Modelling dynamics of Nature based Solutions

Dynamic Vegetation Module

Deltares Systems

Using Delft3D Flexible Mesh

Deltares

Enabling Delta Life

User Manual

Modelling dynamics of Nature-based Solutions

Dynamic Vegetation Modelling using Delft3D Flexible Mesh

User Manual

Version: 0.1
Revision: 00

16 July 2024

Modelling dynamics of Nature-based Solutions, User Manual

Contents

List of Tables

List of Figures

1 Introduction

1.1 Purpose of this modelling suite

Faced with the problems of climate change and socio-economic pressure in many of the world's deltas and rivers, there is a rising interest in nature based solutions (NBS) as a means to combine affordable and adaptive flood risk reduction with other ecosystem services. This has created a demand for tools that can quantify the development and performance of natural or nature-based systems like salt marshes, mangroves and river floodplains. These ecosystems are known to have rapid geomorphological development, in the time scale of decades that interferes with the time scale over which they are expected to provide their flood risk reduction functions. In addition, it has been well described that these dynamics result from a two-way interaction between ecological and physical (hydrodynamic and morphodynamic) processes. Thus, a coupled modelling of bio-geo-morphological systems is needed in order to design and evaluate NBS under scenarios of climate change. This will help to create confidence in the long-term resilience and actual performance of such systems.

Over the last decade Deltares, in collaboration with universities and other research institutes, has developed and applied modelling tools and approaches that simulate the effects of biota in numerical models such as Delft3D and XBeach. Vice versa, the effect of environmental conditions on the development of organisms has been incorporated in D-Water quality and other growth models. The combination of physical and biological models has been applied in several projects. This combination often provided important quantification of anticipated behaviour in academic studies (e.g. Temmerman et al. (2007) and Van Oorschot et al. (2015)), but the practical complexity impeded simple, effective assessments and replicability.

1.2 Philosophies of this modelling suite

In this modelling suite, the ecological model is developed in Python and is coupled an existing model for hydrodynamics, waves or morphology, such as D-Flow FM, D-Waves or XBeach. This manual focusses on the use of the ecological Python model in combination with D-Flow FM (and coupling to D-Waves) — all open source software. By doing so a fast and flexible coupling platform was created to answer questions about the design and evaluation of NBS, thereby enabling assessments that were considered too complicated and costly before.

The ecological model can be as simple as a single habitat suitability rule, or endlessly complex involving multiple species, age classes and stressors. To facilitate the assessment of ecological processes, Delft3D Flexible Mesh has been expanded with summarizing statistics of ecologically relevant parameters (e.g. inundation time, bed shear stress), which elevate the need to store and analyse lengthy time series. In this manual vegetation development is computed in the ecological model. Nevertheless, the ecological model is not restricted to vegetation only. Other biota that interact with hydrodynamics or morphology, such as mussels, algae and microfytobenthos, can also be computed with this modelling suite.

A substantial improvement over the earlier academic tools is the direct exchange of parameters through memory pointers, instead of via files. This is much faster, allows for flexible exchange intervals (i.e. only when really needed) and allows for relatively independent development of both parts of the coupled code. Consequently, coupling an ecological model with other compatible hydrodynamic models is possible if one knows that model's requirements on vegetation specification.

1.3 Purpose and limitations of this guideline

The interdisciplinary nature of biogeomorphological modelling means that it rarely is a topic of standard ecological or hydrodynamic modelling courses or textbooks. Therefore, this guideline aims to give a bit more explanation on key principles and experiences relevant for a useful model setup, rather than just describing how things work technically. When reading this manual and setting up your model, please consider the following:

⬦ The reason for designing this model suite is the assessment of systems where biota affect physics considerably; not to primarily model the behaviour or conditions of biota.

⬦ A basic level of numerical modelling expertise is considered a prerequisite for using these advanced tools. Consequently, basic modelling considerations are not discussed here.

⬦ The meaning of the words 'vegetation' and 'plants' often extends to sessile biota (so including macrofauna like e.g. oysters) as some effects on physics can be very similar. In practice, we found that most projects predominantly deal with flora, only few with fauna.

⬦ The actual implementation of vegetation (physical equations and input file formats) is described in the Technical and/or User Manuals of the involved hydrodynamic models.

⬦ This modelling suite deals with common physical feedbacks and conditions only since these are typically the most important and well-known factors for biota survival. If considered relevant, other factors (stressors) such as nutrients, pH and predation can be prescribed statically, their effects can be incorporated in survival coefficients, or dynamically included by linking to specific models via the Python interface. So far, the development team has no experience in doing so.

⬦ The range of ecosystems where biogeomorphological models can be applied is large. Likewise, the required processes and complexity of such models. Whilst this guideline aims to describe a basic set of generic tools, co-developed with a group of users, it is impossible to cover all possibilities and up to the user to expand functionalities if necessary.

⬦ To improve the transparency and re-usability of models including species or location-specific settings, we aim to set up a platform for easy archiving and sharing, as well as discussion between users.

⬦ This document and the modelling suite are intended to be improved and extended regularly based on user needs and inputs. The development team greatly appreciates feedback on modelling needs and new functionalities.

1.4 Contributing projects

These projects have benefited from and/or contributed to software development, testing or contents of this guideline:

⬦ EU-Hydralab+
⬦ Deltares strategic research program Nature Based Flood Defences
⬦ 'Kennisimpuls grote wateren'
⬦ RAAK-PRO project 'Meer Waarde met Mosselen'
⬦ EU KP7 project REFORM (PhD thesis Mijke van Oorschot)
⬦ NWO project BeSafe (PhD thesis Pim Willemsen)
⬦ Ongoing PhD project Üwe Best
⬦ Mangrove-RESCUE: Mangrove Resilience for Enhanced Safety of Coastal Urbanisations and Environments

2 Getting Started

2.1 Conceptual description

As introduced in section 1.2, the vegetation module in Python is coupled to a hydrodynamic Delft3D Flexible Mesh (Delft3D FM) model. In this suite, the Python environment has two functions: it contains the vegetation module and it 'orchestrates' the interaction between the hydrodynamic and the vegetation modules. 'Orchestrating' means that Python is used to define when and via which parameters the models interact. To do so, it uses the Basic Model Interface technique (BMI), which is described in more detail in section 2.2.1.

The modelling environment is illustrated by a conceptual schematization in Figure 2.1. The hydrodynamic module (D-Flow FM) computes for example the water level, flow velocity and bed shear stress and passes these results to the vegetation module (Python). The vegetation module then computes the vegetation biomass, expressed as a stem density, stem height and stem diameter. These vegetation parameters are used as inputs for D-Flow FM in the next timestep.
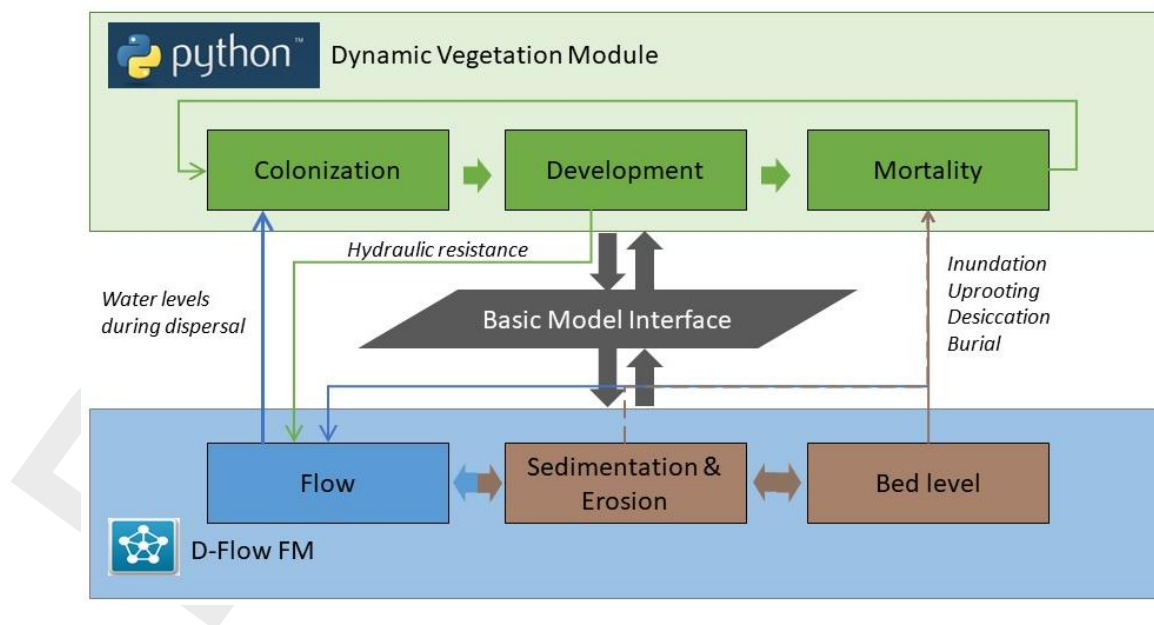
Figure 2.1: Conceptual schematization of the modelling environment.

2.2 Software

2.2.1 Basic Model Interface (BMI)

The vegetation growth module in Python is coupled to D-Flow FM through BMI, a technique developed by the Community Surface Models Development System (CSDMS[1]) and described by Peckham *et al.* (2013). BMI exchanges memory location pointers between the D-Flow FM and Python models. So, model parameters and (state) variables are exchanged through memory, avoiding any file writing and reading when switching between the Python model and D-Flow FM model and vice versa.

Two main BMI functions are used:

1 Initialising the coupled model, running the model for a user-defined time period and final-

---

[1]Source: https://csdms.colorado.edu/wiki/BMI_Description

izing the model.
2 The exchange of variables between the vegetation module (in Python) and the hydrody-
   namic module (in Delft3D Flexible Mesh).

2.2.2 Python

We recommend to use Python version 3.7 and to work through the Spyder interface.

