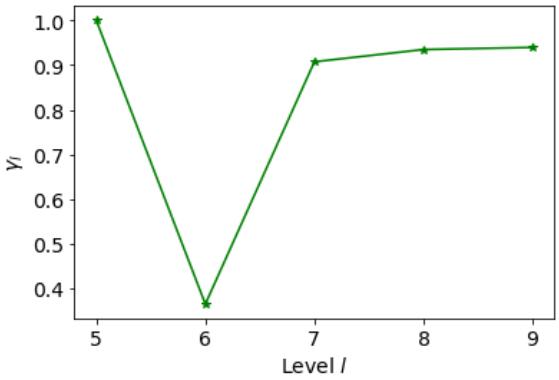
and the SFINCS low resolution result the least accurate, the high resolution result using SFINCS is better than the low resolution result of XBeach. This is a promising outcome for SFINCS and means we can proceed with using it as our low fidelity model in this test case.

For our MLMF simulation, we use grids with 2^l mesh cells in both SFINCS and XBeach, where the coarsest grid-size is $l = 5$ and the finest is $l = 9$. We first analyse the preliminary results from Step 1 of the MLMF algorithm (Algorithm 2), where the values of key MLMF parameters are determined. Figure 6.9 shows how these MLMF parameters vary at different levels and, in particular, Figure 6.9a shows that using the modified correlation formula (6.17) leads to increased correlation between the two models at all levels. This increase is small, though, because Figure 6.9b shows that the optimum value of the modification factor, γ_l, is almost always close to unity (note that by definition $\gamma_l = 1$ at the coarsest level). Using (6.18), this most probably means that there are only small differences between the SFINCS results at the different grid-sizes. Therefore, SFINCS has almost converged and improvements cannot be made by using γ_l to modify the weighting of X_l in (6.15). The exception to this trend in γ_l is at level 6, where a small γ_l causes the sign of the correlation coefficient to change from positive to negative. Note that this large change is not reflected in Figure 6.9a, because we choose to plot the absolute value of the correlation, which is of more interest to this study than the sign of the value.

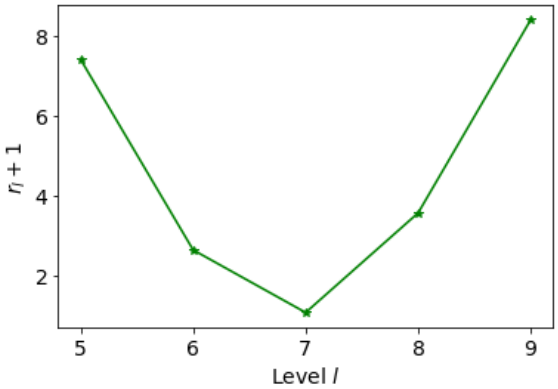
Like with the test case in Section 6.3.1, Figure 6.9c shows that the trend in the modified correlation is similar to the optimum factor of extra LF samples required. This is to be expected because extra samples of the LF model improve accuracy only when the models are highly correlated. Even though r_l is large at finer levels, we recall from the discussion of the key MLMF parameters in Section 6.3.1 that the number of extra samples is given by $r_l N_l^{HF}$ and, therefore, the number of extra samples required actually decreases as the level number increases (see Figure 6.12 later). Finally, Figure 6.9d shows that the absolute value of α_l is largest at finer levels (similar to the trend for r_l).



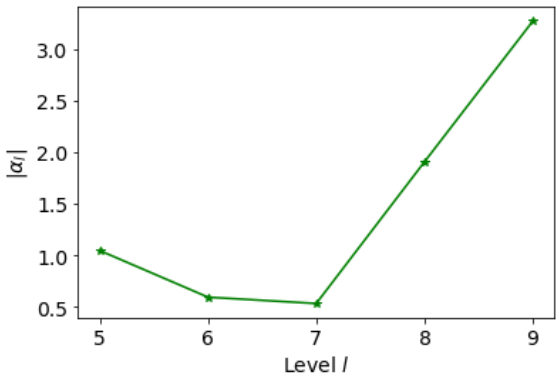
(A) Absolute value of real correlation, ρ between SFINCS and XBeach compared to absolute value of modified correlation (6.17).



(B) Modification factor, γ_l (defined by (6.18)) to increase correlation between models.



(C) Factor of total LF samples ($r_l + 1$) compared to number of HF samples, where r_l is (6.12).

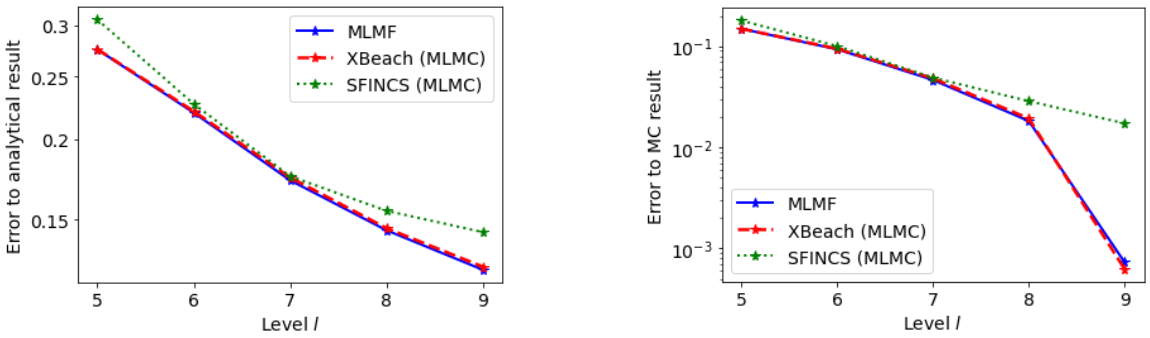


(D) Value of α_l defined by (6.7) and dependent on the correlation and the variance ratio between HF and LF.

FIGURE 6.9. Behaviour of key MLMF parameters at different levels to determine the maximum run-up height in the Carrier-Greenspan test case. Here using the modified correlation (6.17) does not result in large changes to the correlation, most likely because SFINCS has almost converged.

We can now run the next steps in the MLMF algorithm and compare our MLMF results to the analytical result and to the Monte Carlo result estimated using 400,000 samples of XBeach at the finest resolution (512 mesh cells in the x-direction). We also run the MLMC algorithm with SFINCS and XBeach separately. Note that we initially use a tolerance of $\epsilon = 1 \times 10^{-3}$ in (6.13) and (5.15) for MLMF and MLMC, respectively. Figure 6.10a shows that the error to the analytical result decreases uniformly for all methods as the level number increases, with the error using MLMF and MLMC with XBeach lower than that using SFINCS. This justifies that XBeach is the high fidelity model for this test case and shows that the MLMF approach can achieve the same, or lower, error than using only the high fidelity model. Furthermore, Figure 6.10b shows a similar trend for the error to the Monte Carlo result, with the error decreasing uniformly for all methods, and having a similar value for MLMF and MLMC with XBeach.

So far in this section we have only considered a single tolerance value. Therefore, we re-run this test case using different tolerance values ϵ in (6.13) and (5.15) for MLMF and MLMC, respectively. Figure 6.11 shows that the MLMF and XBeach MLMC errors decrease as the tolerance decreases, but that



(A) Error to the analytical result.

(B) Error to the Monte Carlo result.

FIGURE 6.10. Error in the maximum run-up height as the level number increases (*i.e.* as the resolution becomes finer) for the Carrier-Greenspan test case. The errors from using MLMF and from using MLMC with a single model are both shown and a tolerance of $\epsilon = 1 \times 10^{-3}$ is used in (6.13) and (5.15) for MLMF and MLMC, respectively.

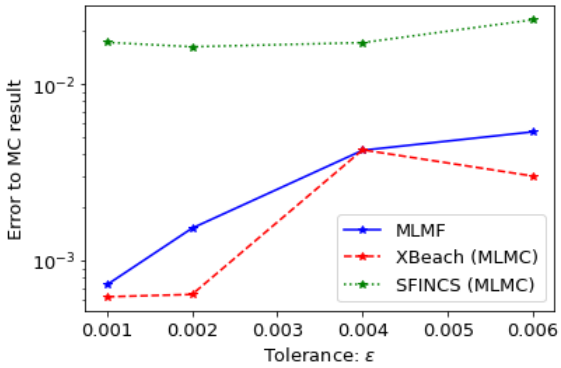


FIGURE 6.11. Error between the MLMF result and the Monte Carlo (MC) result for the Carrier-Greenspan test case as the tolerance value ϵ in (6.13) is varied. This is compared to the error when varying ϵ in (5.15) for MLMC with a single model.

the SFINCS MLMC error is relatively unaffected by ϵ and is high relative to the other errors. Most importantly, this figure shows that MLMF is approximately as accurate as using MLMC with XBeach. Additionally, Figure 6.12 shows that the optimum number of HF samples required by the MLMF algorithm to achieve this accuracy is always less than that required by MLMC. At level 7 (and, to a lesser extent, level 8), the difference between the optimum number of samples required is smaller than at other levels because the correlation between SFINCS and XBeach is lower (see Figure 6.9a), meaning MLMF and MLMC with XBeach are almost equivalent. This highlights the importance of choosing two closely correlated models to ensure optimum efficiency from using MLMF.

As discussed in the previous test case, in reality, the 'true' value of the quantity of interest is not always known and the only parameter available to check statistical accuracy is the tolerance value ϵ. Figure 6.13 compares the computational cost required by MLMF, MLMC and the Monte Carlo method to satisfy the statistical error constraint $\text{Var}\left[\hat{Q}_{M_L}^{HF,CV}\right] < \epsilon^2/2$. As before, for MLMC and MLMF, ϵ is the tolerance and is plotted against the cost required to run the optimum number of samples for this ϵ, whereas for Monte Carlo, ϵ is the square root of twice the variance calculated after each sample and is plotted against

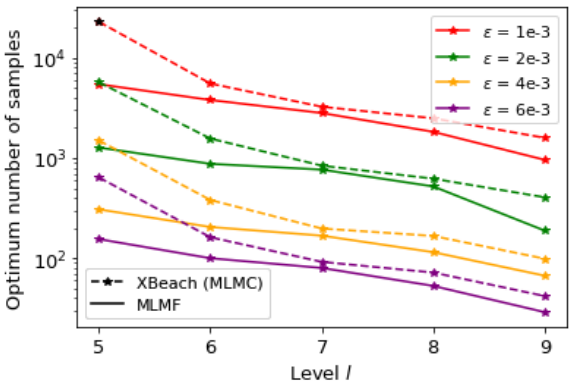


FIGURE 6.12. Optimum number of XBeach (HF) samples required by MLMF (6.13) and MLMC (5.15) for the Carrier-Greenspan test case. The number required by MLMF is always substantially fewer than that required by MLMC.

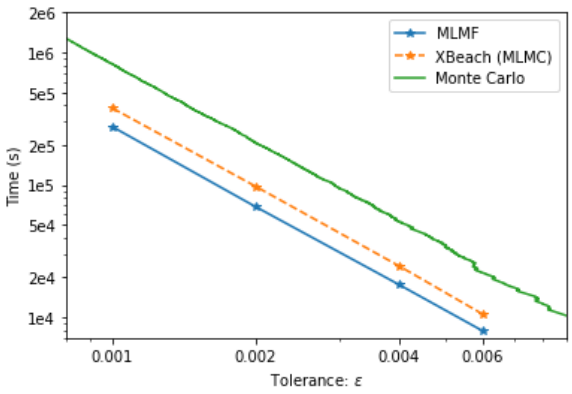


FIGURE 6.13. Comparing the computational cost required to achieve tolerance ϵ using MLMF, XBeach with MLMC and the Monte Carlo method for the Carrier-Greenspan test case.

the time taken to run that number of samples. Thus, Figure 6.13 shows that, compared to the Monte Carlo method, using MLMF is at least three times as efficient for the same tolerance, and MLMC is at least twice as efficient. Whilst these improvements are not as notable as for the previous test case, they nevertheless show that even small differences between the number of optimum HF samples (see Figure 6.12) are sufficient for MLMF to be more efficient than MLMC.

6.3.3 Myrtle Beach test case

The test cases considered so far in this chapter have been relatively simple one-dimensional theoretical test cases. For our final test case, we consider the real-world test case of a dune system near Myrtle Beach, South Carolina, USA (see Figure 6.14). A satellite image of the specific beach of interest is shown in Figure 6.15 and its bedlevel data is shown in Figure 6.16a.

As in the previous test cases, in the MLMF algorithm, we use XBeach as the HF model and SFINCS as the LF model. To simulate the waves in XBeach we use the surfbeat model with the JONSWAP wave spectrum (see Section 2.3.2) and set the significant wave height equal to 4 m, the wave period equal to 12 s, the peak enhancement factor $\mathring{\gamma}$ equal to 3.3 and the main wave angle perpendicular to the shore equal to 124.3°. Note that, as discussed in Section 2.3.3, SFINCS does not explicitly simulate waves and therefore we do not have to define a wave spectrum for it. In order to accurately model waves in XBeach, we need a long stretch of water before the waves reach the beach, which is not present in the domain in Figure 6.16a. Therefore, we extend the domain offshore when running XBeach, as shown in Figure 6.16b, but, for reasons of computational cost, use the original domain in SFINCS. For the larger XBeach domain, we maintain a uniform grid spacing in the original domain region (*i.e.* the region where both XBeach and SFINCS are simulated) so that the grids in each model are the same in that region. In the extended part of the domain, however, we vary the cross-shore grid resolution depending on the bedlevel to make XBeach more computationally efficient, as shown in Figure 6.17a. Note that we use the

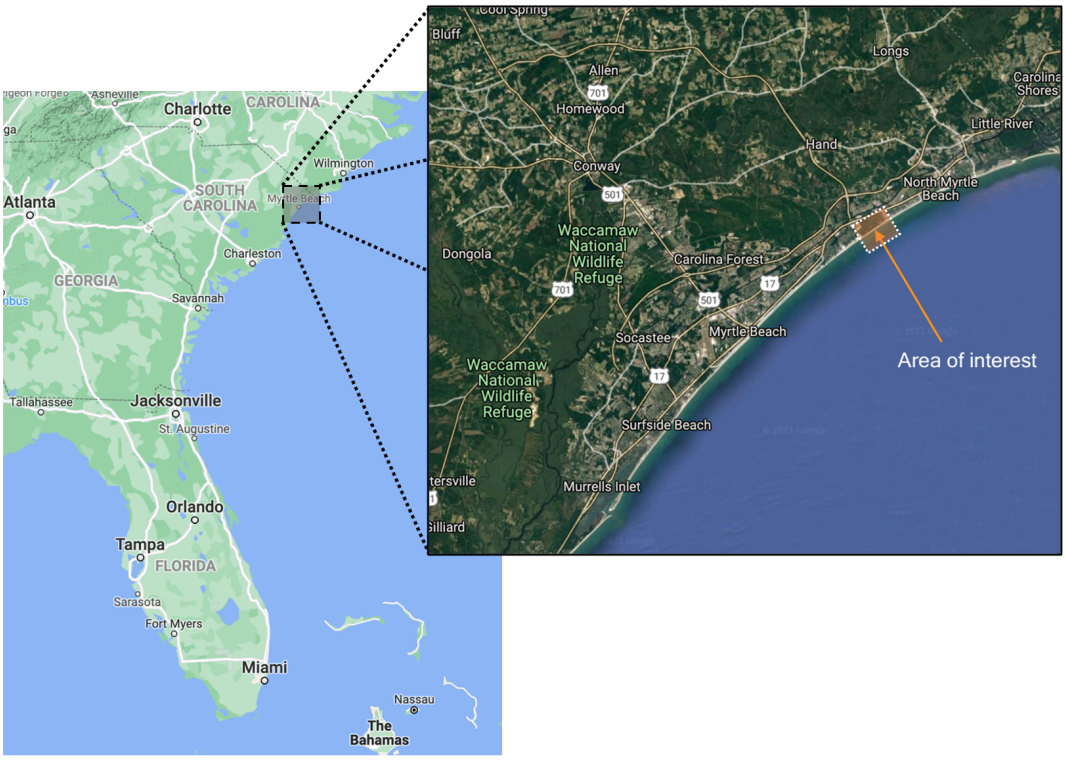
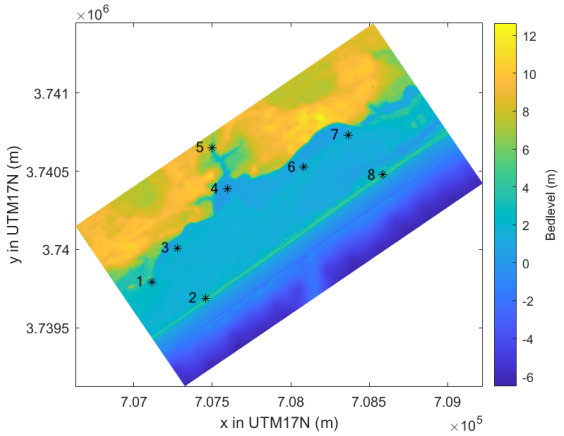


FIGURE 6.14. Location of area of interest in the Myrtle Beach test case. Source: Google Maps.

