APPENDIX

# A Genetic Algorithm to Solve the Optimization Problem (I.1)

In this section, we describe in more detail how the dual genetic algorithm presented in the introduction works.

## Population Creation

First, a population *ψi*=0,...,*Npop* of *Npop* individuals is created (Figure **I.2.1)**). Initially, all these individuals are identical to the mother individual *ψM*. For each of these indi- viduals, a random number of "bumps" (represented by figure [**A.1**](#_bookmark0)) is added to *ψM*. The parameters of these "bumps" (number, amplitude *A*, position, width *L*) are determined randomly: this ensures diversity in the initial population, as illustrated in figure **I.2.1)**.

1

*A*/2 [1 − cos(2*πx*/*L*)]

0.8

0.6

y

0.4

0.2

0

0 0.2 0.4 0.6 0.8 1

x

**Figure A.1 –** Illustration of a sinusoidal half-wrap of amplitude *A* = 1 and length *L* = 1.

The choice of representing our population with sinusoidal evolutions comes from

Fourier who assures that it’s possible to represent any function with an infinite number of sinusoids.

## Selection

For each generation (and therefore iteration), 2 individuals *ψq*, *ψp* (with *q*, *p* ∈ [0, ..., *Npop*]) are randomly selected. These two individuals from the population *Npop* are then compared in the form of a "duel". In the case of minimization, the individual with the highest J cost function is mutated. The other individual is retained. Here, the

cost function J = 1 *ρwg* ∫Ω *H*2*d*Ω is the wave energy integrated over the whole domain.

16

This selection is represented by Figure **I.2.2)** and is algorithmically translated by the

following pseudo-code [**2**](#_bookmark1).

**Algorithm 2** Selection and mutation

**Input:** *Npop* the number of individuals in the population, *ψi*=0,...,*Npop* 0 the population, randint the function returning a random integer, calc\_J the calculation of the cost function, mutation the function performing a mutation.

J

**Output:** The new *ψi* population with a mutation.

1: *p* randint(0, *Npop*) ▷ Random selection of an individual from the population.

←

2: *q* randint(0, *Npop*)

←

3: *p* calc\_J(*ψp*) ▷ Calculation of cost function

J ←

4: *q* calc\_J(*ψq*)

J ←

5: **if** *p* > *q* **then**

J J

6: *p* mutation( *q*) ▷ Cloning + Mutation

J ← J

7: **else**

8: *q* mutation( *p*)

J ← J

9: **end if**

## Mutation

As previously indicated in the [**2**](#_bookmark1)algorithm, the individual undergoing mutation be- comes an evolution of the individual with the lowest J cost function. Mutation acts in a very similar way to population creation. One or more of the individual’s bumps will be selected. They will then be mutated by changing their parameters (number, amplitude *A*, position, width *L*). They can thus be made to move slightly, have a larger (or smaller) amplitude and a slightly different length. Unlike the creation of the population, the pa- rameters are not entirely regenerated; the old parameters evolve by a small amount *ε* in order to ensure the convergence of the population. This small *ε* shift can be seen on the

**I.2.3)** where there is not much variability between the two individuals (the old and the

mutated). Moreover, the mutated individual has a slightly lower cost function than the original individual.

## Final Population

By cleverly choosing a convergence criterion, we decide to stop mutations within the population. If the criterion is well chosen, we should obtain a population that has entirely converged at a single point, as shown in figure **I.2.4)**. In our case, the algorithm stops when J hardly evolves over a certain number of iterations. This gives us the convergence curve figure [**A.2**](#_bookmark2).

Evolution of *J* during the duels

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

900

800

Min *J* of the population

700

600

500

400

300

0 100 200 300 400

Number of duels

**Figure A.2 –** Evolution of J during duels.

Note that the population tends towards a min value. Then, to select the best candi- date, we simply select the candidate with the lowest cost function.

# A few Attempts to Improve the Model

Throughout the thesis, we have considered a single J functional, the wave energy integral. This choice was made deliberately, as we feel that this functional is the most realistic and produces the most realistic results. However, a multitude of different func- tionals and approaches have been tested in order to improve our model. In this section, we present some of the non-concurrent tests that have been carried out to improve our model. We have essentially focused on two cases: a case with linear bathymetry and an experimental case.

## Reference Cases

In order to test our new functional as well as possible, we will carry out numerous sim- ulations based essentially on the two following test cases. These two cases were presented in Chapter 1, so we know where their limits lie and what results we can expect.