2.2.3 Installation

In Python, for the basic setup a number of packages must be installed. Most packages can be
installed via the default method in Anaconda. Aside from some commonly used Python pack-
ages (e.g. Numpy, Scipy, Matplotlib), the use of BMI requires the installation of two additional
packages: *netCDF4* and *faulthandler*. This can be done in Anaconda prior to the installation
of BMI.

The only 'special' case is the BMI wrapper, which creates the interface between the Python
code and the D-Flow FM code. In our experience installation was easiest by using 'pip'. The
BMI wrapper can be downloaded from GitHub[2]. Installation instructions are provided in the
package files and are also included in Appendix A.

---

[2]Link: https://github.com/csdms/bmi

3 Tutorial

The functionality of dynamic vegetation has very recently been developed and can be used with D-Flow FM. A tutorial will be added soon.

4 Processes

4.1 Generic vegetation development processes

The field of biogeomorphology focuses predominantly on interactions between organisms and their environment, i.e. ecology, however basic biological processes like growth need to be understood and described to be able to assess changes over time. Numerical modelling of ecology involves two basic steps: First, the construction of a conceptual model based on knowledge of the natural system, and second, the translation of this concept in mathematical equations (Soetaert and Herman, 2009) that can be solved by numerical methods. The conceptual model involves the identification of main components, also known as state variables (e.g. biomass of riparian trees, number of seagrass shoots per horizontal area), and the flows, or ecological interactions, that describe the exchange of energy or matter between them. In common with morphological modelling, the rate of change of the state variables is governed by the principle of conservation, i.e. the sum of the flows entering and leaving a compartment. The ecological interactions can be described by the product of a maximal rate times the compartment doing the work, times appropriate limiting terms that represent the forcing done by the environment. These principles apply to a wide range of scales of ecological modelling. In biogeomorphological modelling, typically the eco-element to reach scales (meters – tens of kilometres; days to decades) are of interest, not individual organism or river basin development (Baptist, 2005).

Intertidal vegetation can colonize an area either by seedling establishment (Bouma *et al.*, 2016) or clonal growth from already established plants (Silinski *et al.*, 2016). Whether vegetation can establish and subsequently grow depends on local biophysical processes such as inundation time and frequency (Balke *et al.*, 2016) and bed level change (Willemsen *et al.*, 2018, 2022). When those dynamic processes remain below certain spatiotemporal variable thresholds, vegetation is able to establish and grow. Bare tidal flats are first colonized by pioneer vegetation species such as *Salicornia* spp. and *Spartina* spp. in salt marshes and *Sonneratia* spp. and *Avicennia* spp. in mangroves. Once the pioneer salt marsh further develops, the bed elevation increases due to sediment trapping, and physical stresses are attenuated due to the establishing ecosystem, more vegetation species follow.

In mathematical form, the differential equation used by e.g. Akerboom (2018) and Carr *et al.* (2010) to describe seagrass growth reads:

$$\frac{\partial N}{\partial t} = N(t) \left(\max_{Ngrow} F_I(I) F_{phot}(T) N_{lim} R_{lim} - N_{loss} \right) \tag{4.1}$$

where

dN/dt is the rate of change of number of seagrass shoots N,
$N(t)$ the number of shoots a time t in a compartment, and

the part between brackets the maximal rate ($\max_{Ngrow}$) times limiting terms minus the shoot mortality ratio N_{loss} that is independent of environmental conditions. Limiting terms here are photosynthesis inhibited by irradiation $I(F_I)$ or temperature $T(F_{phot})$, a maximum shoot density N_{lim} and a maximum aboveground to belowground biomass ratio R_{lim}.

Other growth prohibitors or loss terms can be added in a similar fashion, to account for stresses due to, for example, wave motion, grazing or salinity. Likewise, the plant growth model can be expanded with additional equations that account for the development of specific parts (roots, leaves, stem thickness) of the plant. However, if such processes are not well known or expected to be only marginally relevant to the problem of interest then they should not be included; keeping a model simple whilst realistic is a significant challenge but crucial

to the successful generation and interpretation of results (cf. Levin (1987), Phillips (1995) and many others).

Plant development involves a number of processes; whether or not all of these needs to be incorporated in a model depends on the purpose and timeframe of the study:

◇ Growth and mortality. On the population level, these processes are always intrinsically active but can be limited or stimulated by environmental factors. Even in healthy populations, plants or part of plants die of old age or other factors that are not modelled explicitly such as diseases.
◇ Stress. Stress occurs when optimal growing conditions are not fully met, or when a negative pressure is exerted. Healthy growing conditions are associated with basic plant needs (resources): temperature, light, carbon dioxide and water for photosynthesis, oxygen for respiration and nutrients, minerals for structural formation and soil for anchoring. Typical stresses are light limitation for aquatic plants or undergrowth, drought or inundation for terrestrial plants, high levels of harmful pollutants (e.g. salt, heavy metals), erosion and deposition, grazing, fire and physical damage by flows or waves.
◇ Establishment and further spatial distribution. Plants can spread out via different mechanisms: seeding, dispersal of vegetative elements and via the root system (clonal growth or the creation of bulbs). The latter is by definition a local process, often responsible for slow but steady expansion and less sensitive to the occurrence of suitable conditions. The first two can cover substantial distances and can be related to flow patterns and suitable (lack of) dynamics, so called windows of opportunity for establishment.
◇ Competition. Whereas competition is an interaction between organisms rather than organisms and environment, it does affect community structure and therefore the interaction of biota with the physical environment.

4.2 Types of biota models

Several types of models can be used to simulate the development of biota to inform the hydrodyamic model. Most experience has been acquired with using population dynamics (PD) based on differential equations describing the change in average abundance (population density) of a population (a group of individuals of the same species) in space and time. In this context, the space descritisation is typically a grid cell of the hydrodynamic model. This is a fairly high-level or top-down approach of keeping track of relevant variables such as size, age and spatial density. This approach has been widely applied in ecology, forestry and conservation studies on a variety of populations (e.g. trees, insects, fish). Also, agent- or individual based models (ABM or IBM) have been linked to hydrodynamic models. This is a more bottom-up approach. Individual based models can incorporate individual-level mechanisms and are thus used whenever one or more of the following aspects, which are hard or impossible to represent in population-level differential equations, are considered essential for answering a research question or solving an applied problem: variation among individuals and of individuals during their life cycle; local interactions among individuals; and adaptive behaviour, which includes physiology and energy budgets. The behaviour of biogemorphological systems has also been mimicked by cellular automata. These are grid-based by definition and offer an appealing visualisation combined with a straightforward interaction between biotics and abiotics but rely on a rule-based description of interactions. Consequently, the modelled development does not lend itself to quantitative studies in strongly changing environments where the prescribed rules may not be valid anymore.

4.3 Salt marshes

In the salt marsh module, two vegetation development models are available: Population dynamics (Temmerman *et al.*, 2007) and Windows of Opportunity (Bouma *et al.*, 2015). Generally the Windows of Opportunity model can be applied on a short-term timescale (i.e. season to years) and the Population Dynamics model on a long-term timescale (i.e. years to decades). The Windows of Opportunity model is implemented by (Odink *et al.*, 2019). The Population dynamics model is implemented by (Willemsen *et al.*, 2022) and described below.

Vegetation establishment and growth for salt marsh vegetation, *Spartina*, is described by the population dynamics balance Equation (4.2). The change of the total stem density over time was calculated as a result of vegetation establishment Equation (4.3), lateral expansion of plants through diffusion to neighbouring cells Equation (4.4), clonal growth of plants up to the maximum carrying capacity by using a logistic function Equation (4.5), plant mortality caused by excessive bed shear stresses as a result of currents and waves Equation (4.6) and plant mortality due to inundation stress Equation (4.7).

$$\frac{\partial n_b}{\partial t} = \left(\frac{\partial n_b}{\partial t}\right)_{est} + \left(\frac{\partial n_b}{\partial t}\right)_{diff} + \left(\frac{\partial n_b}{\partial t}\right)_{growth} - \left(\frac{\partial n_b}{\partial t}\right)_{flowwave} - \left(\frac{\partial n_b}{\partial t}\right)_{inund} \tag{4.2}$$

$$\left(\frac{\partial n_b}{\partial t}\right)_{est} = r_{01}(P_{est}) \cdot n_{b,0} \tag{4.3}$$

$$\left(\frac{\partial n_b}{\partial t}\right)_{diff} = D\left(\frac{\partial^2 n_b}{\partial x^2} + \frac{\partial^2 n_b}{\partial y^2}\right) \tag{4.4}$$

$$\left(\frac{\partial n_b}{\partial t}\right)_{growth} = r \cdot \left(1 - \frac{n_b}{K}\right) n_b \tag{4.5}$$

$$\left(\frac{\partial n_b}{\partial t}\right)_{flowwave} = -n_b \cdot C_\tau \cdot (\tau - \tau_{cr,p}), \quad \text{when} \quad \tau > \tau_{cr,p} \tag{4.6}$$

$$\left(\frac{\partial n_b}{\partial t}\right)_{inund} = -n_b \cdot C_{inund} \cdot (H - H_{cr,p}), \quad \text{when} \quad H > H_{cr,p} \tag{4.7}$$

where

$\frac{\partial n_b}{\partial t}$	represents the change of the stem density per cell over time $[stems\ \text{m}^{-2}\,\text{d}^{-1}]$.
$r_{01}(P_{est})$	is a function generating at random either a 0 (with probability $1 - P_{est}$) or a 1 (with probability P_{est}).
$n_{b,0}$	is the initial stem density [m^{-2}],
D	is the plant diffusion coefficient [m^{-2} yr^{-1}],
x and y	are the horizontal spatial coordinates [m],
r	is the intrinsic growth rate of the stem density [d^{-1}],
K	is the maximum carrying capacity of the stem density [m^{-2}],
C_τ	is the plant mortality coefficient due to bed shear stress [d^{-1} (N m^{-2})$^{-1}$],
τ	is the bed shear stress exerted by flow and waves [N m^{-2}],
$\tau_{cr,p}$	is the critical bed shear stress for plant mortality [N m^{-2}],
C_{inund}	is the plant mortality coefficient due to inundation stress [d^{-1} m^{-1}],
H	is the inundation height [m], and
$H_{cr,p}$	is the critical inundation height at high tide [m].