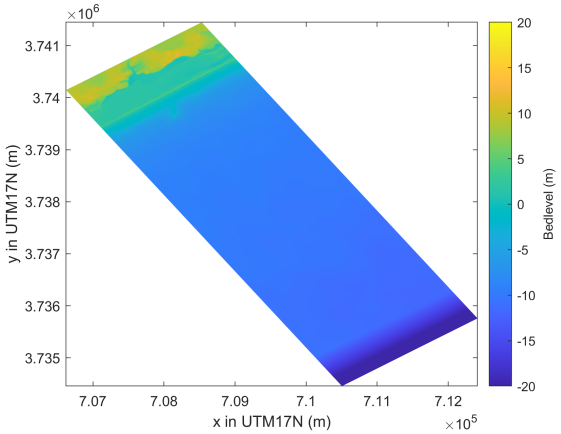
FIGURE 6.15. Satellite image of the beach of interest in the Myrtle Beach test case, where the original domain (shown in more detail in Figure 6.16a) is indicated by an orange rectangle. Source: Geo Image Deltares Toolbox.

cross-shore grid size of the original domain as a lower bound for the grid-size in the extended domain, meaning that the grid in the extended domain also varies at each level, so as to make the cost comparisons between the levels fair. Figure 6.17b shows an example of this grid in XBeach when the cross-shore grid resolution is equal to 2 m.

As this is a real-world study, we must consider tides, as well as waves. These tides can have a large impact on coastal flooding and thus, for this test case, we evaluate the uncertainty in the maximum

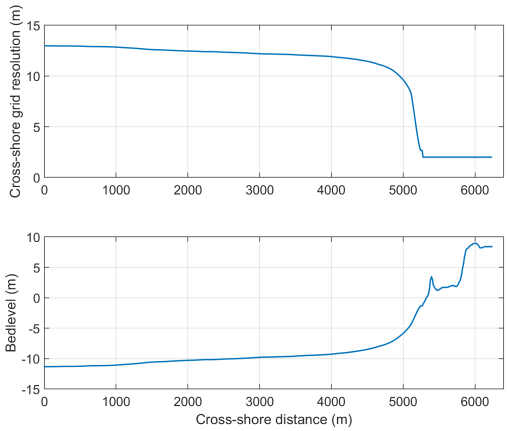


(A) Bedlevel data for the original domain with locations of interest marked with a black *.

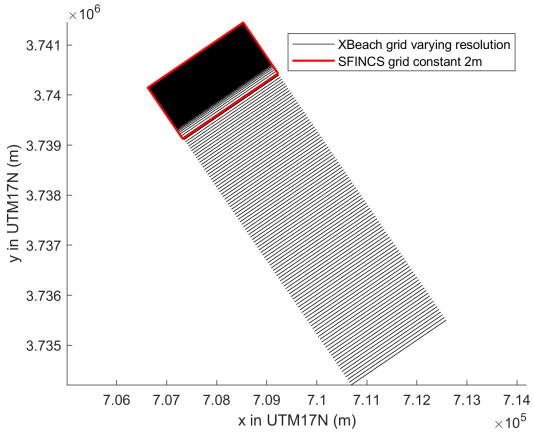


(B) Bedlevel data for the extended domain for XBeach.

FIGURE 6.16. Bedlevel data for the Myrtle Beach test case. Note that x and y are the Universal Transverse Mercator (UTM) co-ordinates for the global zone that Myrtle Beach is located in (17N).



(A) Varying cross-shore grid resolution based on the bedlevel in the extended XBeach domain. Here, for illustration purposes, we show the cross-shore cross-section at the left-hand side edge of the domain.



(B) Comparing grid used in XBeach (in a larger extended domain) to the grid used in SFINCS (in the original domain). Note x and y are the Universal Transverse Mercator (UTM) co-ordinates for the global zone that Myrtle Beach is located in (17N).

FIGURE 6.17. Example of the grid used in XBeach and SFINCS for the Myrtle beach test case with a cross-shore resolution of 2 m in the original domain and a maximum resolution of 2 m in the extended domain.

tide height, h_{tide}. In both SFINCS and XBeach, tides are modelled using a varying elevation height boundary condition at the offshore open boundary and Figure 6.18 shows the way that h_{tide} controls this tidal boundary condition. Note that the tide has a period of 6 h and we run the simulation for 3 h. We assume that the maximum tide height has the distribution $h_{\text{tide}} \sim \mathcal{N}(5, 0.75)$m, and choose the quantity of interest to be the maximum water elevation height at eight different locations in the domain, which are marked with black asterisks in Figure 6.16a. Figure 6.19 shows an example of the maximum elevation height computed by an XBeach simulation for this test case (using the grid in Figure 6.17b) overlaid on a satellite image of the beach. The figure shows there is a clear jump in the maximum elevation height

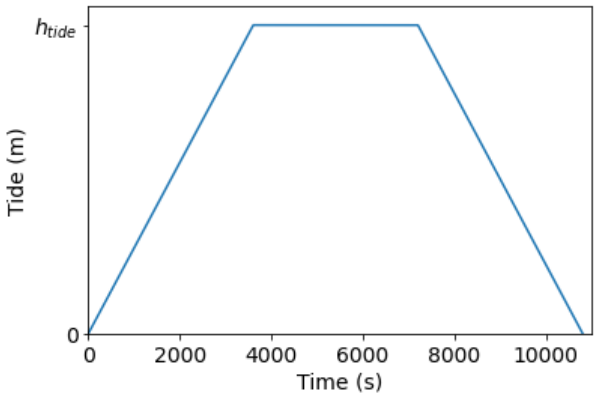


FIGURE 6.18. Tide variation during the Myrtle Beach test case, where h_{tide} is uncertain and has distribution $h_{\text{tide}} \sim \mathcal{N}(5, 0.75)$m.

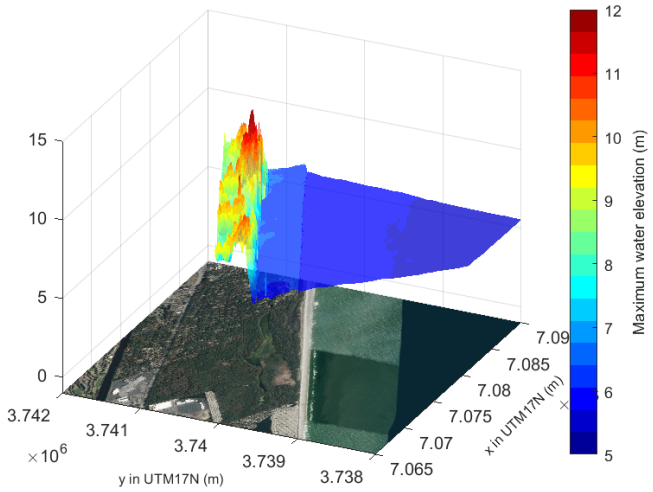


FIGURE 6.19. Maximum elevation height from an example XBeach simulation for the Myrtle Beach test case, showing overtopping. This has been simulated using the grid from Figure 6.17b and $h_{\text{tide}} = 4.97$ m. The maximum elevation height has been overlaid on a satellite image of the location, to highlight the impact of coastal features on the elevation height.

when the water passes the first row of dunes in the system and that, for this particular value of h_{tide}, overtopping occurs.

With the set-up of the test case complete, we now consider the MLMF set-up. For the MLMF simulation, we use grids with $\lceil \frac{155}{4} \times 2^l \rceil$ mesh cells in the cross-shore direction in the original domain, where the coarsest grid-size is $l = 1$ and finest grid-size is $l = 4$. As the cross-shore distance in the original domain is 1240 m, this means the coarsest cross-shore resolution is 16 m and the finest one is 2 m. Note that, throughout this test case, the resolution parallel to the shore is kept equal to 10 m. As a first test, we compare how the values of the variable of interest depend on which model is used and the grid-resolution. In order to be able to distinguish between the results at different locations, Figure 6.20 shows the maximum water depth (*i.e.* maximum water elevation height minus bedlevel) rather than the maximum water elevation height, which is our variable of interest. It shows that SFINCS underpredicts the maximum water depth compared to XBeach and, to a lesser extent, that the coarser resolution also

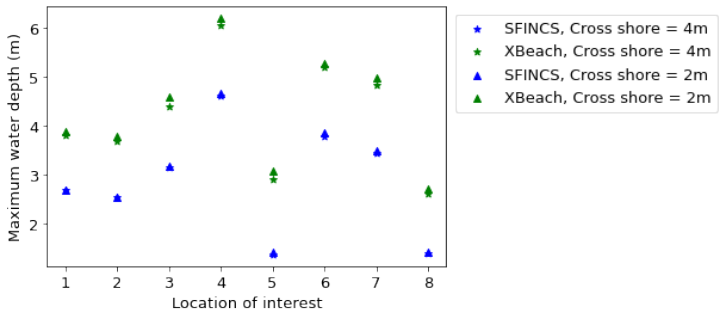


FIGURE 6.20. Comparing the maximum water depth achieved at the eight locations of interest using SFINCS and XBeach at two different resolutions, for the Myrtle Beach test case. A maximum tide height of $h_{\text{tide}} = 4.97\,\text{m}$ is used in all simulations.

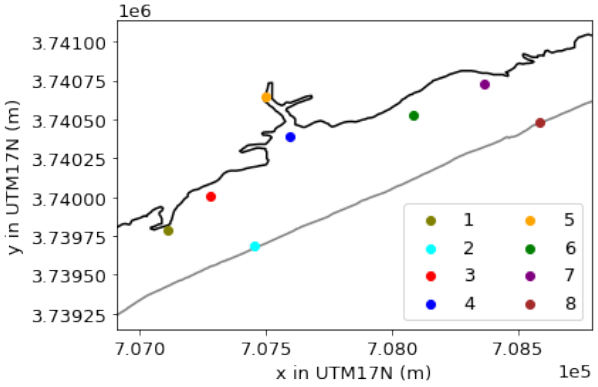


FIGURE 6.21. Locations of interest in the Myrtle Beach test case, with an approximate bedlevel height contour of the shallow cliff edge (black) and the first row of dunes in the system (grey) seen in Figure 6.16a, to help situate them. The locations are colour-coded and these colours are used to represent them throughout this section.

results in an underprediction in both XBeach and SFINCS. The difference between the XBeach and SFINCS results is roughly the same at all locations, and suggests that SFINCS cannot fully capture some processes that the more complex physical and numerical methods in XBeach can. This simple shift is promising though, as it means that the models should be correlated and MLMF can simply adjust for the value shift. Furthermore, the maximum water depth in SFINCS and XBeach follow the same pattern between locations, and thus we can proceed with applying MLMF to this test case. Note that hereafter, in this section, the locations of interest are colour-coded in order to facilitate result interpretation. The colour and placement of all eight locations is shown in Figure 6.21, which also includes an approximate contour of the shallow cliff edge (black) and the first row of dunes in the system (grey) to help situate them in the beach domain.

As with the previous test cases, before running the full MLMF algorithm, we first analyse the values of key MLMF parameters determined in Step 1 of the algorithm (Algorithm 2). Figure 6.22a shows that the modified correlation (6.17) between SFINCS and XBeach generally decreases as the level number increases and that, at the finest level, the correlation is very low at some locations. The right panel of Figure 6.22a shows that the modified correlation method is very beneficial in this test case because it results in a large increase in correlation, especially at level 2. However, at the finest level, it leads

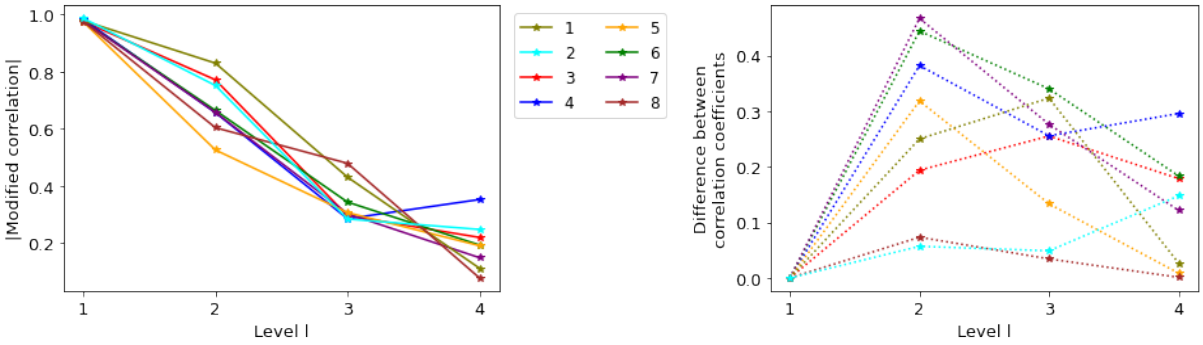
to almost no increase in the low correlation between the models for locations 1, 5 and 8. Figure 6.22b shows that this is also noticeable in the modification factor γ_l (6.18), which at the finest level is close to unity at most locations. As discussed in Section 6.3.2, this likely indicates that there are only small differences in SFINCS results at the two grid-sizes considered in level 4 ($\lceil \frac{155}{4} \times 2^4 \rceil$ and $\lceil \frac{155}{4} \times 2^3 \rceil$ mesh cells), and suggests it has almost converged to a result at these locations. Therefore, any low correlations are due to the different model errors in SFINCS compared to XBeach, rather than to a discretisation error. Locations 1 and 8 are both towards the side edges of the domain and, therefore, the low correlation suggests that XBeach can resolve open boundaries better than SFINCS. Furthermore, location 5 is a difficult point to simulate given it is in an enclosed inlet. The conclusion from Figure 6.22a and 6.22b is therefore that, unsurprisingly, the benefits of the more complex physics implemented in XBeach become greater as the mesh becomes finer.