### Case 1: Simulation of a One Week Storm on a Linear Beach

This first benchmark simulation is presented in Chapter 3 of Cook (2021). This simulation is described as highly morphogenic in that it simulates a storm over a few days with the following parameters:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Physics | Simulation parameters | | | Hydrodynamic | | Morphodynamic | | Domain | | |
| Parameters | ∆*x* | ∆*t* | *Tf* | *Hmax* | *T*0 | Υ | *Mslope* | *L* | *h*0 | slope *α* |
| Values | 1 *m* | 400 *s* | 1 week | 2 *m* | 2 *s* | 4.25e-5 *m*.*s*.*kg*−1 | 20% | 600 *m* | 7 *m* | 11% |

**Table B.1 –** Parameter of the storm simulation.

which are, the spatial step ∆*x* , the time step ∆*t*, the duration of the simulation *Tf* , the maximum water height in forcing *Hmax*, the wave period *T*0, the sand abrasion Υ, the maximum sand slope *Mslope*, the length of the simulation domain *L*, the closing depth *h*0. This is represented by the following forcing and domain figure [**B.1**](#_bookmark4).

Simulation of storm with forcings and géometrie



|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  | Initial Seab  Final Seabe | ed 0  d *f* |
|  |  |  |  |  |  | *h*0 |  |
|  |  |  |  |  |  |  |  |

6

Seabed [m]



4

2

0

0 100 200 300 400 500 600

Distance from deep sea [m]

2.0

*max*

*H*

0(*t*)

*H*

Wave Height [m]

1.5

1.0

0.5

0.00 0.25 0.50 0.75 1.00 1.25 1.50 1.75

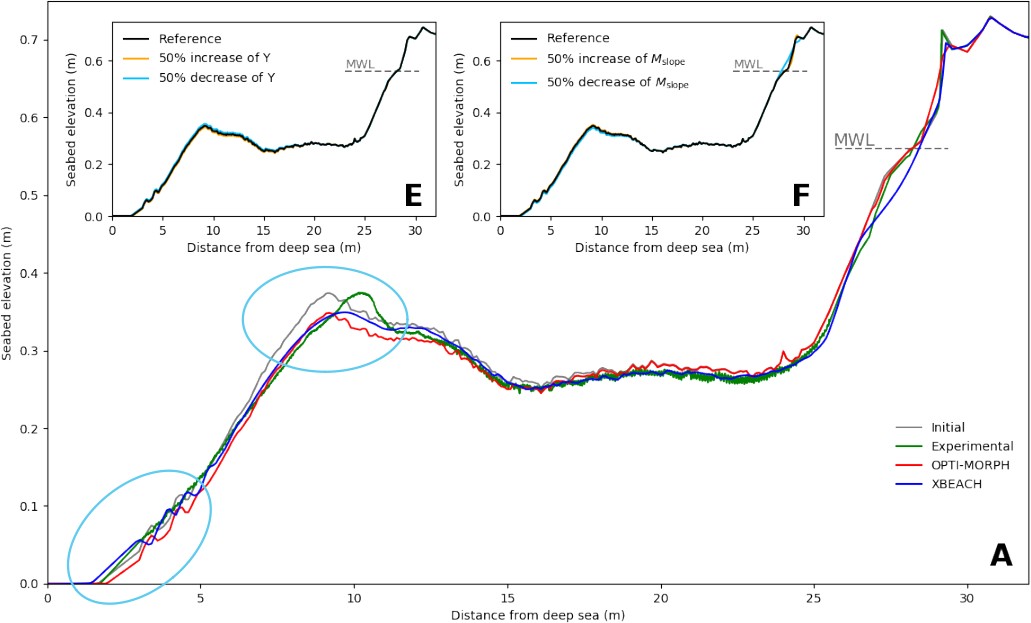
Time [s] 1e6

**Figure B.1 –** Forcing for the one week storm simulation.

The result of this simulation shows the formation of a realistic sand bar. We use this simulation to observe how the model behaves when the physics are changed.

### Case 2: Simulation of a Flume Experiment: COPTER

The second reference simulation is present in the article by Cook (2021). This reference case, called COPTER compares simulations with real data from basin tests. Hydro- morphodynamic simulations are carried out using our OptiMorph model and the well know XBeach model. The results of these simulations are shown in Figure [**B.2**](#_bookmark6).