4.4 Mangroves

Two types of mangrove models have been developed by PhD students: One based on the work by Van Maanen *et al.* (2015) that accounts for mangrove colonization, growth and mortality, established to describe the the development of *Avicennia marina* and *Rhizophora mangle* based on the mangrove population model by Berger and Hildenbrandt (2000). This model is actually a combination of grid-based and individual-based approaches, allowing for the influence of neighbourhood effects (like an IBM) whilst aggegating properties such as age and trunk size over a grid cell. Mangroves establishment is governed by the hydroperiod (sufficient dry time) in a particular grid cell. After establishment, their number decreases but their stem diameter -the state variable- and associated tree height increase, limited by stresses caused by insufficient or excessive inundation and competition for resources with neighbouring trees. The other approach is based on directly linking Mesofon (Grueters *et al.*, 2014). This model is also based on the approach described by Berger and Hildenbrandt (2000) but has been developed further and takes more complex neighbourhood interactions into account and allows for more species. Both adapted mangrove models are still awaiting publication by the respective PhD students. Therefore they are not elaborately described here yet. If you are interested in collaborating on the application and development of these models, please contact us.

4.5 Riparian vegetation

The Riparian Vegetation Module contains formulations to model the colonization, development and mortality of riparian vegetation in river floodplains (Figure 2.1). The processes in the vegetation module are based on the PhD study from Van Oorschot (2017) and this model has been applied in several studies of different river systems (**?**Van Oorschot *et al.*, 2017, 2018; Martínez-Fernández *et al.*, 2018; Kleinhans *et al.*, 2018) and estuaries (Lokhorst *et al.*, 2018; Brückner *et al.*, 2019). The module described here mostly contains similar processes and assumptions. Currently, the module is suitable for modeling riparian trees such as willows and poplars. An addition for undergrowth (grasses and shrubs) with seasonal behaviour is in development.

In the sections below, a description is given for each of the processes in the model. The user-defined parameters are stored in a vegetation info file (<VegetationInfo.csv>) and can be manually adjusted to the vegetation type that is modeled. The default file contains values for a general Salix and a Populus vegetation type. Vegetation parameters that can be adjusted are written in green and parameters from the D-Flow FM model are written in purple.

4.5.1 Colonization

Colonization of vegetation takes place during the seed dispersal period dispersal_period. If a cell is currently dry and was wet in the previous time step, seeds are deposited in cells that have room for establishment. Dry and wet cells are defined by a wetdry threshold Epshu which is taken from the D-Flow FM <*.mdu>-file. When cells are already completely filled with older vegetation, new vegetation cannot settle. Vegetation is added to a cell with a maximum fraction ini_fraction. For each newly colonized vegetation type, the current bed level of the grid-cell is recorded. This value is used to determine the amount of future sedimentation and erosion in the grid-cell which is used for calculating the mortality due to burial or uprooting (section 4.5.4). The following actions are performed during colonization:

◇ Summing fractions of all vegetation types and ages per grid-cell
◇ Check if current time is within seed dispersal window of one of the defined vegetation types
◇ Check available space in each grid-cell
◇ Check if grid-cell is now dry; water depth $< \varepsilon_{hu}$ and was previously wet; maximum water depth $t - 1 > \varepsilon_{hu}$.

$\diamond$ If all conditions are met: vegetation is added to the grid-cell up to the maximum defined fraction

$\diamond$ Colonization takes place in small steps to ensure evenly divided vegetation when there are multiple vegetation types.

4.5.2 Growth

Two types of growth are currently implemented: growth of seedlings and growth of older vegetation. The growth of seedlings is rapid and their size is therefore updated at each vegetation time step. The size of vegetation older than one year is updated each year. Seedling growth is represented by a sigmoid curve, represented by the seedling_sigmoid_fit parameters of each vegetation type. The seedling vegetation parameters that are affected by growth are shoot and root. Stem diameter changes are relatively small and therefore not taken into account in the first year. Seedling shoot growth is calculated with the following formula:

$$s(t) = \frac{c}{1 + e^{(a + b \cdot t)}} + d \tag{4.8}$$

where

$s(t)$ is the shoot height [m] at time t [days],
a and b are sigmoid growth parameters,
c is the maximum seedling shoot height [m] and
d is the initial shoot height after colonization [m].

Seedling root growth is calculated with a logarithmic growth curve with the following formula:

$$r(t) = (c - d) \cdot (1 - e^{-k \cdot t}) + d \tag{4.9}$$

where

$r(t)$ is the root length [m] at time t (days),
k is the fitting parameter,
c is the maximum root length for seedlings [m] and
d is the initial root length after colonization [m].

The growth of shoot, root and stem diameter for older vegetation is based on a predefined logarithmic growth curve with the following formula:

$$s(a) = g \cdot {}^{10}\log(a) \tag{4.10}$$

where $s(a)$ is size of the shoot, root or stem diameter [m] at age a [year], g is the growth parameter. The growth is predefined by an array with the size of the maximum age, max_age, of the corresponding vegetation type. Each year at the beginning of the month defined in shifting_month, the vegetation ages and the size is updated.

4.5.3 Interaction

The current version of BMI does not support the use of multiple vegetation types in one cell. Therefore, a workaround was used that calculates simplified vegetation characteristics per year over all vegetation ages and vegetation types. To include vegetation, Vegetationmodelnr in the <∗.mdu>-file has to be set to 1. This represents the use of the Baptist *et al.* (2007) equation for the calculation of vegetation roughness (Equation (4.11)). Hydraulic resistance caused by vegetation was calculated in each grid cell for flow through vegetation and flow above vegetation.

$$C = \frac{1}{\sqrt{\frac{1}{C_b^2} + \frac{C_d n h_v}{2g}}} + \frac{\sqrt{g}}{\kappa} \ln \frac{h}{h_v} \tag{4.11}$$

where

C	is the Chézy value of the vegetation [m$^{\frac{1}{2}}$ s^{-1}],
C_b	is the Chézy value for the un-vegetated parts,
c_d	is the drag coefficient,
n	is the vegetation density (stem diameter times number of stems per m^2),
h_v	is the height of the vegetation [m],
h	is the water depth [m],
κ	is the Von Kármán constant (0.41) and
g	is the gravitational acceleration (9.81 m s^{-2}).

The vegetation module generates fraction weighted values for stem diameter ($diaveg$), number of stems per m^2 ($rnveg$) and stem height ($stemheight$) and feeds this into the D-Flow FM model per vegetation time step.

4.5.4 Mortality

Vegetation mortality can be caused by uprooting, burial, flooding or desiccation and is age-dependent. Uprooting takes place if the erosion is larger than the length of the vegetation root times an uproot factor (Equation (4.12)). The uproot_factor is the fraction of the root that has to be exposed to be uprooted.

$$Uprooting = erosion > (r \cdot ur) \tag{4.12}$$
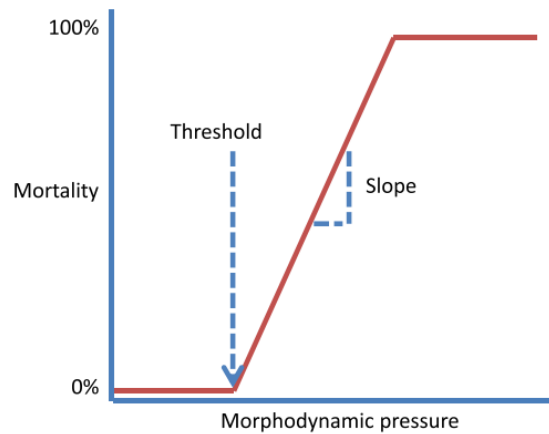
where

r	is the root length [m] and
ur	is the uproot_factor.

Mortality by burial takes place when a shoot is completely covered with sediment (Equation 4.13).

$$Uprooting = sedimentation > s \tag{4.13}$$

where s is the shoot height [m].

Mortality by flooding starts to take place if the consecutive flooding in a grid-cell exceeds a threshold. The threshold and rate at which mortality takes place is dependent on vegetation age and vegetation type (Figure 4.1). A cell is considered wet if the water level is higher than ε_{hu}. Flooding mortality takes place if the consecutive wet period of a grid cell exceeds the threshold. This is calculated in several steps. First the current survival is calculated with the threshold value, a slope determining the rate of mortality and the consecutive flooding days of the grid-cell (Equation (4.14)). Because the survival rate is calculated incremental, the ratio between the previous and the current survived fraction determines the amount of mortality. The previous survival is calculated with Equation (4.15).

Figure 4.1: Method for calculating mortality by flooding and desiccation. Vegetation starts to die after a threshold is exceeded with a rate that is determined by the slope.

$$sr_{(t)} = 100 - (fd - ft) \cdot fs \qquad (4.14)$$
$$sr_{(t-1)} = 100 - ((fd - ts) - ts) \cdot fs \qquad (4.15)$$

where

sr	is the amount of vegetation that survives in a cell [%],
fd	is the amount of consecutive days a grid-cell is flooded,
ft	is the flooding mortality threshold flood_thrshld and
fs	is the flooding rate flood_slp, ts is the timestep of the vegetation processes (days).

All values higher than 100 are reset to 100 and all values lower than 0 are reset to 0. The final fraction of vegetation that persists in a cell is calculated by Equation (4.16).

$$f = f_{t-1} \frac{sr_{(t)}}{sr_{(t-1)}} \qquad (4.16)$$

where f is the fraction of vegetation that is left after mortality is calculated.