Additionally, Figure 6.22c shows that the trend in the optimum factor of extra LF samples, r_l, is very similar to the trend in the correlation, with r_l also decreasing as the level number increases, meaning that fewer extra LF samples are required at finer levels. This is a promising result for computational efficiency because, at these finer levels, the model runs are more expensive. Finally, unlike the previous multi-location test case, Figure 6.22d shows there are no consistent similarities in the values of α_l (6.7) at the different locations, except at level 1, the coarsest level considered. This suggests that, at the coarsest level, the flow behaviour is uniform, but that at higher resolutions the flow becomes more complex and non-uniform.

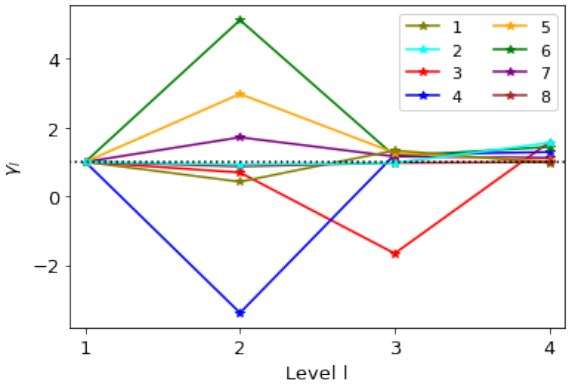
Before running the full MLMF algorithm, we also consider how to assess the accuracy of the MLMF algorithm for this test case. This is a complex computationally expensive real-world problem for which there is no analytical solution and for which approximating a 'true' solution using the standard Monte Carlo method at the finest resolution considered is impractical (each simulation at this resolution takes on average 3 days). Therefore, to assess accuracy, we recall the general theoretical formula for RMSE (5.9) given in Section 5.2

$$\text{RMSE} = \sqrt{\mathbb{E}[(\hat{P} - \mathbb{E}[X_L])^2] + (\mathbb{E}[X_L] - \mathbb{E}[X])^2}, \tag{6.22}$$

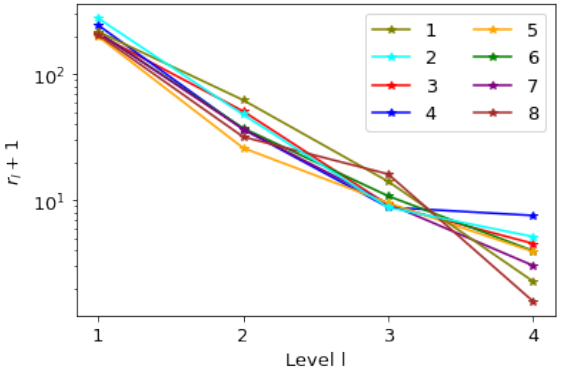
where X is the true solution, X_L is the solution on the finest level (*i.e.* level L), and $\hat{P}$ can be either the MLMF estimator $\hat{Q}^{HF}_{M_l}$ or the MLMC estimator $\hat{Y}^{HF}_{M_l}$. The first term in (6.22) represents the statistical (sampling) error caused by using a Monte Carlo type estimator to represent X_L and is the only term affected by the final MLMC or MLMF estimate (the second term is the numerical discretisation error). Therefore, for this test case, we use the variance of the estimators as a preliminary proxy for the RMSE and use the output generated in Step 1 of the MLMC and MLMF algorithms to estimate them. Figure 6.23 shows how both the MLMF variance (6.9) and MLMC variance (5.6) vary with level l. For both estimators at all locations, the general trend is that the variance decreases as the level number increases. As already discussed in Chapter 5, this is an important result because it means that fewer samples are required on the finer levels. The MLMC variance, however, plateaus and then increases slightly at some locations for the finer levels, whereas the decrease in MLMF is more uniform, indicating that



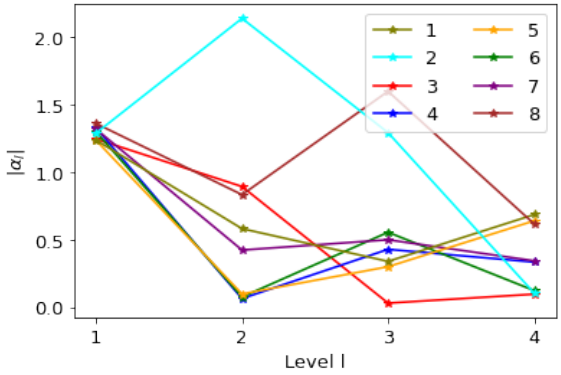
(A) Comparing the real and modified correlation values between SFINCS and XBeach. Left: Absolute value of modified correlation (6.17). Right: Difference between absolute value of modified and real correlation.



(B) Modification factor, γ_l, (defined by (6.18)) to increase correlation between models.



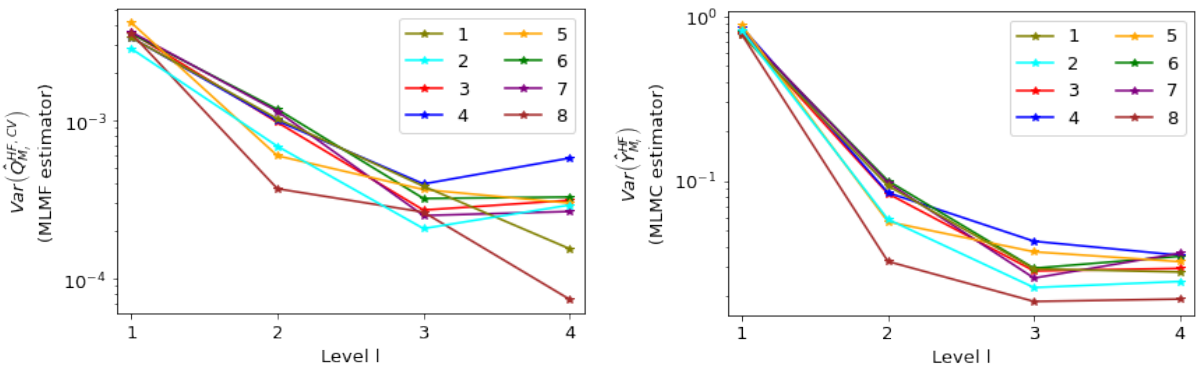
(C) Factor of total LF samples $(r_l + 1)$ compared to number of HF samples, where r_l is (6.12).



(D) Value of α_l defined by (6.7) and dependent on the correlation and the variance ratio between HF and LF.

FIGURE 6.22. Behaviour of key MLMF parameters at different levels to determine the maximum water elevation at eight specific locations in the Myrtle Beach test case. In particular, the modified correlation (6.17) starts high but decreases with the level number, as the benefits of the more complex physics implemented in XBeach become greater.

MLMF is performing better than MLMC for this test case. More importantly, the variance of the MLMF estimator is two orders of magnitude smaller than that of the MLMC estimator. Thus, using the RMSE formula (6.22), MLMF is more accurate than MLMC, although this is difficult to determine without an approximation to the 'true' solution. The smaller variance also means that MLMF will require fewer higher fidelity samples than MLMC and, therefore, be more computationally efficient.



(A) Variance for MLMF estimator

(B) Variance for MLMC estimator using XBeach

FIGURE 6.23. Comparing the different rates at which the variance of the MLMF estimator (6.9) and the variance of the MLMC estimator for XBeach (5.6) decrease as the level number increases (*i.e.* as the resolution becomes finer). Note the MLMF variance is two orders of magnitude smaller than the MLMC variance.

Given these promising results, we can now run the full MLMF algorithm (Algorithm 2) using a tolerance of $\epsilon = 3 \times 10^{-2}$ in (6.13). Figure 6.24 compares how the values of the MLMF and MLMC expectation estimators change as each level l is added to the estimator value. The largest change in the estimators occurs from level 1 to level 2, after which the changes from adding an extra level get progressively smaller. This signifies that both MLMF and MLMC are working as expected and satisfying condition (i) in the key MLMC theorem (Theorem 1) in Section 5.2. At most locations, the final MLMF expectation estimator value agrees with that of MLMC, giving us confidence in the accuracy of both algorithms. An exception occurs at location 1, which is one of the locations where the correlation between the two models is lowest, possibly because of the different way in which SFINCS and XBeach treat open boundaries. However, where the estimator values differ, our variance analysis in Figure 6.23 suggests the MLMF value is more accurate than the MLMC value. For ease of interpretation, Figure 6.25 shows the spatial representation of the final expected value estimated using MLMF at the locations of interest. It shows that the expected maximum elevation height grows as we move inland, especially as the water gets funneled into the inlet. This is a physically realistic result and therefore gives us further confidence in the accuracy of our MLMF algorithm.

The optimum number of samples required to estimate the expected values in Figure 6.24 is shown in Figure 6.26a. The figure shows that, at all locations, this number decreases as the level number increases for both MLMF and MLMC, where the optimum number of HF samples is calculated using (6.13) and (5.15), respectively, with $\epsilon = 3 \times 10^{-2}$. This is an important result for computational efficiency as the 2D nature of this test case means that simulations on the finest resolution are much more expensive than those on the coarsest resolution. An interesting result from Figure 6.26a is that locations 2 and 8 require the least number of HF samples in both the MLMC and MLMF algorithms, whilst locations 4 and 5 require the most. When the locations of interest are offshore, and locations 2 and 8 are the furthest offshore, the water elevation there is relatively certain. In contrast, locations 4 and 5 are further inland – at the inlet and behind the dune system – and predicting elevation height at these locations is more

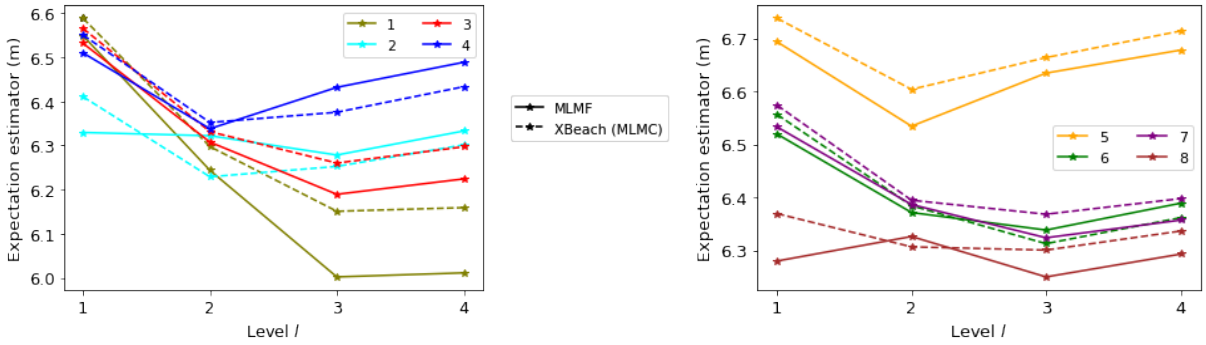


FIGURE 6.24. Comparing how the values of the expectation estimators $\hat{Q}_{M_l}^{HF,CV}$ (MLMF) and $\hat{Y}_{M_l}^{HF}$ (MLMC) change as each level is added to the estimator. These results are generated using $\epsilon = 3 \times 10^{-2}$ in (6.13) and (5.15) for MLMF and MLMC, respectively, and shown for all locations of interest in the Myrtle Beach test case.

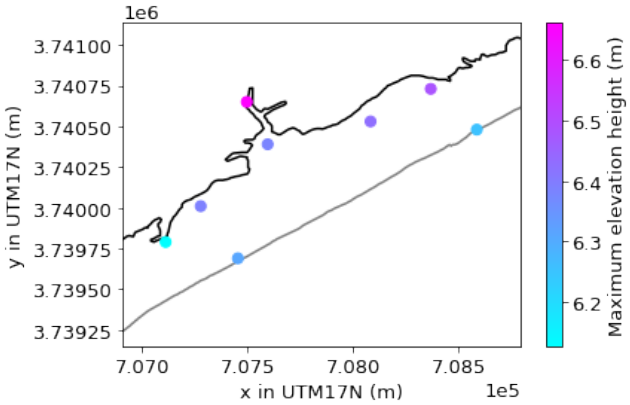
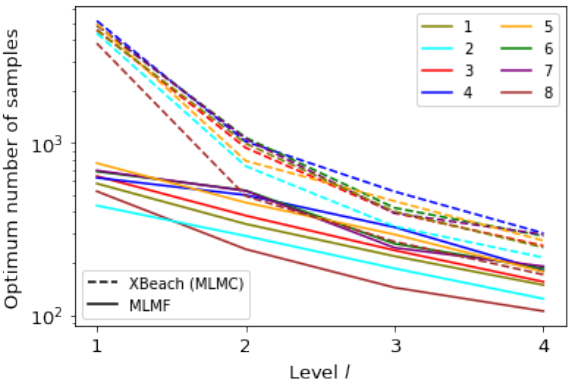


FIGURE 6.25. Spatial representation of the expected value of the maximum elevation height estimated using MLMF with a tolerance of $\epsilon = 3 \times 10^{-2}$ in (6.13).

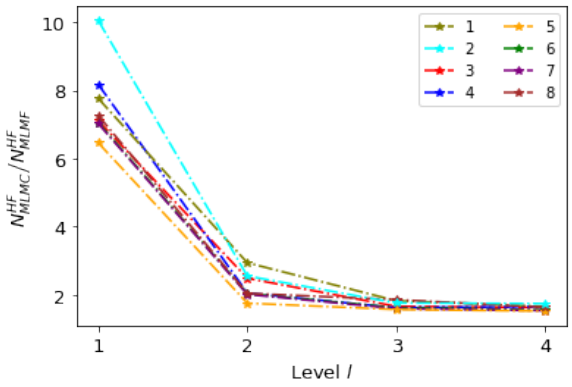
uncertain because it is dependent on the amount of overtopping that has occurred, hence larger numbers of samples are needed to ensure accuracy.

More significantly, Figure 6.26a shows that MLMF always requires fewer XBeach (HF) samples than MLMC, with the biggest difference being at level 1, where MLMF requires an order of magnitude fewer samples. This difference can clearly be seen in Figure 6.26b which shows the number of HF samples required by MLMC divided by the number of those required by MLMF. As the level number increases, the ratio decreases due to the lower correlation between the two models at the finer levels, as seen in Figure 6.22a. Nevertheless, even at the finest level, MLMC still requires twice as many samples as MLMF, which is particularly significant given how computationally expensive the test case is at these finer resolutions.

As in previous test cases, we also consider how the optimum number of samples varies for different tolerance values ϵ in (6.13). Using location 5 as an example, Figure 6.27 shows that, for all ϵ values, the optimum number of HF samples required decreases as the level increases, which represents the same trend as observed in Figure 6.26a. Again, MLMF always requires fewer samples than MLMC, and the largest difference is at level 1 where, for all ϵ values, MLMF requires an order of magnitude fewer samples than MLMC. Figure 6.28 shows that the difference in the optimum number of HF samples required translates



(A) Optimum number of XBeach (HF) samples required by MLMF (6.13) and MLMC (5.15).



(B) Optimum number of XBeach (HF) samples required by MLMC divided by the optimum number required by MLMF.

FIGURE 6.26. Comparing the optimum number of XBeach (HF) samples required by the MLMF and MLMC estimators for the Myrtle Beach test case. Here $\epsilon = 3 \times 10^{-3}$ in (6.13) and (5.15) for MLMF and MLMC, respectively.