**Figure B.2 –** Weaknesses (circled in turquoise) of the Copter simulation at the morphodynamic level.

Analysis of these simulations revealed a number of major areas for improvement. These points have been circled in turquoise on the Figure [**B.2**](#_bookmark6). 1) The movement of sand up- stream of the simulation. The sandy movements at this point are too great with the OptiMorph model. 2) The sedimentary bar is supposed to be moving to the coast. How- ever, it is sagging.

For the first point, it is possible that the experimental surveys present too many uncer- tainties at this level. Indeed, it is very difficult to perceive precisely the morphodynamic displacements at the level of the beater. The second point leads us to believe that some-

thing is missing in our model, such as a current that would allow us to move this bar that we have. The purpose of our functional approach will be to try to physically represent this current or other quantities.

## Improvement by Functional Approach

The first idea that comes to mind improving the model in a rather naive way would be to change what governs the morphodynamics, namely the cost function J . A very large number of functional J was tested in order to take into account more physics. In the work of Cook et al. (2021c), a number of functional has been tested, as shown in the table [**B.2**](#_bookmark7).

|  |  |  |
| --- | --- | --- |
| Keyword | Definition | Commentary |
| CF0 | *d* = −*x*∇*ψ*ES*χ*ΩS | Recommended |
| CF1 | *d* = − B ∇*ψ*ES*χ*ΩS  *x*S |  |
| CF2 | *d* = − *x*2 ∇ E *χ*  B  *x*S *ψ* S ΩS |  |
| CF3 | *d* = −*x x*B∇*ψ*ES*χ*ΩS  ∫ |  |
| CF4 | *d* = − B ∇*ψ*ES*χ*Ω  *x*S ΩS S |  |
| CF5 | *d* = (1 − Λ)CF2 + ΛCF4 | where Λ is the excitation of the seabed |
| CF6 | *d* = (1 − Λ)CF3 + ΛCF4 | where Λ is the excitation of the seabed |

**Table B.2 –** Old cost functions J .

The results of these tests showed that the most physical and relevant functional was the wave energy functional J = EH. This one showed the most relevant results present in (Cook 2021).

In a similar way, we tried out a large number of functional. Some of the functional we tested are shown in the table [**B.3**](#_bookmark8).

|  |  |  |
| --- | --- | --- |
| Keyword | Définition | Commentaire |
| CF8 | *d* = −∇*ψ* ES − *ερU*2 *b χ*ΩS  *or* | Kinetic energy removal |
| CF9 | *d* = −∇*ψ ρU*2 *b χ*ΩS  *or* | Kinetic energy |
| CF10 | *d* = −∇*ψ εCg H*2 *χ*ΩS |  |
| CF11 | *d* = −∇*ψ Cg H χ*ΩS |  |
| CF12 | *d* = −∇*ψ ε*(*Cg H*)2 *χ*ΩS |  |
| CF13 | *d* = −∇*ψ* (*Sxx*) *χ*ΩS | Radiation stress |
| CF14 | *d* = −∇*ψ* (∇*Sxx*) *χ*ΩS  1 ∫ ∫ | Gradient of adiation stress |
| CF15 | *J* = 8 *ρwg* Ω *H*2*dx* + *ρsg* Ω (*ψ*(*t*) − *ψ*0(*τ* − *t*))2*dx*  *S S* | Sand displacement memory |

**Table B.3 –** Nouvelles fonctions de coût J .

these functions have been found by relying on basic physics: calculation of forces, work, kinetic energy, ... but also by relying on the balance of moments (Sous et al. 2020):

*∂*  *ρU*2 = −*gρ∂η*¯ − 1 *∂Sxx* − 1 *τ*¯*b* h*Pa* · *m*−1i , (B.1)

*∂x*

*∂x*

(*η*¯ + *h*)

*∂x*

(*η*¯ + *h*)

where *U* is the depth-averaged velocity, *g* the gravitational acceleration, *ρ* the water density, *η*¯ the wave setup, *h* the still water depth, *Sxx* the radiation stress, and *τb* the bed shear stress.

### Functional with Kinetic Energy Dissipation (CF8)

The first functional aims at introducing a term taking into account the kinetics of a wave. Indeed, if we take into account the kinetics of a wave, it could be that it could artificially simulate a "current" which would allow us to obtain a displacement of the bar.