Mortality by desiccation is calculated in a similar matter as the flooding mortality, only using the subsequent days a grid-cell was dry in combination with the groundwater level, the root length and the capillary fringe (Equation (4.17)). Vegetation only experiences drought stress if the root length cannot reach the groundwater level (Figure 4.2). The groundwater level in a floodplain cell is calculated by distance averaging the water level of the 3 nearest water containing grid-cells.

$$\text{dry if: } gwl + cf < r \qquad (4.17)$$

where

gwl	is the groundwater level [m],
cf	is the capillary_fringe,
r	is the root length [m].
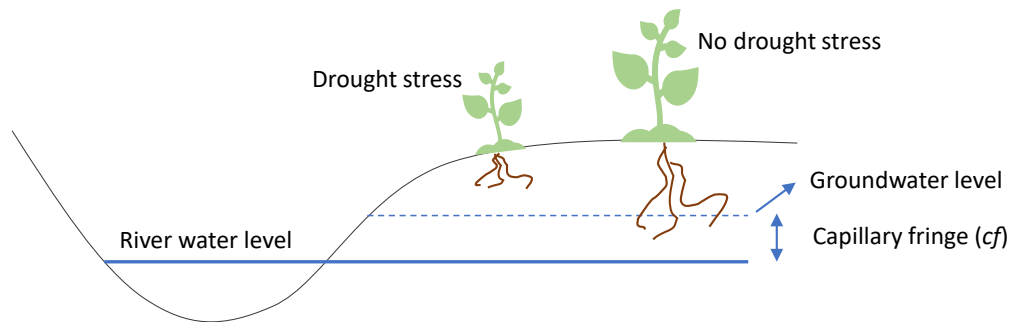
Figure 4.2: *Schematic representation of the groundwater level related to vegetation drought stress.*

4.5.5 Time-scales

The length of the vegetation time step can be set by adapting the VegTS parameter. The amount of vegetation steps that are simulated can be manually determined by setting DefineVeg = 'Y'and setting VegTimeSteps variable, or by letting the model automatically calculate it considering the Start and Stop time of the simulation from the <∗.mdu>-file. For this DefineVeg = 'N' has to be defined. When a Morfac is used, the amount of days a cell is flooded or dry is corrected by the Morfac by calculating it back to the realistic timescale to obtain realistic mortality processes.

4.5.6 Setting vegetation parameters

The vegetation module uses the <∗.csv>-file (<VegetationInfo.csv>) as input. This file should be added to the main model folder. Currently, the model only supports riparian trees. The default file contains values for a general Salix and a Populus vegetation type. Several riparian tree species can be added to the model. For each species all parameters should be added. Some parameters change during the life-span of the tree, these parameters have different values for each life stage (LF). In the first column, the species name is given, this name should be added in the column before each parameter. An explanation for each vegetation parameter is given in Table 4.1. For more details see the chapter on the description of processes.

Table 4.1: *Explanation of the vegetation parameters*

Parameter	Description	Life stage dependent
years_lifestage	The amount of years a species is in a certain life stage	yes
nr_stems	the amount of stems per (stems/m2)	yes
drag_cf	vegetation drag coefficient	yes
dry_thrshld	threshold for desiccation mortality process	yes
dry_slp	slope for desiccation mortality process	yes
flood_thrshld	threshold for flooding mortality process	yes
flood_slp	slope for flooding mortality process	yes
vel_thrshld	threshold for uprooting mortality process	yes
vel_slp	slope for uprooting mortality process	yes
root_depth_min	minimum rooting depth [m]	yes
root_depth_max	maximum rooting depth [m]	yes
uproot_factor	factor determines which fraction of the roots should be exposed before uprooting	yes
ini_fraction	initial fraction of vegetation in grid-cell after colonization	no
max_age	maximum age of vegetation type	no
factor_shoot	growth factor shoot	no
factor_diameter	growth factor stem diameter	no
factor_root	growth factor root	no
ini_shoot	initial size of shoot [m]	no
ini_diameter	initial size of stem diameter [m]	no
ini_root	initial size of root [m]	no
dispersal_period	[day no. start dispersal, month no. start dispersal, day no.end dispersal, month no. end dispersal]	no
shifting_month	month where colonization and aging of vegetation takes place	no
capillary_fringe	capillary fringe, i.e. water available above groundwater level [m]	no
seedling_sigmoid_fit	parameters to calculate sigmoid growth function for seedlings	no
seedling_root	parameters to calculate the root growth for seedlings	no

5 Technical description

5.1 Delft3D Flexible Mesh

The Delft3D Flexible Mesh Suite is a multidisciplinary software suite for 1D, 2D and 3D computations for coastal, river and estuarine areas. Simulations can include hydrodynamic flow, waves, water quality and ecology. See for an extensive description the manuals of D-Flow FM, D-Waves, and D-Morphology.

5.2 Python

BMI enables the exchange of different basic parameters that can be used in Python in a range of vegetation modules. Model geometry parameters describe the sizes and extent of the Delft3D Flexible Mesh model domain. Vegetation variables describe the dimensions of the vegetation used in the coupled model, and can be adapted in a vegetation module in Python and subsequently used in Delft3D Flexible Mesh for hydrodynamic and morphodynamic calculations. Finally, statistical variables describe the hydrodynamic development of the Delft3D Flexible Mesh model. A full list of exchangeable parameters is available in Appendix B.

Model geometry parameters (i.e. parameters without operations in Python):

`ndx = model.get_var ('ndx')`	number of boxes, including boundary boxes
`ndxi = model.get_var('ndxi')`	maximum bed shear stress [$N\,m^{-2}$] due to current and waves
`xzw = model.get_var('xzw')`	x coord. of the center of gravity of the boxes
`yzw = model.get_var('yzw')`	y coord. of the center of gravity of the boxes
`lnx = model.get_var('lnx')`	total number of links between boxes
`lnxi = model.get_var('lnxi')`	number of links between within-domain boxes
`ln = model.get_var('ln')`	link matrix between adjacent boxes [ln,2] matrix
`dx = model.get_var('dx')`	distance between the centers of adjacent boxes
`wu = model.get_var('wu')`	width of the interface between adjacent boxes
`ba = model.get_var('ba')`	surface area of the boxes

Physical vegetation variables:

`rnveg = model_dfm.get_var('rnveg')`	[$1\,m^{-2}$] 3D plant density, 2D part is basis input [$1\,m^{-2}$]
`diaveg = model_dfm.get_var('diaveg')`	[m] 3D plant diameter, 2D part is basis input [m]
`stemheight = model_dfm.get_var('stemheight')`	[m] 2D plant heights [m]
`tau_cre = model_dfm.get_var('TcrEro')`	[$N\,m^{-2}$] critical bed shear stress for erosion

Statistical variables in D-Flow FM/Python (i.e. fixed variables without operations in Python):

```
is_dtint = model_dfm.get_var('is_dtint')
```
number of D-Flow FM timesteps executed over a vegetation timestep

```
is_sumvalsnd = model_dfm.get_var('is_sumvalsnd')
```
sum of (1) bed shear stress, (2) flow velocity, (3) water depth per grid cell over vegetation timestep

```
is_maxvalsnd = model_dfm.get_var('is_maxvalsnd')
```
maximum of (1) bed shear stress, (2) flow velocity, (3) water depth per grid cell over vegetation timestep

5.3 Coupling D-Flow and D-Waves

A stepwise guide to set up a coupled Flow-Wave model is given below:

1 First, set up your D-Flow FM flow model including vegetation as explained in chapter 4.
N.B. Make sure to include block [veg] in MDU-file and vegetation parameters in <∗.ext>-file

2 Next, set up your D-Waves model in SWAN as you would without vegetation
Note that:

⋄ The computational grids of D-Flow FM and D-Waves are different, as unstructured grids are not supported for D-Waves. The coupling includes an interpolation between the computational grids, hence make sure that the resolution of your computational grid(s) in D-Waves is sufficient to link with your flow domain. If necessary you can apply multiple domains in D-Waves through nesting <reference to Nesting explanation, to be included>.
⋄ Larger domain extent for wave to prevent boundary wave effects?
⋄ Depth is defined positive in D-Waves (positive downward), where the bedlevels are defined as negative values in D-Flow FM (positive upward)

3 Include the following keywords for vegetation in your wave model input, <∗.mdw>:

```
FlowVegetation = 1      switch to turn on vegetation
VegSVNPlants = true     switch Spatially Varying Number of Plants
VegDrag = 0.1           spatially-uniform vegetation drag coeff.
```

N.B. In the current version of SWAN used in D-Waves (v40.81) only spatially uniform values for stem height, stem diameter or drag coefficient can be applied. In order to include spatially varying fields, a software update in the SWAN source code is necessary.

4 Set up dynamic vegetation rules in Python script, as explained in chapter 4

5 Orchestrating in Python with coupled Flow-Waves Orchestrating the model in Python through BMI works slightly different for a coupled D-Flow FM–SWAN model. As explained in section 2.1, two orchestrating functions can be distinguished:

⋄ Initializing, looping through and finalizing the model
⋄ Exchange of parameters between modules

The first function is performed by DIMR, hence the DIMR wrapper needs to be called in Python. The second function is performed by D-Flow FM, hence the D-Flow FM wrapper needs to be called in Python.