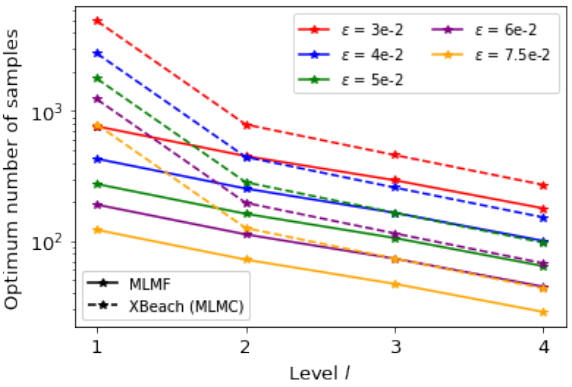


FIGURE 6.27. Optimum number of XBeach (HF) samples required by MLMF (6.13) and MLMC (5.15) at location 5 of the Myrtle Beach test case for varying tolerance ϵ. The number required by MLMF is always substantially fewer than that required by MLMC.

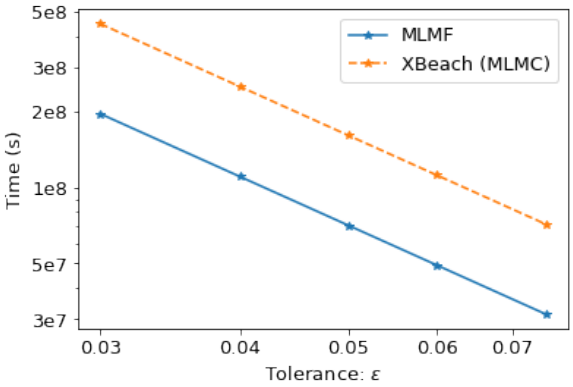


FIGURE 6.28. Comparing the computational cost required to achieve tolerance ϵ at all locations using MLMF and XBeach with MLMC for the Myrtle Beach test case.

to MLMF being more than twice as efficient as MLMC for the same level of statistical accuracy. For such a complex real-world test case, this is a notable result and the speed-up versus the standard Monte Carlo method would undoubtedly be even greater given the computational cost of the fine level runs. Although it is impractical to conduct a full Monte Carlo analysis, the simulations we did conduct at the finest resolution suggest that, in the time it takes MLMF to achieve a tolerance of $\epsilon = 3 \times 10^{-2}$, we would only be able to conduct 600 Monte Carlo runs. We estimate the latter would result in an ϵ value of order 1, *i.e.* an ϵ value two orders of magnitude worse than that achieved by MLMF. Therefore, this test case concretely demonstrates that applying MLMF means we can conduct uncertainty analysis in complex real-world problems in an accurate and efficient manner, which would have been impossible using the standard Monte Carlo methods previously applied in this field.

6.4 Cumulative distribution functions

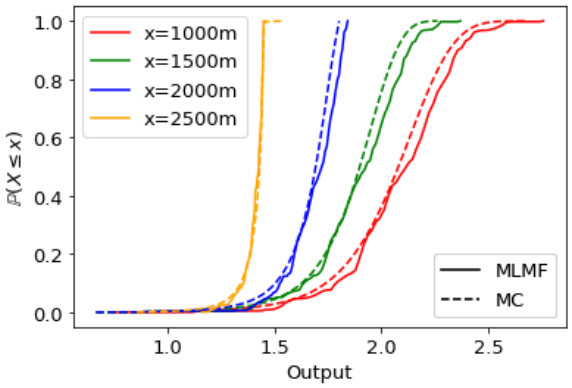
The objective of the MLMF framework, like with the MLMC framework, is to find the expectation of the output variable of interest. However, the probability/risk of a variable exceeding a certain value is often also valuable to practitioners. We can assess this probability by using the MLMF outputs to generate a cumulative distribution function (CDF) using a modified version of the inverse transform sampling method from Gregory and Cotter (2017), as previously outlined in Section 5.4. Recall from the latter that the method evaluates the inverse cumulative distribution function $F^{-1}(u)$, where $u \sim \mathcal{U}[0,1]$. If F is strictly increasing and absolutely continuous, then $x \equiv F^{-1}(u)$ is unique. A simple consistent estimate for x can then be found by sorting the samples such that $X^1 < X^2 < ... < X^N$ and then $\hat{F}^{-1}(u) = X^{\lceil N \times u \rceil}$ converges in probability to x as $N \to \infty$ (see Section 5.4). Modifying the inverse cumulative distribution function for MLMC (5.29), the MLMF version is then

$$
\begin{aligned}
F_L^{-1}(u) \simeq & R^{HF}(X)_{l_\mu}^{\lceil N_{l_\mu}^{HF} \times u \rceil} + \alpha_{l_\mu} \left(\mathring{R}^{LF}(X)_{l_\mu}^{\lceil N_{l_\mu}^{HF} \times u \rceil} - \hat{E}\left[\mathring{Y}_l^{LF} \right] \right) \\
& + \sum_{l=l_\mu+1}^{L} \left(R^{HF}(X)_l^{\lceil N_l^{HF} \times u \rceil} - R^{HF}(X)_{l-1}^{\lceil N_{l-1}^{HF} \times u \rceil} \right) \\
& + \sum_{l=l_\mu+1}^{L} \alpha_l \left(\mathring{R}^{LF}(X)_l^{\lceil N_l^{HF} \times u \rceil} - \mathring{R}^{LF}(X)_{l-1}^{\lceil N_l^{HF} \times u \rceil} - \hat{E}\left[\mathring{Y}_l^{LF} \right] \right),
\end{aligned}
\tag{6.23}
$$

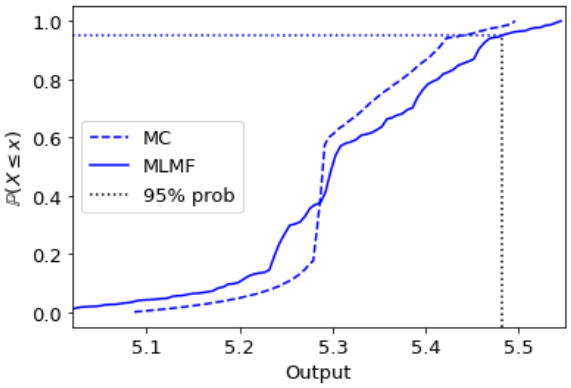
where $R^{HF}(X)_l^i$ and $\mathring{R}^{LF}(X)_l^i$ represent the i^{th} order statistic of X_l on each level l of the high fidelity model and modified correlation low fidelity model (see 6.15), respectively. Note that there cannot be exact cancellation because using this method means the approximations at each level are no longer unbiased.

6.4.1 Applying inverse transform sampling to MLMF outputs

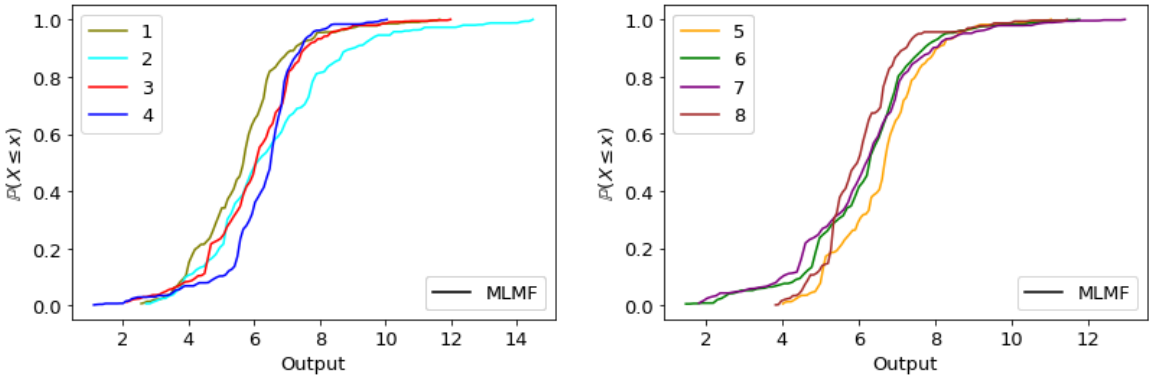
In this section, we apply the modified inverse transform sampling method (6.23) to the MLMF outputs from each of the test cases in Section 6.3. We thus produce a CDF for each different variable of interest, which can then be used to assess the risk of extreme flooding occurring. To generate the MLMF outputs, note that we use the lowest tolerance values considered for each test case. In Figures 6.29a and 6.29b, we demonstrate the accuracy of the MLMF-generated CDFs by showing that they agree with the Monte-Carlo-generated CDFs for the non-breaking wave and Carrier-Greenspan test cases, respectively. In Table 6.1, we quantify this agreement by calculating the L^2 error norm between the two CDFs. Note that, as in Section 6.4, we evaluate the CDFs at 100 equally spaced points and, therefore, the implementation of this norm is equivalent to calculating the RMSE between the two CDFs. The small error norms in Table 6.1 give us confidence in our new modified inverse transform sampling method's ability to accurately generate CDFs. In particular, they give us confidence in the accuracy of the MLMF-generated CDFs for the Myrtle Beach test case shown in Figure 6.29c, for which a Monte Carlo simulation is impractical to run (see Section 6.3.3).



(A) MLMF CDF compared with Monte Carlo (MC) CDF for the non-breaking wave test case (Section 6.3.1.



(B) MLMF CDF compared with Monte Carlo (MC) CDF for the Carrier-Greenspan test case (Section 6.3.2.



(C) MLMF CDF for the Myrtle Beach test case (Section 6.3.3.)

FIGURE 6.29. CDFs generated from MLMF outputs using the modified inverse transform sampling method compared with those generated using Monte Carlo (MC) outputs, where possible.

TABLE 6.1. L^2 error norm between the MLMF and Monte Carlo CDFs for the non-breaking wave and Carrier-Greenspan test cases.

Test Case	L^2 **Error Norm**
Non-breaking wave test case - 1000 m	3.55×10^{-2}
Non-breaking wave test case - 1500 m	4.53×10^{-2}
Non-breaking wave test case - 2000 m	5.99×10^{-2}
Non-breaking wave test case - 2500 m	2.79×10^{-2}
Carrier-Greenspan test case	7.63×10^{-2}

These CDFs can be used to readily assess risk and thus greatly add to our understanding of the test cases as, for example, Figure 6.29b shows that the probability the run-up height will exceed 5.48 m is 5%. This figure could then be used to inform a local authority that it would be unwise to place a permanent building structure below this height, but a temporary beach structure might be ok. Taking a different example from the real-world Myrtle Beach test case, Figure 6.29c informs that there is a small but significant probability of the maximum elevation height at the inlet (location 5) exceeding 10 m. This is despite the fact that the expected value is only 6.66 m (see Figure 6.25), which might have led the local authority to

believe that they were safe from a 10 m elevation height. This illustrates how important it is for assessing the risk of extreme flooding that our MLMF algorithm can accurately and efficiently calculate both the expected values and CDFs of output variables.

6.5 Conclusion

In this chapter, we have presented the first successful application of MLMF in the coastal engineering field and one of the first successful applications of this method in any field. Using both theoretical and real-world test cases, we have shown that MLMF can significantly improve computational efficiency for the same or better statistical accuracy compared to the standard Monte Carlo method. In particular, we have demonstrated that this enables uncertainty analysis to be conducted in real-world coastal environments which would have been practically unfeasible with the statistical methods previously applied in this field. Using our new modified inverse transform sampling technique, we are also able to accurately generate the cumulative distribution function (CDF) for output variables of interest, which is of great value to decision makers when assessing risk. Furthermore, the expected values and CDFs of output variables can be computed at multiple locations simultaneously with no additional computational cost, demonstrating the flexibility of MLMF. In future work, this will enable the construction of large risk maps showing the expected value and CDF of variables of interest at all locations in the domain, facilitating accurate and timely decision making.

Finally, we have highlighted the benefits of using a multifidelity approach and shown that using SFINCS as a low fidelity model and XBeach as a high fidelity model makes MLMF notably more computationally efficient than MLMC for the same or higher accuracy. Multifidelity approaches thus represent a very rewarding avenue for further research and, in future work, we will extend the MLMF approach to include our *Thetis* model as an additional higher fidelity model which we expect will improve the overall accuracy of the results.

Code availability

The relevant code for the MLMF framework presented in this chapter is stored at `https://github.com/mc4117/MLMF_coastal`.

CHAPTER 7

Conclusion

Assessing the risk from coastal flooding and erosion is vital for the many communities that live in and rely on coastal zones. In this thesis, we have developed a series of novel advanced numerical and statistical techniques which improve the accuracy, decrease the computational cost and quantify the uncertainty in hydro-morphodynamic models used to simulate flooding and sediment transport.

The first part of this thesis (Chapters 2–4) focused on developing and applying a new 2D depth-averaged coupled hydro-morphodynamic model within the finite element based coastal ocean model *Thetis* utilising DG-based discretisations. This model was shown to be robust and accurate for problems in both fully wet and wet-dry domains, as well as being more efficient and robust than some widely-used models.

The potential for this model to make significant novel contributions to the complex problem of modelling sediment transport was further enhanced by the addition of novel capabilities, which are likely to be highly valuable to a variety of users and stakeholders. Firstly, we developed the first freely available adjoint hydro-morphodynamic framework. This framework is capable of calibrating, inverting and performing sensitivity analysis for any uncertain parameter with respect to any output functional. Moreover, as both the dimension and the number of parameters have little effect on the computational cost, our framework can assess uncertainty in multiple and/or spatially varying uncertain parameters in a single model run, making it more practical and efficient than other standard approaches. The knowledge that can be gained by using our framework is key to improving the understanding of coastal and fluvial hazards, in particular tsunami hazards, hence contributing to mitigating their impacts.

Secondly, we presented the first application of mesh movement to a depth-averaged hydro-morphodynamic model. This implementation improves the accuracy, efficiency and robustness of our model, particularly in cases with a moving wet-dry interface. Thus, we can accurately and efficiently simulate more complex test cases, whose simulation had previously been impractical. Furthermore, the improved model efficiency means model calibration can be more readily conducted as more simulations can be performed for an equivalent cost.

In future work, we will combine the moving mesh and adjoint frameworks. This will mean we can use meshes with smaller numbers of elements in the adjoint framework, decreasing the computational and memory costs, and allowing the adjoint framework to be applied to larger and more complex problems. Furthermore, integrating the two methods will also mean we can use the adjoint method to accurately determine the optimum parameters for the moving mesh framework. This will lead to even greater gains in model accuracy and efficiency. Moreover, this framework will be particularly suited to the inversion of historical tsunami wave profiles from sediment deposits, thus improving understanding of tsunami events

(Dourado et al., 2021), as well as to the inversion of complex underwater friction fields which are crucial for organisations such as the National Oceanographic Centre to obtain accurate model forecasts (Warder et al., 2021).

The second part of this thesis (Chapters 5–6) focused on developing statistical techniques to assess uncertainty in hydro-morphodynamic models. We applied both the *multilevel* Monte Carlo method (MLMC), which accelerates the standard Monte Carlo method by using a hierarchy of resolution levels, and the *multilevel multifidelity* Monte Carlo method (MLMF), which combines the MLMC approach with a multi-model multifidelity approach. This is the first application of MLMC in the coastal engineering field and one of the first applications of MLMF in any field. Both these methods accurately estimate the expected value of an output variable and its distribution in theoretical and real-world test cases in a much more computationally efficient manner than the standard Monte Carlo method, with MLMF being more efficient than MLMC in all test cases considered. This speed-up means practitioners can now perform crucial uncertainty analysis that was previously unfeasible to run.