6 Time scales

6.1 Initial conditions and spin-up time

As with traditional morphological models, biogeomorphological models require time spin-up, i.e. to let the simulated environment to adapt to the prescribed forcing conditions. For hydro-dynamic models this spin-up time is typically a hours-days, depending on the domain and the forcing conditions. For morphological models, especially when fine sediment is involved, the adapation time hence the spin-up time is much longer; if the objective is not just to simulate bathymetric changes but to simulate concentrations and mud properties too, this can be up to decades. For biogeomorphological simulations, the required spin-up time also depends on the domain and the objective of the study. Studying the behaviour of an isolated nature-based solution (e.g. managed realignment) right after its construction requires barely any spin-up time since this starts from a disturbed environment, meaning that the initial conditions can be prescribed well, and that the environment is not in equilibrium at the start time of the simulation anyway. For studies into the long-term behaviour of estuarine or riverine systems, where the time-varying properties of the sediment and biota (e.g. bed composition, vegetation age) are crucial for th, years to decades are required to reach the equilibrium that can be considered as the starting point of the various study scenarios. Oftentimes, this cannot simply be overcome by describing the initial conditions precisely because those cannot be fully known. Even if these could be observed in the field (e.g. using remote sensing) over the entire model domain, there is no equilibrium yet with the process as described in the numerical model. In studies like these, it is good practice to develop a baseline scenario that has been allowed to reach (dynamic) equilibrium for multiple decades, possibly aided by quicker aggregated or coarser models in the first phase, which can be used to prescribe the initial conditions of the development scenarios to be studied.

6.2 Scaling biology in time in numerical models

The simplest way of scaling, i.e. accelerating, time in numerical models is to increase the computational power: a faster CPU will decrease runtimes of simulations. Given the limits to processor speeds, parallelization is often required to substantially decrease computation times. Parallelization means subdividing the computational domain in subdomains that each run on their own computational core. The necessary communication between these sub-domains requires some computational power too, consequently the increase in computation speed does not scale linearly with the number of processors.

Besides this straightforward 'brute force' approach, numerical models offer more advanced techniques for scaling both morphology and biology in time, independently. For morphology, which changes on timescales of minutes to decades versus the timescale of seconds to minutes of the hydrodynamic forcing, two distinct approaches are available: An 'online' approach, where morphological change occurs every hydrodynamic timestep (typically seconds), and an 'offline' approach that applies morphological change only after a given interval. In both cases, numerical models offer the advantage of using a morphological acceleration factor (Morfac; Roelvink (2006); Figure 6.1) that increases the rate of bed level change. For example, using a Morfac of 20 allows for studies of 20 years morphological development with only 1 year of computationally expensive hydrodynamics, whereas a Morfac of 26 allows for the use of one spring-neap cycle to simulate an entire year.
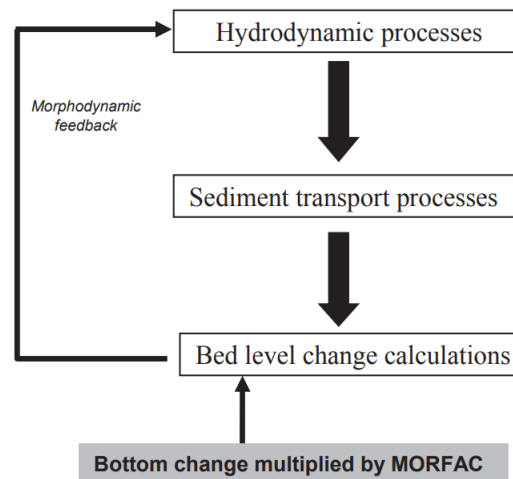
Figure 6.1: General structure of the Morfac concept in morphodynamic models (from Ranasinghe et al. (2010)).

The disadvantages of using a (large) Morfac are the risk of wiggles and numerical instabilities due to large rates of change (see example in Figure 6.2), and a temporal mismatch in the feedback relationships between hydrodynamics and morphology. The latter is particularly relevant for systems with considerable variations in forcing input. The Morfac chosen should be sufficiently small, based on the expected dynamics of the modelled system, and the simulation results should be checked for sensitivity to this setting to avoid instabilities to occur. Roelvink (2006) and Ranasinghe *et al.* (2010) amongst others provide guidelines on suitable morphological acceleration techniques.

For biology, which typically changes at even longer timescales (weeks to decades), this 'online' approach seldom makes sense, apart from situations where organisms can rapidly deplete nutrients or are exposed to rapidly changing conditions, such as algae modelling. For plants of interest in biogeomorphogical studies, development cannot meaningfully be simulated or measured at a timescale of seconds. Consequently, an online approach would accelerate small errors. Typically, a week is a more applicable and robust time interval in situations with strong feedbacks and relatively rapid growth such as seagrass beds or freshwater macrophytes. Whereas a week may be a suitable timescale for updating vegetation health or condition, variations within this period, e.g. hourly water levels related to light available for photosynthesis, are still required for correct modelling of vegetation development (e.g., Carr *et al.* (2010); Akerboom (2018), case study in Figure 6.3). For riparian, tree-like vegetation growth is slower and less governed by strong and frequent plant-water interaction, which allows for the use of longer interaction intervals hence larger biological scaling factors. As for morphology, numerical convergence remains an issue and needs to be addressed by choosing sufficiently small time steps and suitable iteration schemes for the differential equations that describe growth, to prevent overshooting.
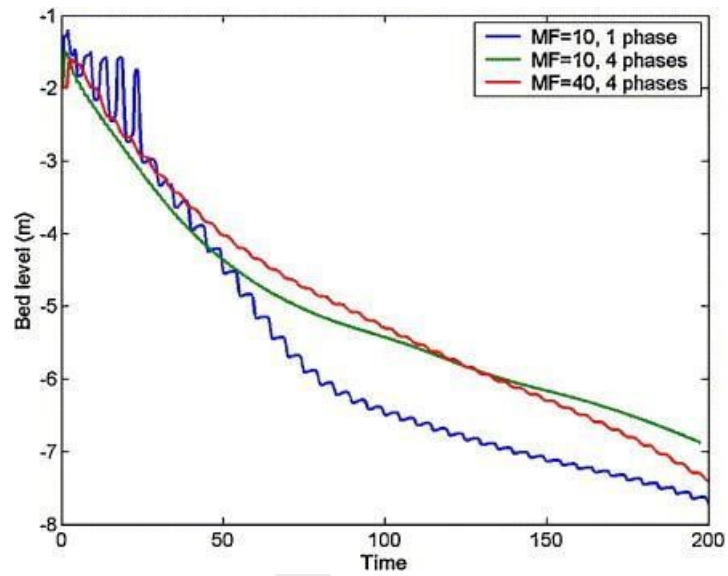
Figure 6.2: *An example of bed level development over time for different Morfac settings, showing wiggles and differences in final results for coarser approaches (from: Roelvink, 2009).*
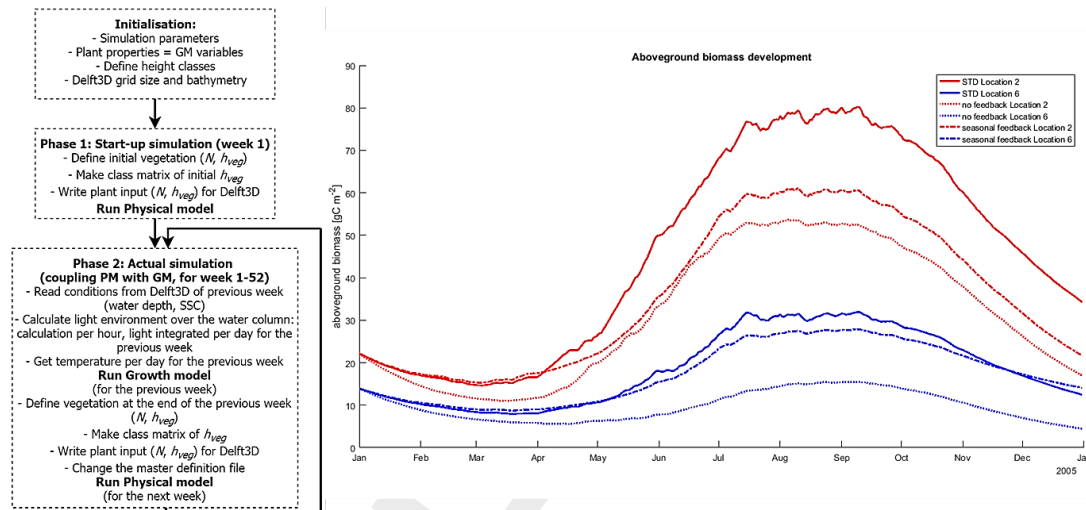
Figure 6.3: *Case study based on the Rødsand coastal lagoon. a) The flow diagram for the simulation of eelgrass development with weekly vegetation updating based on an hourly simulated light environment, illustrating the use of vegetation classes with hydrodynamically relevant properties. b) Eelgrass biomass development over a year, illustrating the effect of optimal coupling (STD; hourly light and weekly updating) of the physical model (PM in left panel) with the growth model (GM) (continuous lines), seasonal coupling (dash/dot line) and no updating (dotted line). The red lines are for a shallow location (1.6 m depth), the blue lines are deeper (3 m). Without time-dependent feedback, the biomass is substantially lower throughout the year as the initial low plant cover has a limited effect on sediment stabilisation, hence the water remains relatively turbid all year. The seasonal feedback initially leads to a marginally higher biomass in April as the feedback is not affected by eelgrass decline during winter. However, in the growing season growth is slower because the light climate does not benefit from the higher sediment stabilising biomass as it does in the STD simulation. In fall, the opposite occurs for the deeper location: instead of the weekly declining biomass, the high biomass from October acts as a stabiliser in the hydrodynamic model, giving a more favourable light climate and less rapid decline. (from: Akerboom (2018)).*

Because biological updating occurs at a very low frequency in comparison with hydrodynamic updating, the computation time of an ecological model is not normally an issue in biogeomorphological simulations. Nevertheless, including biology does introduce additional timescales (see also the next section 'Forcings: calm conditions vs storms and floods') that need to be taken into account correctly, possibly leading to a larger number of hydrodynamic timesteps to be calculated. The longest introduced timescale is that which is related to biological growth. For seasonal plants, which have a very different effects on flow in summer compared to winter, the example of a single spring-neap cycle with a Morfac of 26 to simulate a full year will not work. The shorter introduced timescale is the one related to death or establishment, which can range from seconds to weeks. The time required for such processes, i.e. the time that critical conditions occur, can be expressed in so-called windows of opportunity, e.g. Balke et al. (2011). By keeping track of such windows, e.g. the time the water level exceeds a critical value, one eliminates the need to account for the possibility of critical changes occurring every model time step. As long as no window opens up, the model can keep running using the more efficient growth-related timeframe.