In this thesis we focused on using Monte Carlo methods to predict average expected outputs based on uncertain input parameters. However, understanding extreme events is also critical, especially for long-term coastal protection. One way to understand extreme events is to use the Markov chain Monte Carlo method (Behrens et al., 2004; Renard et al., 2006; Gaume et al., 2010; Arif et al., 2021). Therefore, in future work, we will apply the multilevel Markov chain Monte Carlo method (MLMCMC) to specifically sample for these events. Using our knowledge of the gains that can be made using multifidelity approaches, we will extend MLMCMC to develop a multilevel multifidelity Markov chain Monte Carlo method. We do not believe such a method has been applied to extreme event quantification and, thus, this represents an exciting avenue for future research with benefits to a wide range of fields.

In conclusion, this thesis has made significant novel contributions to solving the complex problem of hydro-morphodynamic modelling. This research is particularly timely due to the increased risk to coastal zones as a result of climate change. As the latter's impacts become more severe, the importance of hydro-morphodynamic modelling will only increase. The comprehensive toolkit of techniques developed in this work provides a crucial foundation for future researchers and stakeholders seeking to assess and mitigate coastal erosion and flooding risks in an accurate and efficient manner.

Bibliography

Allsop, N.W.H., Pullen, T., Bruce, T., van der Meer, J.W., Schüttrumpf, H., Kortenhaus, A., 2007. EurOtop Wave Overtopping of Sea Defences and Related Structures: Assessment Manual. Boyens Medien GmbH & Co. KG.

Alnæs, M.S., Logg, A., Ølgaard, K.B., Rognes, M.E., Wells, G.N., 2014. Unified form language: A domain-specific language for weak formulations of partial differential equations. ACM Transactions on Mathematical Software (TOMS) 40, 1–37.

Amoudry, L.O., 2008. A review on coastal sediment transport modelling. Proudman Oceanographic Laboratory, Liverpool, UK.

Amoudry, L.O., Souza, A.J., 2011. Deterministic coastal morphological and sediment transport modeling: A review and discussion. Reviews of Geophysics 49.

Andrews, D.G., McIntyre, M.E., 1978. An exact theory of nonlinear waves on a lagrangian-mean flow. Journal of fluid Mechanics 89, 609–646.

Apel, H., Thieken, A.H., Merz, B., Blöschl, G., 2004. Natural Hazards and Earth System Sciences Flood risk assessment and associated uncertainty. Natural Hazards and Earth System Sciences 4, 295–308.

Apsley, D.D., Stansby, P.K., 2008. Bed-load sediment transport on large slopes: Model formulation and implementation within a rans solver. Journal of Hydraulic Engineering 134, 1440–1451.

Arif, M., Khan, F., Ahmed, S., Imtiaz, S., 2021. Evolving extreme events caused by climate change: A tail based bayesian approach for extreme event risk analysis. Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability , 1748006X21991036.

Asbeck, W.F., Ferguson, H.A., Schoemaker, H.J., 1953. New designs of breakwaters and seawalls with special reference to slope protection, in: Proc. 18th Int. Nav. Congress, Rome, Sect, p. 174.

Athanasiou, P., van Dongeren, A., Giardino, A., Vousdoukas, M.I., Ranasinghe, R., Kwadijk, J., 2020. Uncertainties in projections of sandy beach erosion due to sea level rise: an analysis at the european scale. Scientific reports 10, 1–14.

Avdis, A., Candy, A.S., Hill, J., Kramer, S.C., Piggott, M.D., 2018. Efficient unstructured mesh generation for marine renewable energy applications. Renewable Energy 116, 842–856.

Awanou, G., 2015. Quadratic mixed finite element approximations of the Monge–Ampère equation in 2d. Calcolo 52, 503–518.

Balzano, A., 1998. Evaluation of methods for numerical simulation of wetting and drying in shallow water flow models. Coastal Engineering 34, 83–107. doi:10.1016/S0378-3839(98)00015-5.

Barral, N., Knepley, M.G., Lange, M., Piggott, M.D., Gorman, G.J., 2016. Anisotropic mesh adaptation in Firedrake with PETSc DMPlex. arXiv preprint arXiv:1610.09874 .

Bart, L.J.C., 2017. Long-term modelling with XBeach : combining stationary and surfbeat mode in an integrated approach. Masters thesis. TU Delft, Delft, The Netherlands.

Barthélemy, E., 2004. Nonlinear shallow water theories for coastal waves. Surveys in Geophysics 25, 315–337.

Bastian, P., Müller, E.H., Müthing, S., Piatkowski, M., 2019. Matrix-free multigrid block-preconditioners for higher order discontinuous galerkin discretisations. Journal of Computational Physics 394, 417–439.

Bates, P.D., Horritt, M.S., Fewtrell, T.J., 2010. A simple inertial formulation of the shallow water equations for efficient two-dimensional flood inundation modelling. Journal of Hydrology 387, 33–45.

de Beer, A., McCall, R., Long, J., Tissier, M., Reniers, A., 2020. Simulating wave runup on an intermediate–reflective beach using a wave-resolving and a wave-averaged version of xbeach. Coastal Engineering 163, 103788.

Behrens, C.N., Lopes, H.F., Gamerman, D., 2004. Bayesian analysis of extreme events with threshold estimation. Statistical Modelling 4, 227–244.

Benkhaldoun, F., El Mahi, I., Sari, S., Seaid, M., 2013. An unstructured finite volume method for coupled models of suspended sediment and bed-load transport in shallow water flows. International journal for numerical methods in fluids 72, 967–993. doi:10.1002/d.3771.

Bilanceri, M., Beux, F., Elmahi, I., Guillard, H., Salvetti, M.V., 2011. Comparison of explicit and implicit time advancing in the simulation of a 2d sediment transport problem, in: Finite Volumes for Complex Applications VI Problems & Perspectives. Springer, pp. 125–133.

Budd, C.J., Cullen, M.J.P., Walsh, E.J., 2013. Monge–Ampére based moving mesh methods for numerical weather prediction, with applications to the eady problem. Journal of Computational Physics 236, 247–270.

Budd, C.J., Huang, W., Russell, R.D., 2009. Adaptivity with moving grids. Acta Numerica , 1–131doi:10.1017/S0962492906400015.

Budd, C.J., Williams, J., 2009a. Moving mesh generation using the parabolic Monge–Ampère equation. SIAM Journal on Scientific Computing 31, 3438–3465.

Budd, C.J., Williams, J., 2009b. Moving mesh generation using the parabolic monge–ampère equation. SIAM Journal on Scientific Computing 31, 3438–3465.

Callaghan, D.P., Ranasinghe, R., Roelvink, D., 2013. Probabilistic estimation of storm erosion using analytical, semi-empirical, and process based storm erosion models. Coastal Engineering 82, 64–75.

Carrier, G.F., Greenspan, H.P., 1958. Water waves of finite amplitude on a sloping beach. Journal of Fluid Mechanics 4, 97–109.

Chen, H., Cao, A., Zhang, J., Miao, C., Lv, X., 2014. Estimation of spatially varying open boundary conditions for anumerical internal tidal model with adjoint method. Mathematics and Computers in Simulation 97, 14–38.

Clare, M., 2018. Modelling sediment transport in and around offshore wind farms. Master's thesis. Imperial College London. London, UK.

Clare, M.C.A., Percival, J.R., Angeloudis, A., Cotter, C.J., Piggott, M.D., 2021. Hydro-morphodynamics 2D modelling using a discontinuous Galerkin discretisation. Computers & Geosciences 146.

Clark, S.R., Wei, W., Cai, X., 2010. Numerical analysis of a dual-sediment transport model applied to lake okeechobee, florida, in: 2010 Ninth International Symposium on Parallel and Distributed Computing, IEEE. pp. 189–194.

Cliffe, K.A., Giles, M.B., Scheichl, R., Teckentrup, A.L., 2011. Multilevel Monte Carlo methods and applications to elliptic PDEs with random coefficients. Computing and Visualization in Science 14, 3–15.

Cohn, N., Ruggiero, P., 2016. The influence of seasonal to interannual nearshore profile variability on extreme water levels: Modeling wave runup on dissipative beaches. Coastal Engineering 115, 79–92.

Comblen, R., Lambrechts, J., Remacle, J.F., Legat, V., 2010. Practical evaluation of five partly discontinuous finite element pairs for the non-conservative shallow water equations. International Journal for Numerical Methods in Fluids 63, 701–724.

Cotter, C.J., Ham, D.A., Pain, C.C., Reich, S., 2009. Lbb stability of a mixed galerkin finite element pair for fluid flow simulations. Journal of Computational Physics 228, 336–348.

Dalledonne, G., Kopmann, R., Riehme, J., Naumann, U., 2017. Uncertainty analysis approximation for non-linear processes using Telemac-AD, in: Proceedings of the XXIVth TELEMAC-MASCARET User Conference, 17 to 20 October 2017, Graz University of Technology, Austria, Graz University of Technology, Graz. pp. 65–71. URL: https://hdl.handle.net/20.500.11970/104486.

Danilov, S., 2013. Ocean modeling on unstructured meshes. Ocean Modelling 69, 195–210.

Dastgheib, A., Roelvink, J., Wang, Z., 2008. Long-term process-based morphological modeling of the marsdiep tidal basin. Marine Geology 256, 90–100.

Dawson, C., Sun, S., Wheeler, M.F., 2004. Compatible algorithms for coupled flow and transport. Computer Methods in Applied Mechanics and Engineering 193, 2565–2580.

Delandmeter, P., 2017. Discontinuous Galerkin finite element modelling of geophysical and environmental flows. Ph.D. thesis. Universite catholique de Louvain.

Deltares, 2014. Delft3D-FLOW Simulation of multi-dimensional hydrodynamic flows and transport phenomena including sediments. User Manual. Delft, The Netherlands. URL: https://oss.deltares.nl/documents/183920/185723/Delft3D-FLOW_User_Manual.pdf.

DHI, 2017. MIKE Hydro River User Guide. Hørsholm, Denmark.

Díaz-Sánchez, R., López-Gutiérrez, J., Lechuga, A., Negro, V., 2014. Runup variability due to time dependence and stochasticity in the beach profiles: two extreme cases of the spanish coast. Journal of Coastal Research , 1–6.

Dissanayake, P., Brown, J., Karunarathna, H., 2014. Modelling storm-induced beach/dune evolution: Sefton coast, Liverpool Bay, UK. Marine Geology 357, 225–242. doi:10.1016/J.MARGEO.2014.07.013.

Donea, J., Huerta, A., Ponthot, J.P., Rodríguez-Ferran, A., 2017. Arbitrary Lagrangian–Eulerian methods. Encyclopedia of Computational Mechanics Second Edition , 1–23.

Dourado, F., Costa, P., La Selle, S., Andrade, C., Silva, A., Bosnic, I., Gelfenbaum, G., 2021. Can modeling the geologic record contribute to constraining the tectonic source of the 1755 ce great lisbon earthquake? Earth and Space Science 8, e2020EA001109.

Elfverson, D., Hellman, F., Målqvist, A., 2016. A Multilevel Monte Carlo Method for Computing Failure Probabilities. SIAM/ASA Journal on Uncertainty Quantification 4, 312–330.

Engelund, F., 1974. Flow and bed topography in channel beds. Journal of the Hydraulics Division 100.

Engl, H.W., Hanke, M., Neubauer, A., 1996. Regularization of inverse problems. volume 375. Springer Science & Business Media.

Environment Agency, 2020. National Flood and Coastal Erosion Risk Management Strategy for England. Technical Report. Environment Agency. Bristol, UK.

Epshteyn, Y., Rivière, B., 2007. Estimation of penalty parameters for symmetric interior penalty Galerkin methods. Journal of Computational and Applied Mathematics 206, 843–872.

Farhat, C., Degand, C., Koobus, B., Lesoinne, M., 1998. Torsional springs for two-dimensional dynamic unstructured fluid meshes. Computer methods in applied mechanics and engineering 163, 231–245.

Farrell, P.E., Ham, D.A., Funke, S.W., Rognes, M.E., 2013. Automated derivation of the adjoint of high-level transient finite element programs. SIAM Journal on Scientific Computing 35, 369–393.

Firedrake Project, 2021. Fix split pull request #1609. URL: https://github.com/firedrakeproject/firedrake/pull/1609/files.

Fleeter, C.M., Geraci, G., Schiavazzi, D.E., Kahn, A.M., Marsden, A.L., 2020. Multilevel and multifidelity uncertainty quantification for cardiovascular hemodynamics. Computer methods in applied mechanics and engineering 365, 113030.

Funke, S., Farrell, P., Piggott, M., 2017. Reconstructing wave profiles from inundation data. Computer Methods in Applied Mechanics and Engineering 322, 167–186.

Funke, S.W., 2012. The automation of PDE-constrained optimisation and its applications. Ph.D. thesis. Imperial College London. doi:10.13140/2.1.3688.0967.

Garcia, M., Parker, G., 1991. Entrainment of bed sediment into suspension. Journal of Hydraulic Engineering 117, 414–435.

Gaume, E., Gaál, L., Viglione, A., Szolgay, J., Kohnová, S., Blöschl, G., 2010. Bayesian mcmc approach to regional flood frequency analyses involving extraordinary flood events at ungauged sites. Journal of hydrology 394, 101–117.

Geology Cafe, 2015. Chapter 11 - rivers, streams, and water underground. http://geologycafe.com/class/chapter11.html. Accessed on 2020-09-11.

Geraci, G., Eldred, M., Iaccarino, G., 2015. A multifidelity control variate approach for the multilevel monte carlo technique. Center for Turbulence Research Annual Research Briefs , 169–181.

Geraci, G., Eldred, M.S., Iaccarino, G., 2017. A multifidelity multilevel monte carlo method for uncertainty propagation in aerospace applications, in: 19th AIAA Non-Deterministic Approaches Conference, p. 1951.

Gerritsen, H., de Goede, E., Platzek, F., van Kester, J., Genseberger, M., Uittenbogaard, R., 2008. Validation Document Delft3D-FLOW; a software system for 3D flow simulations. Technical Report. Deltares. Delft, The Netherlands.

Giles, M.B., 2008. Multilevel Monte Carlo Path Simulation. Operations Research 56, 607–617. doi:10.1287/opre.1070.0496.

Giles, M.B., 2015. Multilevel Monte Carlo methods. Acta Numerica 24, 259–328.

Giles, M.B., Nagapetyan, T., Ritter, K., 2015. Multilevel monte carlo approximation of distribution functions and densities. SIAM/ASA journal on Uncertainty Quantification 3, 267–295. doi:10.1137/140960086.

Goto, K., Takahashi, J., Oie, T., Imamura, F., 2011. Remarkable bathymetric change in the nearshore zone by the 2004 Indian Ocean tsunami: Kirinda Harbor, Sri Lanka. Geomorphology 127, 107–116.

Gregory, A., Cotter, C.J., 2017. On the calibration of multilevel Monte Carlo ensemble forecasts. Quarterly Journal of the Royal Meteorological Society 143, 1929–1935. doi:10.1002/qj.3052.

Harris, D.L., Rovere, A., Casella, E., Power, H., Canavesio, R., Collin, A., Pomeroy, A., Webster, J.M., Parravicini, V., 2018. Coral reef structural complexity provides important coastal protection from waves under rising sea levels. Science Advances 4.

Hasselmann, K., Barnett, T., Bouws, E., Carlson, H., Cartwright, D., Enke, K., Ewing, J., Gienapp, H., Hasselmann, D., Kruseman, P., et al., 1973. Measurements of wind-wave growth and swell decay during the Joint North Sea Wave Project (JONSWAP). Ergänzungsheft 8-12 .

Heemink, A.W., Mouthaan, E.E.A., Roest, M.R.T., Vollebregt, E.A.H., Robaczewska, K.B., Verlaan, M., 2002. Inverse 3D shallow water flow modellingof the continental shelf. Continental Shelf Research 22, 465–484.

Hervouet, J., 2007. Hydrodynamics of Free Surface Flows, Modelling with the Finite-element Method. John Wiley & Sons Ltd, West Sussex, UK.

Hervouet, J.M., 1999. TELEMAC, a hydroinformatic system. Houille Blanche-revue Internationale De L Eau 54, 21–28. doi:10.1051/lhb/1999029.

Hervouet, J.M., Pavan, S., Ricchiuto, M., 2017. Residual distribution advection schemes in Telemac. Technical Report. INRIA. Bordeaux, France. URL: https://hal.inria.fr/hal-01571827v2.

Hieu, M.T., Nowak, W., Kopmann, R., 2015. Using algorithmic differentiation for uncertainty analysis, in: Moulinec, C., Emerson, D. (Eds.), XXII TELEMAC-MASCARET Technical User Conference October 15-16, Warrington. pp. 52–58. URL: https://hdl.handle.net/20.500.11970/104343.

Holman, R., 1986. Extreme value statistics for wave run-up on a natural beach. Coastal Engineering 9, 527–544.

Huang, W., Ren, Y., Russell, R.D., 1994. Moving mesh partial differential equations (MMPDEs) based on the equidistribution principle. SIAM Journal on Numerical Analysis 31, 709–730.

Huang, W., Russell, R.D., 1998. Moving mesh strategy based on a gradient flow equation for two-dimensional problems. SIAM Journal on Scientific Computing 20, 998–1015.

Hudson, J., Sweby, P.K., 2005. A high-resolution scheme for the equations governing 2D bed-load sediment transport. International Journal for Numerical Methods in Fluids 47, 1085–1091.

Hunt Jr, I.A., 1959. Design of seawalls and breakwaters. Journal of the Waterways and Harbors Division 85, 123–152.

Hunter, N.M., Horritt, M.S., Bates, P.D., Wilson, M.D., Werner, M.G., 2005. An adaptive time step solution for raster-based storage cell modelling of floodplain inundation. Advances in water resources 28, 975–991.

Huybrechts, N., Villaret, C., Hervouet, J.M., 2010. Comparison between 2D and 3D modelling of sediment transport: Application to the dune evolution, in: Proceedings of the 5th International Conference on fluvial Hydraulics, Braunschweig, Germany. pp. 887–893.

Jacobs, C.T., Piggott, M.D., 2015. Firedrake-Fluids v0.1: Numerical modelling of shallow water flows using an automated solution framework. Geoscientific Model Development 8, 533–547. doi:10.5194/gmd-8-533-2015.

Jaffe, B.E., Goto, K., Sugawara, D., Gelfenbaum, G., La Selle, S., 2016. Uncertainty in tsunami sediment transport modeling. Journal of Disaster Research 11, 647–661. doi:10.20965/jdr.2016.p0647.

Jansen, K., Müller, E.H., Scheichl, R., 2020. Multilevel monte carlo algorithm for quantum mechanics on a lattice. Physical Review D 102, 114512.

Kärnä, T., de Brye, B., Gourgue, O., Lambrechts, J., Comblen, R., Legat, V., Deleersnijder, E., 2011. A fully implicit wetting–drying method for DG-FEM shallow water models, with an application to the Scheldt Estuary. Computer Methods in Applied Mechanics and Engineering 200, 509–524.

Kärnä, T., Kramer, S., Mitchell, L., Ham, D., Piggott, M., Baptista, A., 2018. Thetis coastal ocean model: discontinuous Galerkin discretization for the three-dimensional hydrostatic equations. Geoscientific Model Development 11, 4359–4382.

Katsiolides, G., Müller, E.H., Scheichl, R., Shardlow, T., Giles, M.B., Thomson, D.J., 2018. Multilevel Monte Carlo and improved timestepping methods in atmospheric dispersion modelling. Journal of Computational Physics 354, 320–343.

Kazhyken, K., Videman, J., Dawson, C., 2021. Discontinuous Galerkin methods for a dispersive wave hydro-sediment-morphodynamic model. Computer Methods in Applied Mechanics and Engineering 377, 113684. URL: https://doi.org/10.1016/j.cma.2021.113684, doi:10.1016/j.cma.2021.113684, arXiv:2010.06167.

Kobayashi, N., Lawrence, A.R., 2004. Cross-shore sediment transport under breaking solitary waves. Journal of Geophysical Research: Oceans 109. doi:10.1029/2003jc002084.

Kopmann, R., Merkel, U., Riehme, J., 2012. Using Reliability Analysis in Morphodynamic Simulation with TELEMAC-2D / SISYPHE, in: XIXth TELEMAC-MASCARET User Conference, Oxford, UK, October 18-19, 2012., HR Wallingford, Wallingford, Oxfordshire, UK. pp. 119–125.

Kubatko, E.J., Westerink, J.J., Dawson, C., 2006. An unstructured grid morphodynamic model with a discontinuous Galerkin method for bed evolution. Ocean modelling 15, 71–89.

Kulkarni, V., Sahoo, N., 2013. Module 5: Viscous Incompressible flow; Lecture 5 : Internal Flow – Part IV. https://nptel.ac.in/courses/101103004/module5/lec5/4.html.

Lakkis, O., Pryer, T., 2013. A finite element method for nonlinear elliptic problems. SIAM Journal on Scientific Computing 35, A2025–A2045.

Landsberg, A., Chtchelkanova, A., Lind, C., Boris, J., Young, T., 1998. Fast3D user and programmer reference manual. US Naval Research Laboratory. Washington DC, USA.

Larson, M., Kraus, N.C., 1989. SBEACH: numerical model for simulating storm-induced beach change. Report 1. Empirical foundation and model development. Technical Report. Coastal Engineering research center Vicksburg Ms.

Leijnse, T., 2018. Computationally Efficient Modelling of Compound Flooding due to Tropical Cyclones with the Explicit Inclusion of Wave-Driven Processes. Ph.D. thesis. TU Delft, Delft, The Netherlands. URL: http://repository.tudelft.nl/.

Leijnse, T., van Ormondt, M., Nederhoff, K., van Dongeren, A., 2021. Modeling compound flooding in coastal systems using a computationally efficient reduced-physics solver: Including fluvial, pluvial, tidal, wind- and wave-driven processes. Coastal Engineering 163. URL: https://doi.org/10.1016/j.coastaleng.2020.103796, doi:10.1016/j.coastaleng.2020.103796.

Lesser, G.R., Roelvink, J.v., van Kester, J.T.M., Stelling, G., 2004. Development and validation of a three-dimensional morphological model. Coastal engineering 51, 883–915.

Leveque, R.J., 1996. High-resolution conservative algorithms for advection in incompressible. SIAM Journal on Numerical Analysis 33, 627–665.

Li, L., Huang, Z., 2013. Modeling the change of beach profile under tsunami waves: A comparison of selected sediment transport models. Journal of Earthquake and Tsunami 7. doi:10.1142/S1793431113500012.

Maddison, J.R., Panourgias, I., Farrell, P.E., 2017. libsupermesh version 1.0. Technical Report. University of Edinburgh.

Mayne, D.A., Usmani, A.S., Crapper, M., 2002. An Adaptive Finite Element Solution for Cohesive Sediment Transport, in: Proceedings in Marine Science. Elsevier. volume 5, pp. 627–641. doi:10.1016/S1568-2692(02)80044-4.

McCall, R.T., De Vries, J.V.T., Plant, N., Van Dongeren, A., Roelvink, J., Thompson, D., Reniers, A., 2010. Two-dimensional time dependent hurricane overwash and erosion modeling at santa rosa island. Coastal Engineering 57, 668–683.

McManus, T.M., Percival, J.R., Yeager, B.A., Barral, N., Gorman, G.J., Piggott, M.D., 2017. Moving mesh methods in Fluidity and Firedrake. Technical Report July. Imperial College London. doi:10.13140/RG.2.2.27670.24648.

McRae, A.T.T., Cotter, C.J., Budd, C.J., 2018. Optimal-transport-based mesh adaptivity on the plane and sphere using finite elements. SIAM Journal on Scientific Computing 40, 1121–1148. doi:10.1137/16M1109515.

Melby, J., Caraballo-Nadal, N., Kobayashi, N., 2012a. Wave runup prediction for flood mapping. Coastal Engineering Proceedings 1, management.79.

Melby, J.A., et al., 2012b. Wave runup prediction for flood hazard assessment. Technical Report. US Army Corps of Engineers.

Merkel, U., Riehme, J., Naumann, U., 2013. Reverse Engineering of Initial & Boundary Conditions with TELEMAC and Algorithmic Differentiation, in: Kopmann, R., Goll, A. (Eds.), XXth TELEMAC-MASCARET. User Conference 2013., Karlsruhe. pp. 25–30. URL: https://hdl.handle.net/20.500.11970/100423.

Merkel, U., Riehme, J., Naumann, U., 2016. River flow analysis with adjoints - An efficient, universal methodology to quantify spatial interactions and sensitivities, in: Erpicum, S., Dewals, B., Archambeau, P., Pirotton, M. (Eds.), Sustainable Hydraulics in the Era of Global Change. Taylor & Francis

Group, London, UK, pp. 538–544.

Morgan, J.A., Kumar, N., Horner-Devine, A.R., Ahrendt, S., Istanbullouglu, E., Bandaragoda, C., 2020. The use of a morphological acceleration factor in the simulation of large-scale fluvial morphodynamics. Geomorphology 356, 107088.

Naumann, U., Riehme, J., 2005. A differentiation-enabled fortran 95 compiler. ACM Transactions on Mathematical Software (TOMS) 31, 458–474.

Naumann, U., Riehme, J., 2008. Sensitivity Analysis in Sisyphe with the AD-Enabled NAGWare Fortran Compiler. Technical Report. Department of Computer Science, RWTH Aachen University. Aachen, Germany. URL: https://www.researchgate.net/publication/267305598.

Nicholas, A., Ashworth, P., Smith, G.S., Sandbach, S., 2013. Numerical simulation of bar and island morphodynamics in anabranching megarivers. Journal of Geophysical Research: Earth Surface 118, 2019–2044.

Pan, W., Kramer, S.C., Piggott, M.D., 2019. Multi-layer non-hydrostatic free surface modelling using the discontinuous Galerkin method. Ocean Modelling 134, 68–83.

Papanicolaou, A.T.N., Elhakeem, M., Krallis, G., Prakash, S., Edinger, J., 2008. Sediment transport modeling review - current and future developments. Journal of Hydraulic Engineering 134, 1–14.

Park, H., Cox, D.T., 2016. Empirical wave run-up formula for wave, storm surge and berm width. Coastal Engineering 115, 67–78.

Park, S.W., Ahn, J., 2019. Experimental and numerical investigations of primary flow patterns and mixing in laboratory meandering channel. Smart Water 4, 4. URL: https://doi.org/10.1186/s40713-019-0016-y, doi:10.1186/s40713-019-0016-y.

Parkinson, S.D., Funke, S.W., Hill, J., Piggott, M.D., Allison, P.A., 2017. Application of the adjoint approach to optimise the initial conditions of a turbidity current with the AdjointTurbidity 1.0 model. Geoscientific Model Development 10, 1051–1068.

Peherstorfer, B., Willcox, K., Gunzburger, M., 2018. Survey of multifidelity methods in uncertainty propagation, inference, and optimization. Siam Review 60, 550–591.

Pender, D., Karunarathna, H., 2013. A statistical-process based approach for modelling beach profile variability. Coastal Engineering 81, 19–29.

Piggott, M.D., Farrell, P.E., Wilson, C.R., Gorman, G.J., Pain, C.C., 2009. Anisotropic mesh adaptivity for multi-scale ocean modelling. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences 367, 4591–4611. doi:10.1098/rsta.2009.0155, arXiv:https://royalsocietypublishing.org/doi/pdf/10.1098/rsta.2009.0155.

Piggott, M.D., Gorman, G.J., Pain, C.C., Allison, P.A., Candy, A.S., Martin, B.T., Wells, M.R., 2008a. A new computational framework for multi-scale ocean modelling based on adapting unstructured meshes. International Journal for Numerical Methods in Fluids 56, 1003–1015.

Piggott, M.D., Pain, C.C., Gorman, G.J., Marshall, D.P., Killworth, P.D., 2008b. Unstructured Adaptive Meshes for Ocean Modeling, in: Hecht, M.W., Hasumi, H. (Eds.), Ocean Modeling in an Eddying Regime. American Geophysical Union, pp. 383–408.

Piggott, M.D., Pain, C.C., Gorman, G.J., Power, P.W., Goddard, A.J.H., 2005. h, r, and hr adaptivity with applications in numerical ocean modelling. Ocean Modelling 10, 95–113. doi:10.1016/j.ocemod.2004.07.007.

Pörtner, H.O., Roberts, D.C., Masson-Delmotte, V., Zhai, P., 2019. The Ocean and Cryosphere in a Changing Climate A Special Report of the Intergovernmental Panel on Climate Change. Technical Report. IPCC. Geneva, Switzerland.

Ranasinghe, R., Swinkels, C., Luijendijk, A., Roelvink, D., Bosboom, J., Stive, M., Walstra, D., 2011. Morphodynamic upscaling with the morfac approach: Dependencies and sensitivities. Coastal engineering 58, 806–811.

Rathgeber, F., Ham, D.A., Mitchell, L., Lange, M., Luporini, F., McRae, A.T., Bercea, G., Markall, G., Kelly, P.H., 2017. Firedrake: automating the finite element method by composing abstractions. ACM Transactions on Mathematical Software (TOMS) 43, 24.

Renard, B., Lang, M., Bois, P., 2006. Statistical analysis of extreme events in a non-stationary context via a bayesian framework: case study with peak-over-threshold data. Stochastic environmental research and risk assessment 21, 97–112.

Riehme, J., Kopmann, R., Naumann, U., 2010. Uncertainty quantification based on forward sensitivity analysis in Sispyhe, in: Pereira, J.C.F., Sequeira, A. (Eds.), V European Conference on Computational Fluid Dynamics, Lisbon, Portugal.

Riesenkamp, M., 2011. Probabilistic modelling of extreme beach erosion using XBeach. Ph.D. thesis. TU Delft, Delft, The Netherlands. URL: http://repository.tudelft.nl/.

Roelvink, D., McCall, R., Mehvar, S., Nederhoff, K., Dastgheib, A., 2018. Improving predictions of swash dynamics in XBeach: The role of groupiness and incident-band runup. Coastal Engineering 134, 103–123. doi:10.1016/j.coastaleng.2017.07.004.

Roelvink, D., Reniers, A., Van Dongeren, A., De Vries, J.V.T., McCall, R., Lescinski, J., 2009. Modelling storm impacts on beaches, dunes and barrier islands. Coastal engineering 56, 1133–1152.