For idealized biogeomorphic models, it is feasible to perform a theoretical stability analysis (Bärenbold et al., 2016). For more complex, process-based models such as Temmerman et al. (2007) and Van Oorschot et al. (2015) there is no such mathematical quantification

and choosing correct timescales for the problem at hand is a matter of expert judgment. Fagherazzi *et al.* (2012) provide an overview of approaches used in numerical models for salt marsh development. Schwarz *et al.* (2018) define a colonization dominance index (CDI), which is the ratio between lateral expansion rate and expansion probability. Although this CDI does not provide a quantitative stability criterion like the Courant-Friedrichs-Lewy number does for hydrodynamic calculations, it does provide a conceptual framework (Figure 6.4) for the emergence of self-organisation, hence the need for a strong coupling of ecology and morphodynamics, in biogeomorphological systems.

6.3 Forcings: calm conditions vs. stroms and floods

Unlike 'classical' morphological simulations where high energy events (floods, storms) can be driving most change in the system (Lesser *et al.* (2004), Roelvink and Reniers (2011)), biogeomorphological simulations need to account for long-lasting calm conditions too because the biological development during these periods can be considerable (examples in Bouma *et al.* (2016); Corenblit *et al.* (2007)). Growth, both in terms of individual plant size and patch size through lateral clonal expansion, is the dominant development process in calm conditions. Note that this growth can also be negative, i.e. decay, due to sub-optimal living conditions or seasonal dynamics (light, temperature) that are independent of geomorphological forcings. High energy events typically lead to destruction or removal of organisms due to excessive forces or lack of oxygen as a consequence of high water levels. This removal can offer opportunities for other organisms to establish. Therefore, transition periods such as the waning stage of a flood wave are often crucial for establishment, despite their short duration. As such, the inclusion of biota introduces an additional time scale to the geomorphic model, leading to a stiffer model, i.e. with less flexibility to scale time.
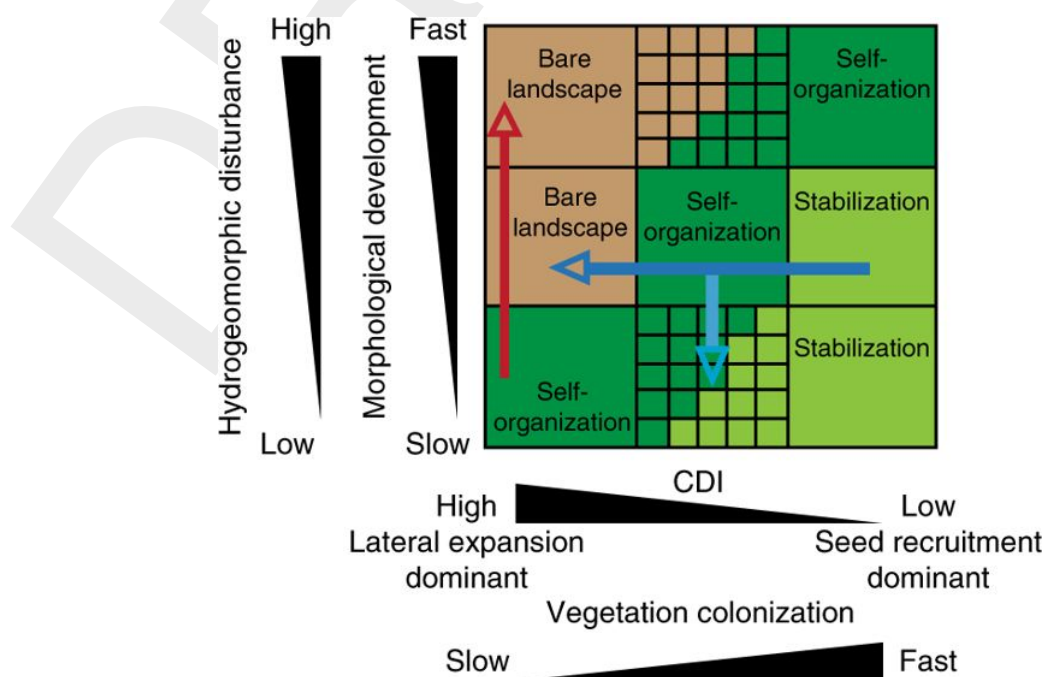
Figure 6.4: *Conceptual model illustrating the need for a strong (frequent) coupling of models for ecological (horizontal axis) and physical (vertical axis) development in self-organizing systems. The arrows indicate possible changes over time, e.g. the red arrow displays the transition resulting from low colonization rates combined with increasing morphological development. (from: Schwarz et al., 2018)*

To illustrate the possible importance of including events and biological responses, Figure 6.5 shows results of two numerical simulations by Akerboom (2018) for the Rødsand lagoon; one using actual weather conditions, the other with a fictitious severe storm in spring. The effect of this storm on biomass persists throughout the year.
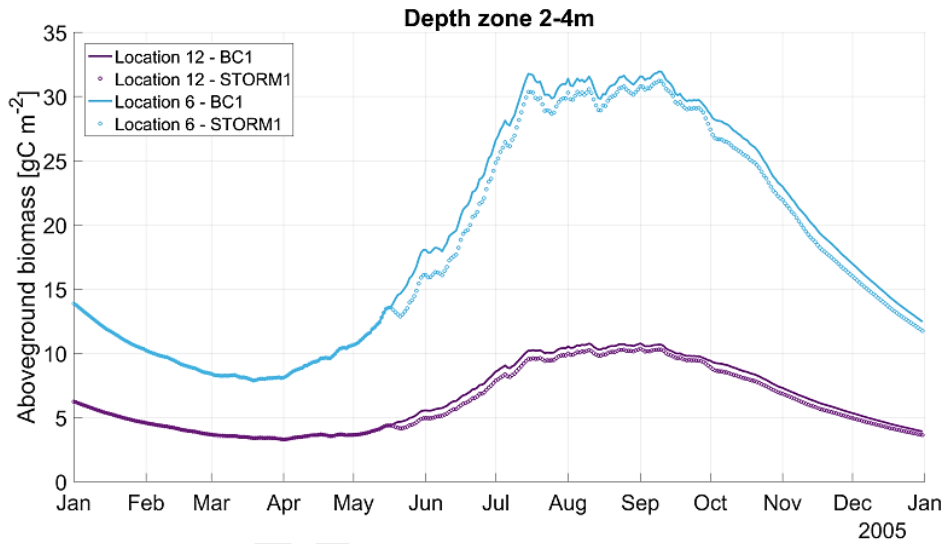
Figure 6.5: *Simulation of eelgrass development over a year in the Rødsand lagoon, show-ing the combined effect of seasonal variations in growth conditions, day-to-day variability in weather forcing and the long-lasting negative effect of a storm half May (dotted lines). (from: Akerboom (2018)).*
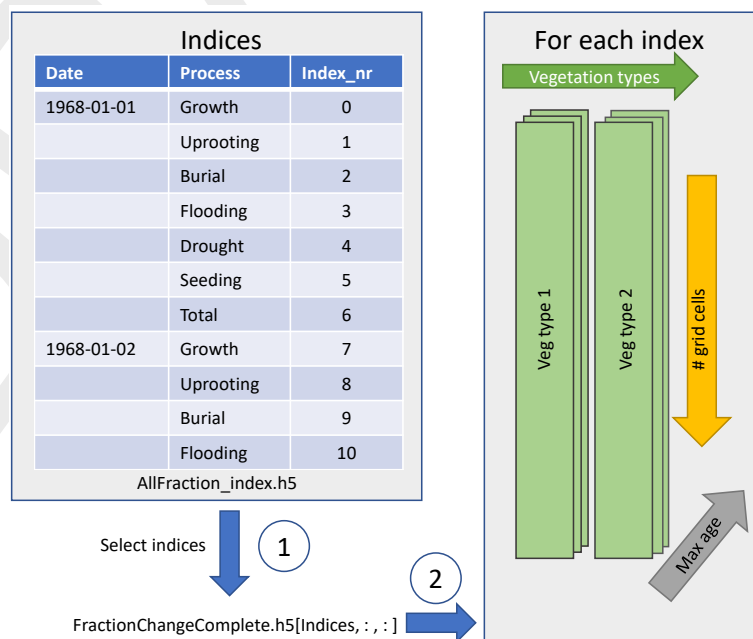
7 Post-processing and Data Storage

...

7.1 Data storage

The vegetation parameters (presence, size, dimensions) as used by the hydrodynamic model are stored along with the hydro- and morphodynamic parameters in the $<*$.nc$>$-file of the simulation. Other parameters describing the vegetation have to be stored by the user, i.e. exported from Python.

7.2 Data storage Riparian Vegetation Module

The vegetation model writes output in $<*$.h5$>$-format. The main vegetation data is stored in the file $<$FractionChangeComplete$>$. This file contains data of the fraction of vegetation in each grid-cell, for each time step, for each of the vegetation types after each of the processes. It needs the index file $<$AllFraction_index.h5$>$ to select the data for extraction (Figure 7.1). The data is stored after each process is completed. First a selection of the indices should be made for which to extract the data. For each selected index, fraction data can be extracted for all vegetation types, for all grid cells and all vegetation ages.

Result = [Index_nr, vegetation types, grid-cells, maximum age]

Figure 7.1: Structure for data storage and extraction.

7.3 Post-processing

The results of the vegetation module can be post-processed to visualize and inspect the results. There is a script available that provides basic inspection of the results. It allows you to extract and visualize one time step and one process per year. The figures that can be created with the <VegModulePostprocessing∗∗.py> script are:

◇ Line plot of vegetation cover over time expressed as a percentage of cover per year in the whole model grid for all ages for each vegetation type and for the sum of all vegetation types
◇ Line plot of vegetation cover per vegetation type per life stage
◇ stacked area plot of vegetation cover per vegetation type stacked over life stages
◇ contour plot with bed level and maximum vegetation fraction in grid cell over all ages and all vegetation types. For this plot a specific model time step has to be selected.
◇ contour plot with bed level and maximum vegetation age in grid cell over all vegetation types. For this plot a specific model time step has to be selected.
◇ contour plot with bed level and vegetation type in grid cell. A relative fraction threshold can be set above which the vegetation type is assumed to be dominant. For this plot a specific model time step has to be selected.

Several Python libraries have to be installed first.

◇ `conda install -c anaconda seaborn`
◇ `conda install -c anaconda matplotlib`
◇ `conda install -c anaconda h5py`
◇ `conda install -c anaconda netcdf4`

In addition, the visualisation script needs the functions in the <VisFunctions.py> script.

References

Akerboom, L. F. D., 2018. *Modelling Climate Change-Related Feedbacks between Seagrasses and Suspended Sediment in the Rødsand Lagoon (Denmark)*. Master's thesis, TU Delft.

Balke, T., T. J. Bouma, E. M. Horstman, E. L. Webb, P. L. A. Erftemeijer and P. M. J. Herman, 2011. "Windows of Opportunity: Thresholds to Mangrove Seedling Establishment on Tidal Flats." *Inter-Research Science Publisher* DOI: https://doi.org/10.3354/meps09364.

Balke, T., M. Stock, K. Jensen, T. J. Bouma and M. Kleyer, 2016. "A global analysis of the seaward salt marsh extent: The importance of tidal range." *Water Resources Research* .

Baptist, M., 2005. *Modelling Floodplain Biogeomorphology*. Ph.D. thesis, TU Delft.

Baptist, M., V. Babovic, J. Rodriguez Uthurburu, R. Uittenbogaard, A. Mynett and A. Verwey, 2007. "On inducing equations for vegetation resistance." *Journal of Hydraulic Research* 45 (4): 435–450. DOI: 10.1080/00221686.2009.9521996, ISBN 0022-1686, ISSN 0022-1686.

Bärenbold, F., B. Crouzy and P. Perona, 2016. "Stability analysis of ecomorphodynamic equations." *Water Resources Research* pages n/a–n/a. DOI: 10.1002/2015WR017492, ISSN 00431397, URL http://doi.wiley.com/10.1002/2015WR017492.

Berger, U. and H. Hildenbrandt, 2000. "A new approach to spatially explicit modelling of forest dynamics: Spacing, ageing and neighbourhood competition of mangrove trees." *Ecological Modelling* 132 (3): 287–302. DOI: 10.1016/S0304-3800(00)00298-2, ISSN 03043800.

Bouma, T. J., J. van Belzen, T. Balke, J. van Dalen, P. Klaassen, A. M. Hartog and P. M. J. Herman, 2015. "Windows of opportunity for salt marsh vegetation establishment on bare tidal flats: The importance of temporal and spatial variability in hydrodynamic forcing." *Journal of Geophysical Research: Biogeosciences* .

Bouma, T. J., J. van Belzen, T. Balke, J. van Dalen, P. Klaassen, A. M. Hartog and P. M. J. Herman, 2016. "Short-term mudflat dynamics drive long-term cyclic salt marsh dynamics." *Limnology and Oceanography* .

Brückner, M. Z., C. Schwarz, W. M. van Dijk, M. van Oorschot, H. Douma and M. G. Kleinhans, 2019. "Salt Marsh Establishment and Eco-Engineering Effects in Dynamic Estuaries Determined by Species Growth and Mortality." *Journal of Geophysical Research: Earth Surface* 124 (12): 2962–2986. DOI: 10.1029/2019JF005092, ISSN 21699011.

Carr, J., P. D'odorico, K. McGlathery and P. L. Wiberg, 2010. "Stability and bistability of seagrass ecosystems in shallow coastal lagoons: role of feedbacks with sediment resuspension and light attenuation." *Journal of Geophysical Research: Biogeosciences* 115 (G3).

Corenblit, D., E. Tabacchi, J. Steiger and A. M. Gurnell, 2007. "Reciprocal interactions and adjustments between fluvial landforms and vegetation dynamics in river corridors: A review of complementary approaches." *Earth-Science Reviews* 84 (1-2): 56–86. DOI: 10.1016/j.earscirev.2007.05.004, ISSN 00128252.

Fagherazzi, S., M. L. Kirwan, S. M. Mudd, G. R. Guntenspergen, S. Temmerman, A. D'Alpaos, J. van de Koppel, J. M. Rybczyk, E. Reyes, C. Craft and J. Clough, 2012. "Numerical Models of Salt Marsh Evolution: Ecological, Geomorphic and Climatic Factors." *Reviews of Geophysics* 50.

Grueters, U., T. Seltmann, H. Schmidt, H. Horn, A. Pranchai, A. G. Vovides, R. Peters, J. Vogt, F. Dahdouh-Guebas and U. Berger, 2014. "The mangrove forest dynamics model meso-FON." *Ecological Modelling* 291: 28–41. DOI: 10.1016/j.ecolmodel.2014.07.014, ISSN 03043800.

Kleinhans, M. G., B. de Vries, L. Braat and M. van Oorschot, 2018. "Living landscapes: Muddy and vegetated floodplain effects on fluvial pattern in an incised river." *Earth Surface Processes and Landforms* 43 (14): 2948–2963. DOI: 10.1002/esp.4437, ISSN 10969837.

Lesser, G., J. Roelvink, J. van Kester and G. Stelling, 2004. "Development and validation of a three-dimensional morphological model." *Coastal Engineering* 51 (8-9): 883–915. DOI: 10.1016/j.coastaleng.2004.07.014, ISSN 03783839, URL http://linkinghub.elsevier.com/retrieve/pii/S0378383904000870.

Levin, S. A., 1987. "Scale and Predictability in Ecological Modeling." In S. J. Vincent T.L., Cohen Y., Grantham W.J., Kirkwood G.P., ed., *Modeling and Management of Resources under Uncertainty. Lecture Notes in Biomathematics, vol 72*, pages 2–10. Springer, Berlin, Heidelberg.

Lokhorst, I. R., L. Braat, J. R. Leuven, A. W. Baar, M. Van Oorschot, S. Selaković and M. G. Kleinhans, 2018. "Morphological effects of vegetation on the tidal-fluvial transition in Holocene estuaries." *Earth Surface Dynamics* 6 (4): 883–901. DOI: 10.5194/esurf-6-883-2018, ISSN 2196632X.

Maanen, B. van, G. Coco and K. R. Bryan, 2015. "On the ecogeomorphological feedbacks that control tidal channel network evolution in a sandy mangrove setting." *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Science* 471: 20150115. DOI: 10.1098/rspa.2015.0115, ISSN 1364-5021, URL http://rspa.royalsocietypublishing.org/content/471/2180/20150115.

Martínez-Fernández, V., M. Van Oorschot, J. De Smit, M. González del Tánago and A. Buijse, 2018. "Modelling feedbacks between geomorphological and riparian vegetation responses under climate change in a Mediterranean context." *Earth Surface Processes and Landforms* DOI: 10.1002/esp.4356, ISSN 10969837.

Odink, S., W. P.W.J.M., B. Smits, B. Borsje and S. Hulscher, 2019. "Long-term marsh growth and retreat in an online coupled hydrodynamic, morphodynamic and ecological model."

Oorschot, M. van, 2017. *Riparian vegetation interacting with river morphology: modelling long-term ecosystem responses to invasive species, climate change, dams and river restoration*. Ph.D. thesis, Utrecht University.

Oorschot, M. van, M. Kleinhans, T. Buijse, G. Geerling and H. Middelkoop, 2018. "Combined effects of climate change and dam construction on riverine ecosystems." *Ecological Engineering* 120 (May): 329–344. DOI: 10.1016/j.ecoleng.2018.05.037, ISSN 09258574, URL https://doi.org/10.1016/j.ecoleng.2018.05.037.

Oorschot, M. van, M. Kleinhans, G. Geerling, G. Egger, R. Leuven and H. Middelkoop, 2017. "Modeling invasive alien plant species in river systems: Interaction with native ecosystem engineers and effects on hydro-morphodynamic processes." *Water Resources Research* 53 (8). DOI: 10.1002/2017WR020854, ISSN 19447973.

Peckham, S., E. Hutton and B. Norris, 2013. "A component-based approach to integrated modeling in the geosciences: The design of CSDMS." *Computers and Geosciences* .

Phillips, J. D., 1995. "Biogeomorphology and landscape evolution: The problem of scale." *Geomorphology* 13 (1-4): 337–347. DOI: 10.1016/0169-555X(95)00023-X, ISSN 0169-555X, URL https://www.sciencedirect.com/science/article/abs/pii/0169555X9500023X.

Ranasinghe, R., C. Swinkels, A. Luijendijk, J. Bosboom, D. Roelvink, M. J. F. Stive and D. J. Walstra, 2010. "Morphodynamic upscaling with the Morfac approach." In *Coastal Engineering 2010*, pages 1–7.

Roelvink, J. A., 2006. "Coastal morphodynamics evolution techniques." *Coastal Engineering* 53: 277-287.

Roelvink, J. A. and A. J. H. M. Reniers, 2011. "A guide to coastal morphology modeling." *Advances in Coastal and Ocean Engineering* 12: 3-21.

Schwarz, C., O. Gourgue, J. van Belzen, Z. Zhu, T. J. Bouma, J. van de Koppel, G. Ruessink, N. Claude and S. Temmerman, 2018. "Self-Organization of a Biogeomorphic Landscape Controlled by Plant Life-History Traits." *Nature Geoscience* .