Roelvink, D., Van Dongeren, A., McCall, R., Hoonhout, B., Van Rooijen, A., Van Geer, P., De Vet, L., Nederhoff, K., Quataert, E., 2015. XBeach Technical Reference: Kingsday Release. Technical Report. Deltares, Delft, The Netherlands.

Ruggiero, P., Komar, P.D., McDougal, W.G., Marra, J.J., Beach, R.A., 2001. Wave runup, extreme water levels and the erosion of properties backing beaches. Journal of coastal research , 407–419.

Saito, T., Ito, Y., Inazu, D., Hino, R., 2011. Tsunami source of the 2011 tohoku-oki earthquake, japan: Inversion analysis based on dispersive tsunami simulations. Geophysical Research Letters 38.

Sánchez-Linares, C., de la Asunción, M., Castro, M.J., González-Vida, J.M., Macías, J., Mishra, S., 2016. Uncertainty quantification in tsunami modeling using multi-level Monte Carlo finite volume method. Journal of Mathematics in Industry 6, 1–26.

Segur, H., 2009. Lecture 8 : The Shallow-Water Equations. Woods Hole Oceanographic Institution, Woods Hole, Massachusetts, USA.

Shu, C.W., 2009. Discontinuous galerkin methods: general approach and stability. Numerical solutions of partial differential equations 201.

Simmons, J.A., Harley, M.D., Marshall, L.A., Turner, I.L., Splinter, K.D., Cox, R.J., 2017. Calibrating and assessing uncertainty in coastal numerical models. Coastal Engineering 125, 28–41.

Singh, U., Crosato, A., Giri, S., Hicks, M., 2017. Sediment heterogeneity and mobility in the morphodynamic modelling of gravel-bed braided rivers. Advances in water resources 104, 127–144.

Soulsby, R., 1997. Dynamics of marine sands, a manual for practical applications. Thomas Telford, London, UK.

Stockdon, H.F., Holman, R.A., Howd, P.A., Sallenger Jr, A.H., 2006. Empirical parameterization of setup, swash, and runup. Coastal engineering 53, 573–588.

Suanez, S., Cancouët, R., Floc'h, F., Blaise, E., Ardhuin, F., Filipot, J.F., Cariolet, J.M., Delacourt, C., 2015. Observations and predictions of wave runup, extreme water levels, and medium-term dune erosion during storm conditions. Journal of Marine Science and Engineering 3, 674–698.

Sverdrup, H.U., 1947. Wind, sea and swell. Theory of relations for forecasting. US Navy Hydrog. Office, Pub. 601, 44.

Syvitski, J., Slingerland, R., Burgess, P., Meiburg, E., Murray, A.B., Wiberg, P., Tucker, G., Voinov, A., 2010. Morphodynamic Models: An Overview. River, Coastal and Estuarine Morphodynamics , 3–20.

Talmon, A., Struiksma, N., Mierlo, M.V., 1995. Laboratory measurements of the direction of sediment transport on transverse alluvial-bed slopes. Journal of Hydraulic Research 33, 495–517.

Tang, H., Wang, J., Weiss, R., Xiao, H., 2018. TSUFLIND-EnKF: Inversion of tsunami flow depth and flow speed from deposits with quantified uncertainties. Marine Geology 396, 16–25.

Tang, T., 2005. Moving mesh methods for computational fluid dynamics. Contemporary mathematics 383, 141–173.

Tassi, P., Rhebergen, S., Vionnet, C., Bokhove, O., 2008. A discontinuous Galerkin finite element model for river bed evolution under shallow flows. Computer Methods in Applied Mechanics and Engineering 197, 2930–2947.

Tassi, P., Villaret, C., 2014. Sisyphe v6.3 User's Manual. EDF R&D. Chatou, France.

Thacker, W.C., 1981. Some exact solutions to the nonlinear shallow-water wave equations. Journal of Fluid Mechanics 107, 499–508.

Torsethaugen, K., Haver, S., 2004. Simplified double peak spectral model for ocean waves, in: The Fourteenth International Offshore and Polar Engineering Conference, OnePetro.

Unguendoli, S., 2018. Propagation of uncertainty across modeling chains to evaluate hydraulic vulnerability in coastal areas. Ph.D. thesis. Università di Bologna, Bologna, Italy. URL: http://amsdottorato.unibo.it/8599/1/Unguendoli{_}Silvia{_}Tesi.pdf.

Van Gent, M.R., Coeveld, E.M., De Vroeg, H., Van De Graaff, J., 2007. Dune erosion prediction methods incorporating effects of wave periods, in: Coastal Sediments '07 - Proceedings of 6th International Symposium on Coastal Engineering and Science of Coastal Sediment Processes, pp. 612–625.

Van Rijn, L.C., 1980. Storm surge barrier Oosterschelde-computation of siltation in dredged trenches: Semi-empirical model for the flow in dredged trenches. Deltares, Delft, The Netherlands.

Van Rijn, L.C., 1984. Sediment Transport, Part II: Suspended Load Transport. Journal of Hydraulic Engineering 110, 1613–1641.

Villaret, C., Hervouet, J.M., Kopmann, R., Merkel, U., Davies, A.G., 2013. Morphodynamic modeling using the Telemac finite-element system. Computers & Geosciences 53, 105–113.

Villaret, C., Kopmann, R., Wyncoll, D., Riehme, J., Merkel, U., Naumann, U., 2016. First-order uncertainty analysis using Algorithmic Differentiation of morphodynamic models. Computers & geosciences 90, 144–151.

Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S.J., Brett, M., Wilson, J., Millman, K.J., Mayorov, N., Nelson, A.R.J., Jones, E., Kern, R., Larson, E., Carey, C.J., Polat, İ., Feng, Y., Moore, E.W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E.A., Harris, C.R., Archibald, A.M., Ribeiro, A.H., Pedregosa, F., van Mulbregt, P., SciPy 1.0 Contributors, 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods 17, 261–272. doi:10.1038/s41592-019-0686-2.

Vouriot, C.V.M., Angeloudis, A., Kramer, S.C., Piggott, M.D., 2019. Fate of large-scale vortices in idealized tidal lagoons. Environmental Fluid Mechanics 19, 329–348.

Vousdoukas, M.I., Mentaschi, L., Voukouvalas, E., Bianchi, A., Dottori, F., Feyen, L., 2018. Climatic and socioeconomic controls of future coastal flood risk in europe. Nature Climate Change 8, 776–780.

Vousdoukas, M.I., Ranasinghe, R., Mentaschi, L., Plomaritis, T.A., Athanasiou, P., Luijendijk, A., Feyen, L., 2020. Sandy coastlines under threat of erosion. Nature Climate Change 10, 260–263. URL: https://www.nature.com/articles/s41558-020-0697-0, doi:10.1038/s41558-020-0697-0.

Wallwork, J., 2021. Mesh Adaptation and Adjoint Methods for Finite Element Coastal Ocean Modelling. Ph.D. thesis. Imperial College London.

Wallwork, J.G., Barral, N., Ham, D.A., Piggott, M.D., 2020a. Anisotropic goal-oriented mesh adaptation in Firedrake, in: 28th Intl Meshing Roundtable, Zenodo. pp. 83—-100. doi:10.5281/zenodo.3653101.

Wallwork, J.G., Barral, N., Kramer, S.C., Ham, D.A., Piggott, M.D., 2020b. Goal-oriented error estimation and mesh adaptation for shallow water modelling. Springer Nature Applied Sciences 2, 1053–1063. doi:10.1007/s42452-020-2745-9.

Warder, S.C., Horsburgh, K.J., Piggott, M.D., 2021. Adjoint-based sensitivity analysis for a numerical storm surge model. Ocean Modelling , 101766.

Warner, J.C., Sherwood, C.R., Signell, R.P., Harris, C.K., Arango, H.G., 2008. Development of a three-dimensional, regional, coupled wave, current, and sediment-transport model. Computers & Geosciences 34, 1284–1306.

Warren, I., Bach, H., 1992. MIKE 21: a modelling system for estuaries, coastal waters and seas. Environmental Software 7, 229–240.

Van der Wegen, M., Roelvink, J., 2008. Long-term morphodynamic evolution of a tidal embayment using a two-dimensional, process-based model. Journal of Geophysical Research: Oceans 113.

Weller, H., Browne, P., Budd, C., Cullen, M., 2016. Mesh adaptation on the sphere using optimal transport and the numerical solution of a Monge–Ampère type equation. Journal of Computational Physics 308, 102–123.

White, Jr, A.B., 1979. On selection of equidistributing meshes for two-point boundary-value problems. SIAM Journal on Numerical Analysis 16, 472–502.

Wu, W., 2007. Computational river dynamics. CRC Press, Boca Raton, Florida, USA.

Yen, C.l., Lee, K.T., 1995. Bed topography and sediment sorting in channel bend with unsteady flow. Journal of Hydraulic Engineering 121, 591–599.

Zhou, F., Chen, G., Huang, Y., Yang, J.Z., Feng, H., 2013. An adaptive moving finite volume scheme for modeling flood inundation over dry and complex topography. Water Resources Research 49, 1914–1928. doi:10.1002/wrcr.20179.

APPENDIX A

Deriving the continuous adjoint equations for hydro-morphodynamic models

In this appendix, we derive the continuous adjoint equations for the fully wet version of the hydro-morphodynamic model, noting that those for the wet-dry version are very similar. Recall, from Section 2.2.3, that the combined system of equations in the hydro-morphodynamic model are solved alternately, and not as a monolithic system. Thus, we derive the continuous adjoint for each set of equations separately.

A.1 Hydrodynamic equations

We start by deriving the continuous adjoint of the hydrodynamic equations (2.55) and (2.56), following Funke (2012). Recall the hydrodynamic equations are

$$\frac{\partial \eta}{\partial t} + \nabla_h \cdot (h\overline{\mathbf{u}}) = 0; \tag{A.1}$$

$$\frac{\partial \overline{\mathbf{u}}}{\partial t} + \overline{\mathbf{u}} \cdot \nabla_h \overline{\mathbf{u}} + \mathbf{g}\nabla_h \eta = \nu \nabla_h^2 \overline{\mathbf{u}} - C||\overline{\mathbf{u}}||\overline{\mathbf{u}}, \tag{A.2}$$

where C represents the friction, and all other variables are as defined in Section 2.2. For simplicity, we drop the overline for the depth-averaged variables. The equation system (A.1–A.2) is evaluated on the domain $\Omega \times (0,T)$ with generic initial conditions $\mathbf{u}|_{\Omega\times 0} = \mathbf{u}_0$ and $\eta|_{\Omega\times 0} = \eta_0$, and a generic Dirichlet boundary condition, $\mathbf{u}|_{\partial\Omega\times(0,T)} = g_{\mathbf{u}}$. Note that in order to ensure this problem is well-posed, a Dirichlet boundary condition cannot be applied on both $\mathbf{u}$ and η at the same place. Therefore, we only apply a boundary condition on $\mathbf{u}$, but the derivation would be similar if we had chosen to apply the boundary condition on η instead.

This equation system can be written in the abstract form $F(\mathbf{u}, \eta) = 0$, where $\mathbf{u}$ satisfies the boundary condition $\mathbf{u}|_{\partial\Omega\times(0,T)} = g_{\mathbf{u}}$, *i.e.* $\mathbf{u}$ is a member of the function space $(W \times W)_{g_{\mathbf{u}}} = \{\mathbf{w} \in H^2(\Omega) \times H^2(\Omega)\ s.t.\ \mathbf{w}|_{\partial\Omega\times(0,T)} = g_{\mathbf{u}}\}$. Note that, equivalently, throughout this appendix the function space W is defined as

$$W = \{w \in H^2(\Omega)\ s.t.\ w|_{\partial\Omega\times(0,T)} = 0\}, \tag{A.3}$$

and $W \times W = \{\mathbf{w} \in H^2(\Omega) \times H^2(\Omega)\ s.t.\ \mathbf{w}|_{\partial\Omega\times(0,T)} = 0\}$. The weak form of the hydrodynamic system is then

(A.4)
$$\int_{\Omega\times(0,T)} \lambda_\eta \left(\frac{\partial \eta}{\partial t} + \nabla_h \cdot (h\mathbf{u})\right) dx\,dt + \int_{\Omega\times(0,T)} \lambda_\mathbf{u}^T \left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla_h\mathbf{u} + \mathbf{g}\nabla_h\eta - \nu\nabla_h^2\mathbf{u} + C||\mathbf{u}||\mathbf{u}\right) dx\,dt$$
$$+ \int_{\Omega\times\{0\}} \hat{\alpha}_1(\eta - \eta_0)\,dx + \int_{\Omega\times\{0\}} \hat{\alpha}_2(\mathbf{u} - \mathbf{u}_0)\,dx = 0 \qquad \forall\lambda_\eta \in H^1(\Omega) \text{ and } \forall\lambda_\mathbf{u} \in W \times W,$$

where $\hat{\alpha}_i$ are parameters to be determined, and λ_η and $\lambda_\mathbf{u}$ are the adjoint solutions of elevation and velocity, respectively.

As discussed in Chapter 3, the adjoint equation is defined as

(A.5)
$$\left\langle \lambda, \frac{\partial F}{\partial \sigma}\mathbf{v} \right\rangle = \left\langle \frac{\partial J}{\partial \sigma}, \mathbf{v} \right\rangle,$$

where $\sigma = (\mathbf{u}, \eta)$ represents the forward solution fields, $\lambda = (\lambda_\mathbf{u}, \lambda_\eta)$ the adjoint solution fields, and J is the chosen output functional. Thus, for the hydrodynamic equations, the weak form of the adjoint equation is

(A.6)
$$\int_{\Omega\times(0,T)} \lambda_\eta \left(\frac{\partial v_\eta}{\partial t} + \nabla_h\cdot(hv_\mathbf{u}) + \nabla_h\cdot(v_\eta\mathbf{u})\right) dx\,dt$$
$$+ \int_{\Omega\times(0,T)} \lambda_\mathbf{u}^T \left(\frac{\partial v_\mathbf{u}}{\partial t} + v_\mathbf{u}\cdot\nabla_h\mathbf{u} + \mathbf{u}\cdot\nabla_h v_\mathbf{u} + \mathbf{g}\nabla_h v_\eta - \nu\nabla_h^2 v_\mathbf{u} + C\left(||\mathbf{u}||v_\mathbf{u} + \frac{\mathbf{u}}{||\mathbf{u}||}\mathbf{u}\cdot v_\mathbf{u}\right)\right) dx\,dt$$
$$+ \int_{\Omega\times\{0\}} \hat{\alpha}_1 v_\eta\,dx + \int_{\Omega\times\{0\}} \hat{\alpha}_2 v_\mathbf{u}\,dx = \int_{\Omega\times(0,T)} \frac{\partial J^T}{\partial \mathbf{u}} v_\mathbf{u} + \frac{\partial J^T}{\partial \eta} v_\eta\,dx\,dt$$
$$\forall v_\eta \in H^1(\Omega) \text{ and } \forall v_\mathbf{u} \in W \times W,$$

where v_η and $v_\mathbf{u}$ are test functions for elevation and velocity, respectively.