Silinski, A., J. van Belzen, E. Fransen, T. J. Bouma, P. Troch, P. Meire and S. Temmerman, 2016. "Quantifying critical conditions for seaward expansion of tidal marshes: A transplantation experiment." *Estuarine, Coastal and Shelf Science* .

Soetaert, K. and P. M. Herman, 2009. *A practical guide to ecological modelling: using R as a simulation platform*, vol. 7. Springer, Berlin.

Temmerman, S., T. J. Bouma, J. Van de Koppel, D. Van der Wal, M. B. De Vries and P. M. J. Herman, 2007. "Vegetation Causes Channel Erosion in a Tidal Landscape." *Geology* v. 35: 631-634.

Van Oorschot, M., M. G. Kleinhans, G. W. Geerling, H. Middelkoop, E. Mosselman and A. D. Buijse, 2015. "Distinct Patterns of Interactions between Vegetation and River Morphology." In *REFORM International Conference on River and Stream Restoration, Wageningen, The Netherlands*.

Willemsen, P. W. J. M., B. W. Borsje, S. J. M. H. Hulscher, D. Van der Wal, Z. Zhu, B. Oteman and T. J. Bouma, 2018. "Quantifying Bed Level Change at the Transition of Tidal Flat and Salt Marsh: Can We Understand the Lateral Location of the Marsh Edge?" *Journal of Geophysical Research: Earth Surface* .

Willemsen, P. W. J. M., B. Smits, B. W. Borsje, P. Herman, J. Dijkstra, T. J. Bouma and S. J. M. H. Hulscher, 2022. "Modeling decadal salt marsh development: Variability of the salt marsh edge under influence of waves and sediment availability." *Water Resources Research* .

A Installation Guide

In this appendix ...

B BMI parameters

Model geometry parameters (i.e. parameters without operations in Python):

◇ ndx = model.get_var ('ndx')   number of boxes, including boundary boxes
◇ ndxi = model.get_var('ndxi')   number of non-boundary, i.e. within-domain boxes
◇ xzw = model.get_var('xzw')   x coord. of the center of gravity of the boxes
◇ yzw = model.get_var('yzw')   y coord. of the center of gravity of the boxes
◇ lnx = model.get_var('lnx')   total number of links between boxes
◇ lnxi = model.get_var('lnxi')   number of links between within-domain boxes
◇ ln = model.get_var('ln')   link matrix between adjacent boxes [ln,2] matrix
◇ dx = model.get_var('dx')   distance between the centers of adjacent boxes
◇ wu = model.get_var('wu')   width of the interface between adjacent boxes
◇ ba = model.get_var('ba')   surface area of the boxes

Physical vegetation variables:

◇ rnveg = model_dfm.get_var('rnveg')   [1/m2] 3D plant density, 2D part is basis input (1/m2)
◇ diaveg = model_dfm.get_var('diaveg')   [m] 3D plant diameter, 2D part is basis input (m)
◇ stemheight = model_dfm.get_var('stemheight')   [m] 2D plant heights (m)
◇ tau_cre = model_dfm.get_var('TcrEro')   [N m-2] critical bed shear stress for erosion

Statistical variables in D-Flow FM/Python (i.e. fixed variables without operations in Python):

◇ is_dtint = model_dfm.get_var('is_dtint')   number of Delft3D Flexible Mesh timesteps executed over a vegetation timestep
◇ is_sumvalsnd = model_dfm.get_var('is_sumvalsnd')   sum of (1) bed shear stress, (2) flow velocity, (3) water depth per grid cell over vegetation timestep
◇ is_maxvalsnd = model_dfm.get_var('is_maxvalsnd')   maximum of (1) bed shear stress, (2) flow velocity, (3) water depth per grid cell over vegetation timestep

Hydrodynamic parameters in D-Flow FM, showing their current status (i.e. parameters without operations in Python):

◇ hs = model_dfm.get_var('hs')   [m] waterdepth at cell centre = s1 - bl
◇ s0 = model_dfm.get_var('s0')   [m] waterlevel at start of timestep
◇ s1 = model_dfm.get_var('s1')   [m] waterlevel at end of timestep
◇ ucx = model_dfm.get_var('ucx')   [m s-1] cell center velocity, global x-dir
◇ ucy = model_dfm.get_var('ucy')   [m s-1] cell center velocity, global y-dir
◇ ucz = model_dfm.get_var('ucz')   [m s-1] cell center velocity, global z-dir
◇ ucmag = model_dfm.get_var('ucmag')   [m s-1] cell center velocity magnitude
◇ taus = model_dfm.get_var('taus')   [kg m s-2] cell centre tau N m-2
◇ hwav = model_dfm.get_var('hwav')   [m] root mean square wave height
◇ twav = model_dfm.get_var('twav')   [m] wave period
◇ Uorb = model_dfm.get_var('Uorb')   [m s-1] orbital velocity

Morphodynamic parameters in D-Flow FM, showing their current status (i.e. parameters without operations in Python):

◇ bl = model_dfm.get_var('bl')   [m] bottom level (positive upward)
◇ bodsed = model_dfm.get_var('bodsed')   [kg m-2] Available sediment in the bed in flow cell center
◇ dpsed = model_dfm.get_var('dpsed')   [m] Sediment thickness in the bed in flow cell cen-

ter
◇ msed = model_dfm.get_var('msed')   [kg m-2] Available sediment in a layer of the bed in flow cell center
◇ thlyr = model_dfm.get_var('thlyr')   [m] Thickness of a layer of the bed in flow cell center
◇ sbcx = model_dfm.get_var('sbcx')   [kg s-1 m-1] bed load transport due to currents, x-component
◇ sbcy = model_dfm.get_var('sbcy')   [kg s-1 m-1] bed load transport due to currents, y-component
◇ sbwx = model_dfm.get_var('sbwx')   [kg s-1 m-1] bed load transport due to waves, x-component
◇ sbwy = model_dfm.get_var('sbwy')   [kg s-1 m-1] bed load transport due to waves, y-component
◇ sscx = model_dfm.get_var('sscy')   [kg s-1 m-1] suspended load transport due to currents, x-component
◇ sscy = model_dfm.get_var('sscy')   [kg s-1 m-1] suspended load transport due to currents, y-component
◇ sswx = model_dfm.get_var('sswx')   [kg s-1 m-1] suspended load transport due to waves, x-component
◇ sswy = model_dfm.get_var('sswy')   [kg s-1 m-1] suspended load transport due to waves, y-component

Water quality parameters in DFM, showing their current status (i.e. parameters without operations in Python):

◇ sa0 = model_dfm.get_var('sa0')   [1e-3] salinity (ppt) at start of timestep
◇ sa1 = model_dfm.get_var('sa1')   [1e-3] salinity (ppt) at end of timestep
◇ tem0 = model_dfm.get_var('tem0')   [degC] water temperature at start of timestep
◇ tem1 = model_dfm.get_var('tem1')   [degC] water temperature at end of timestep
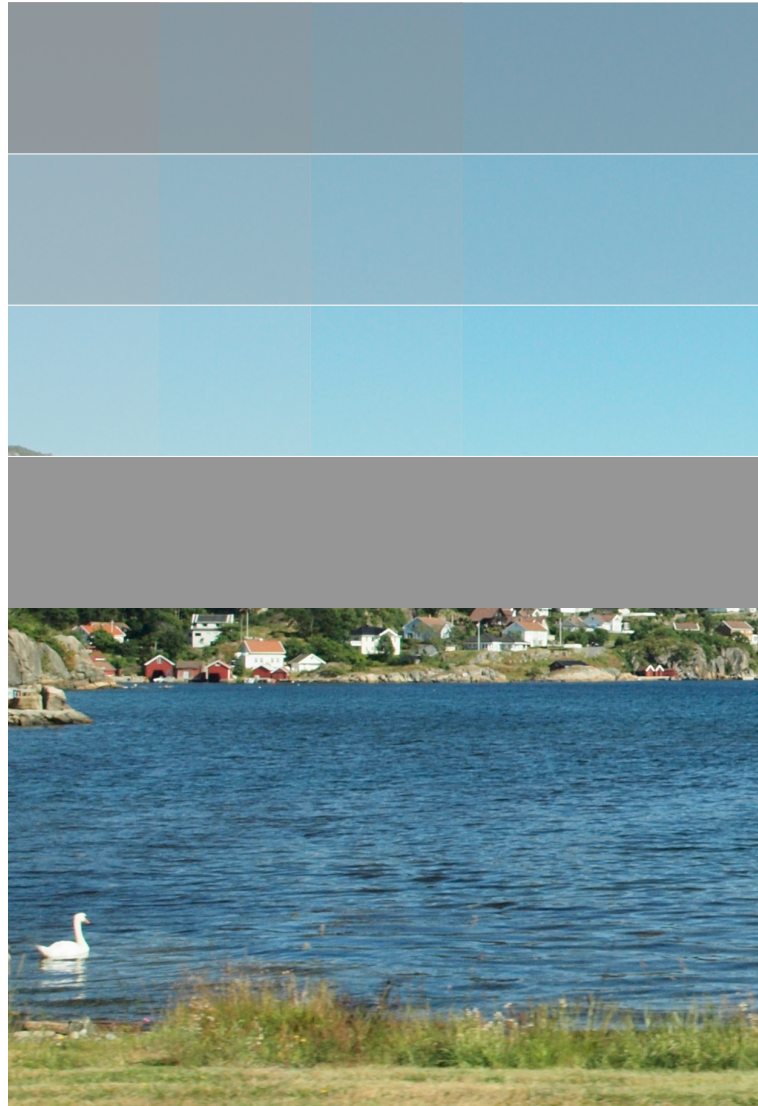
es systems

+31 (0)88 335 8188
software@deltares.nl
www.deltares.nl/software