We then integrate each term by parts so that the derivatives are applied to the adjoint solution λ rather than to v_η and $v_\mathbf{u}$. Thus

(A.7)
$$\int_{\Omega\times(0,T)} v_\eta \left(-\frac{\partial \lambda_\eta}{\partial t} - \mathbf{g}\nabla_h\cdot\lambda_\mathbf{u} - \mathbf{u}\cdot\nabla_h\lambda_\eta\right) +$$
$$\int_{\Omega\times(0,T)} v_\mathbf{u}^T \left(-\frac{\partial \lambda_\mathbf{u}}{\partial t} + (\nabla_h\mathbf{u})\lambda_\mathbf{u} - (\nabla_h\cdot\mathbf{u})\lambda_\mathbf{u} - \mathbf{u}\cdot\nabla_h\lambda_\mathbf{u} - \nu\nabla_h^2\lambda_\mathbf{u} + C\left(||\mathbf{u}||\lambda_\mathbf{u} + \frac{\mathbf{u}\cdot\lambda_\mathbf{u}}{||\mathbf{u}||\mathbf{u}}\right) - h\nabla_h\lambda_\eta\right) dx\,dt$$
$$+ \int_{\partial\Omega\times(0,T)} v_\eta\,(\lambda_\mathbf{u}\cdot\mathbf{n} + \lambda_\eta\mathbf{u}\cdot\mathbf{n}) + \int_{\partial\Omega\times(0,T)} \left[\left(2(\mathbf{u}\cdot\mathbf{n})v_\mathbf{u}^T\lambda_\mathbf{u} + \nu v_\mathbf{u}^T\frac{\partial\lambda_\mathbf{u}}{\partial\mathbf{n}}\right) - \nu\frac{\partial v_\mathbf{u}}{\partial\mathbf{n}}^T\lambda_\mathbf{u} + h\lambda_\eta v_\mathbf{u}\cdot\mathbf{n}\right] ds\,dt$$
$$+ \int_{\Omega\times\{T\}} v_\eta\lambda_\eta + v_\mathbf{u}^T\lambda_\mathbf{u}\,dx + \int_{\Omega\times\{0\}} v_\eta(\hat{\alpha}_1 - \lambda_\eta) + v_\mathbf{u}^T(\hat{\alpha}_2 - \lambda_\mathbf{u})\,dx = \int_{\Omega\times(0,T)} \frac{\partial J^T}{\partial\mathbf{u}} v_\mathbf{u} + \frac{\partial J}{\partial\eta} v_\eta\,dx\,dt$$
$$\forall v_\eta \in H^1(\Omega) \text{ and } \forall v_\mathbf{u} \in W \times W.$$

By definition of the function space W (A.3), $v_\mathbf{u} = \lambda_\mathbf{u} = (0,0)$ at $\partial\Omega \times (0,T)$, which means that all the integrals on this boundary in (A.7) are equal to zero. Therefore, using the Fundamental Lemma of the

Calculus of Variations, the strong form of the adjoint version of the hydrodynamic equations is

$$-\frac{\partial \lambda_{\mathbf{u}}}{\partial t} + (\nabla_h \mathbf{u})\lambda_{\mathbf{u}} - \nabla_h \cdot (\lambda_{\mathbf{u}} \mathbf{u}) - \nu \nabla_h^2 \lambda_{\mathbf{u}} + C\left(||\mathbf{u}||\lambda_{\mathbf{u}} + \frac{\mathbf{u}\cdot\lambda_{\mathbf{u}}}{||\mathbf{u}||\mathbf{u}}\right) - h\nabla_h \lambda_\eta = \frac{\partial J^T}{\partial \mathbf{u}}, \tag{A.8}$$

$$-\frac{\partial \lambda_\eta}{\partial t} - \mathbf{g}\nabla_h \cdot \lambda_{\mathbf{u}} - \mathbf{u}\cdot\nabla_h \lambda_\eta = \frac{\partial J^T}{\partial \eta}, \tag{A.9}$$

in $\Omega \times (0,T)$ with boundary conditions

$$\lambda_{\mathbf{u}} = 0 \quad \text{at } \partial\Omega \times (0,T), \tag{A.10a}$$

$$\lambda_{\mathbf{u}} = 0 \quad \text{at } \Omega \times \{T\}, \tag{A.10b}$$

$$\lambda_\eta = 0 \quad \text{at } \Omega \times \{T\}, \tag{A.10c}$$

and $\hat{\alpha}_1 = \lambda_\eta$ and $\hat{\alpha}_2 = \lambda_{\mathbf{u}}$ at $\Omega \times \{0\}$. Note that (A.10a) comes from the choice of function space W and acts as a no-slip condition. For simplicity, in order for this problem to be well-posed, we have assumed that the system is closed as otherwise there would be a Dirichlet boundary condition on an outflow boundary. For an open system like that in the test cases considered in this thesis, the problem becomes much more complex, and this illustrates the advantages we gain from using a programme like *pyadjoint* to automatically derive the adjoint equations.

In summary, the adjoint equations (A.8) and (A.9) are similar to the forward hydrodynamic equations, but with reversed time derivatives and some reversed flow velocities. Moreover, information propagates backwards in time with adjoint equations and, thus, instead of an initial condition at $t = 0$, a final condition at $t = T$ is specified.

A.2 Non-conservative sediment concentration equation

We next derive the continuous adjoint of the non-conservative sediment concentration equation (2.57) using the same methodology as in the previous section. Recall this equation is

$$\frac{\partial \bar{c}}{\partial t} + F_{\text{corr}}\bar{\mathbf{u}}\cdot\nabla_h \bar{c} = \epsilon_s \nabla_h^2 \bar{c} + \frac{E_b - w_s \alpha_c \bar{c}}{h}, \tag{A.11}$$

where all variables are as defined in Section 2.2. This equation is evaluated on the domain $\Omega \times (0,T)$ with generic initial condition $c|_{\Omega\times 0} = c_0$ and Dirichlet boundary condition $c|_{\partial\Omega\times(0,T)} = g_c$. Here we have written the deposition term D_b as $w_s\hat{\alpha}_c\bar{c}$ to better illustrate that the deposition term is dependent on $\bar{c}$. For simplicity, hereafter we drop the overline for the depth-averaged variables.

This sediment concentration equation system can be written in the abstract form $F(c) = 0$, where c satisfies the boundary condition $c|_{\partial\Omega\times(0,T)} = g_c$, *i.e.* c is a member of the function space $W_{g_c} = \{w \in H^2(\Omega)\, s.t.\, w|_{\partial\Omega\times(0,T)} = g_c\}$. The weak form of the system is then

$$\int_{\Omega\times(0,T)} \lambda_c \left(\frac{\partial c}{\partial t} + F_{\text{corr}}\mathbf{u}\cdot\nabla_h c - \frac{E_b - w_s\hat{\alpha}_c c}{h}\right) + \epsilon_s \nabla_h \lambda_c \cdot \nabla_h c \; dx\, dt + \int_{\Omega\times\{0\}} \hat{\alpha}_3 (c - c_0)\, dx = 0 \qquad \forall \lambda_c \in W, \tag{A.12}$$

where we have integrated the diffusion term by parts. Here, $\hat{\alpha}_3$ is a parameter to be determined and λ_c is the adjoint solution. Therefore, using (A.5) as before, the weak form of the adjoint equation is

$$\int_{\Omega\times(0,T)} \lambda_c \left(\frac{\partial v_c}{\partial t} + F_{\text{corr}}\mathbf{u}\cdot\nabla_h v_c + D v_c \right) + \epsilon_s \nabla_h \lambda_c \cdot \nabla_h v_c \, dx\, dt + \int_{\Omega\times\{0\}} \hat{\alpha}_3 v_c \, dx = \int_{\Omega\times(0,T)} \frac{\partial J^T}{\partial c} v_c \, dx\, dt, \quad \forall v_c \in W, \tag{A.13}$$

where v_c is a test function for the sediment concentration and, for simplicity, we use D to denote $w_s\alpha_c/h$.

We then integrate by parts so that the derivatives are applied to the adjoint solution λ_c. Thus

$$\begin{aligned}\int_{\Omega\times(0,T)} v_c \left(-\frac{\partial \lambda_c}{\partial t} - F_{\text{corr}}\mathbf{u}\cdot\nabla_h\lambda_c - (\nabla_h\cdot F_{\text{corr}}\mathbf{u})\lambda_c - \epsilon_s\nabla_h^2\lambda_c + D\lambda_c \right) dx\, dt \\ + \int_{\partial\Omega\times(0,T)} v_c \left(\lambda_c F_{\text{corr}}\mathbf{u}\cdot\mathbf{n} + \epsilon_s \frac{\partial \lambda_c}{\partial \mathbf{n}} \right) ds\, dt + \int_{\Omega\times\{T\}} \lambda_c v_c \, dx + \int_{\Omega\times\{0\}} (\hat{\alpha}_3 - \lambda_c)\, v_c \, dx \\ - \int_{\partial\Omega\times(0,T)} \epsilon_s \lambda_c \frac{\partial v_c}{\partial n} \, ds\, dt = \int_{\Omega\times(0,T)} \frac{\partial J^T}{\partial c} v_c \, dx\, dt, \quad \forall v_c \in W.\end{aligned} \tag{A.14}$$

By definition of W, $v_c = \lambda_c = 0$ at $\partial\Omega \times (0,T)$ and, therefore, the boundary integral here is equal to zero. Hence, using the Fundamental Lemma of the Calculus of Variations, the strong form of the adjoint sediment concentration equation is

$$-\frac{\partial \lambda_c}{\partial t} - \nabla_h\cdot(F_{\text{corr}}\lambda_c\mathbf{u}) - \epsilon_s\nabla_h^2\lambda_c + D\lambda_c = \frac{\partial J^T}{\partial c}, \tag{A.15}$$

in $\Omega \times (0,T)$ with boundary conditions

$$\lambda_c = 0 \qquad \text{at } \partial\Omega\times(0,T), \tag{A.16a}$$

$$\lambda_c = 0 \qquad \text{at } \Omega\times\{T\}, \tag{A.16b}$$

and $\hat{\alpha}_3 = \lambda_c$ at $\Omega \times \{0\}$. Note that (A.16a) comes from the definition of the function space W. As with the hydrodynamic equations, we assume that we have a closed system to ensure that the problem is well-posed.

In summary, the adjoint equation (A.15) is similar to the forward sediment concentration equation, but with reversed time and flow derivatives. Additionally, like with the hydrodynamic equations, a final condition at $t = T$ is specified rather than an initial condition, because of the backwards propagation of information.

A.3 Exner equation

The final equation in the system is the Exner equation (2.58), which is

$$\frac{(1-p')}{m_f}\frac{dz_b}{dt} + \nabla_h\cdot\mathbf{Q_b} = D_b - E_b, \tag{A.17}$$

where all variables are as defined in Section 2.2. This equation is evaluated on the domain $\Omega \times (0,T)$ with a generic Dirichlet boundary condition $z_b|_{\partial\Omega\times(0,T)} = g_z$ and an initial condition of $z_b = z_0$ at

$\Omega \times \{0\}$. Deriving the continuous adjoint of this equation is very complicated because the formulae for D_b, E_b and $\mathbf{Q_{b*}}$ have several very complex non-linear dependencies on the bedlevel, z_b. Therefore, using a programme like *pyadjoint*, as we do, is the only practical way to find the adjoint of this equation. Given these difficulties and the fact that using the implementation approach taken in this work makes it unnecessary in practice, here we do not derive the continuous adjoint of the Exner equation.

The adjoint form of the Exner equation also depends on whether the slope effect and secondary current corrections from Section 2.2.1.2 are applied to the bedload transport flux $\mathbf{Q_b}$, as these corrections introduce derivatives of z_b. We derive, as an example, the continuous adjoint for the Exner equation with the slope effect magnitude correction. Note we assume for simplicity that only the time derivative and the correction depend on z_b and, thus, this is **not** the correct continuous adjoint equation, only an illustration.

If the slope effect magnitude correction is applied, then $\mathbf{Q_b}$ is multiplied by $\left(1 - \Upsilon \frac{\partial z_b}{\partial s}\right)$ (see 2.44), where $\frac{\partial}{\partial s} = \cos\xi \frac{\partial}{\partial x} + \sin\xi \frac{\partial}{\partial y}$ and ξ is the flow angle. The equation system can be written in the form $F(z_b) = 0$, where z_b is a member of the function space $W_{g_z} = \{w \in H^2(\Omega)\ s.t.\ w|_{\partial\Omega\times(0,T)} = g_z\}$. Then, using (A.5) as before, the weak form of the adjoint equation is

$$\int_{\Omega\times(0,T)} \lambda_z \left(\frac{(1-p')}{m_f} \frac{\partial v_z}{\partial t} - \nabla_h \cdot \left(\Upsilon \mathbf{Q_{b*}} \frac{\partial v_z}{\partial s} \right) \right) + \int_{\Omega\times\{0\}} \hat{\alpha}_4 v_z \, dx = \int_{\Omega\times(0,T)} \frac{\partial J^T}{\partial z_b} v_z \quad \forall v_z \in W, \tag{A.18}$$

where v_z is a test function for the bedlevel, $\lambda_z \in W$ is the adjoint solution, $\hat{\alpha}_4$ is a parameter to be determined and $\mathbf{Q_{b*}}$ is the original unmodified bedload transport. As with the previous equations, we integrate by parts which gives

$$\begin{aligned} &\int_{\Omega\times(0,T)} -v_z \frac{(1-p')}{m_f} \frac{\partial \lambda_z}{\partial t} \, dx\, dt + \int_{\partial\Omega\times(0,T)} \left(v_z \frac{\partial \lambda_z}{\partial s} \Upsilon \mathbf{Q_{b*}} \cdot \mathbf{n} - \Upsilon(\cos\xi + \sin\xi) \lambda_z \mathbf{Q_{b*}} \cdot \nabla_h v_z \right) ds\, dt \\ &\quad + \int_{\Omega\times\{T\}} \frac{(1-p')}{m_f} \lambda_z v_z \, dx + \int_{\Omega\times\{0\}} \left(\hat{\alpha}_4 - \frac{(1-p')}{m_f} \lambda_z \right) v_z \, dx = \int_{\Omega\times(0,T)} \frac{\partial J^T}{\partial z_b} v_z \quad \forall v_z \in W. \end{aligned} \tag{A.19}$$

By definition of W, $v_z = \lambda_z = 0$ at $\partial\Omega \times (0,T)$ and, therefore, the boundary integral here is equal to zero. Hence, using the Fundamental Lemma of the Calculus of Variations, the strong form of the adjoint Exner equation with a slope effect magnitude correction is

$$-\frac{(1-p')}{m_f} \frac{d\lambda_z}{dt} - \frac{\partial}{\partial s} \left(\Upsilon \mathbf{Q_{b*}} \cdot \nabla_h \lambda_z \right) = 0, \tag{A.20}$$

with boundary conditions

$$\lambda_z = 0 \text{ at } \partial\Omega \times (0,T), \tag{A.21a}$$

$$\lambda_z = 0 \text{ at } \Omega \times \{T\}, \tag{A.21b}$$

and $\hat{\alpha}_4 = \frac{(1-p')}{m_f} \lambda_z$. Note that (A.21a) comes from the definition of the function space W.

In summary, the adjoint equation (A.20) is similar to the forward equation, but the order in which the spatial derivatives are applied is switched and the time derivative is reversed. If, instead, we had chosen

to apply either the slope effect angle correction or the secondary current correction, then the effect on the derivatives would have been similar, because the corrections are all applied in a similar way (see Section 2.2.1.2). Note also that, as with the previous adjoint equations, a final condition at $t = T$ is specified rather than an initial condition because of the backwards propagation of information.