= 1

∫J

16

(*ρgH*2 *ερU*2

Ω*s*

*orb*

−

)*dx* (B.2)

with *ε* in *m*−1 that we will fix for the moment arbitrarily. In order to implement this functional, it is necessary to differentiate it according to *ψ* (calculation of ∇*ψ*J ). The term JH has already been differentiated **2.1**, it only remains to differentiate the term

∫Ω*s*

*ερU*

2

*orb*

*dx*. This is easily done via the following expression for the orbital velocity:

*Uorb* : Ω × [0, *h*0] −→ R+

(*x*, ) cosh (*k*(*x*) (*h*(*x*) − (*h*0 − *ψ*))). (B.3)

*ψ* ›−→

cosh(*k*(*x*)*h*(*x*))

We start again from the calculation of the gradient which allows us to obtain the new expression:

1

∇*ψ*J = 4 *ρ gH∂ψ H* + *ε∂ψUorb* (B.4)

with the only term here that we don’t know: *∂ψUorb*. With the equation ([**B.3**](#_bookmark11)), we obtain

:

*∂ψUorb* = *∂ψu v* − *u ∂ψv v*−2

with:

*u* = cosh (*k* (*h* − (*h*0 − *ψ*)))

*v* = cosh(*kh*)

*∂ψu* = sinh(*k* (*h* − (*h*0 − *ψ*)) (*h* + *ψ* − *h*0)*kψ*

*∂ψv* = sinh(*kh*)(*kψh* − *k*)

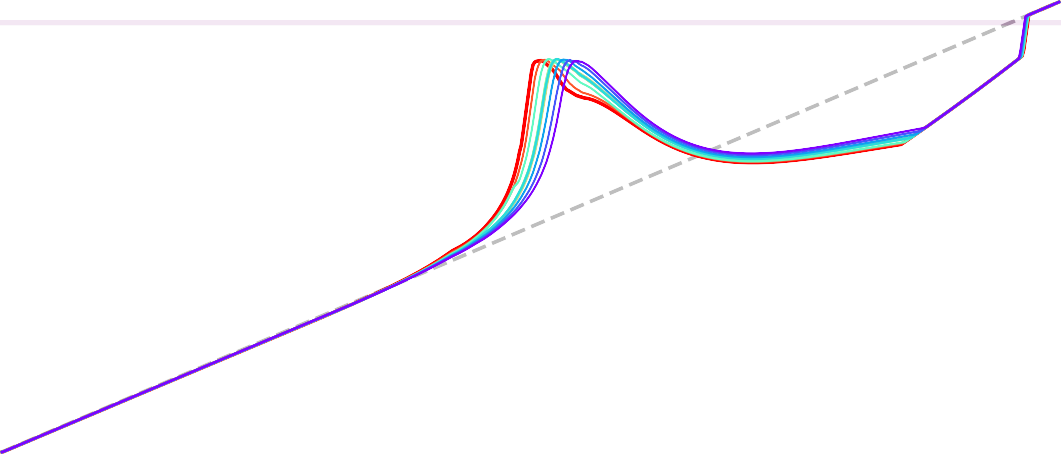
which is easy to calculate because we know the values of *hψ* and *kψ* which were specified in the part **2**.

We now try to run simulations on Case 1 ([**B.1.1**](#_bookmark3)) and Case 2 ([**B.1.2**](#_bookmark5)) with different values of *ε* from 0.001 to 0.009. The results are shown in figures [**B.3**](#_bookmark12) and [**B.4**](#_bookmark13).

Evolution of  at the end of the simulation for différents values of epsilon

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
|  | initial  *h*0 |  |  |  |  |  |  |
|  |
|  |
|  | *f* with epsilon=0 *f* with epsilon=0 | (CF0)  .001 |  |  |  |  |  |
|  |
|  |
|  | *f* with epsilon=0  *f* with epsilon=0 | .002  .003 |  |  |  |  |  |
|  |
|  | *f* with epsilon=0  *f* with epsilon=0  with epsilon=0 | .004  .005  .006 |  |  |  |  |  |
|  |
|  | *f*  *f* with epsilon=0 *f* with epsilon=0 | .007  .008 |  |  |  |  |  |
|  |
|  | *f* with epsilon=0 | .009 |  |  |  |  |  |
|  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

7



6

5

Seabed [m]

4

3

2

1

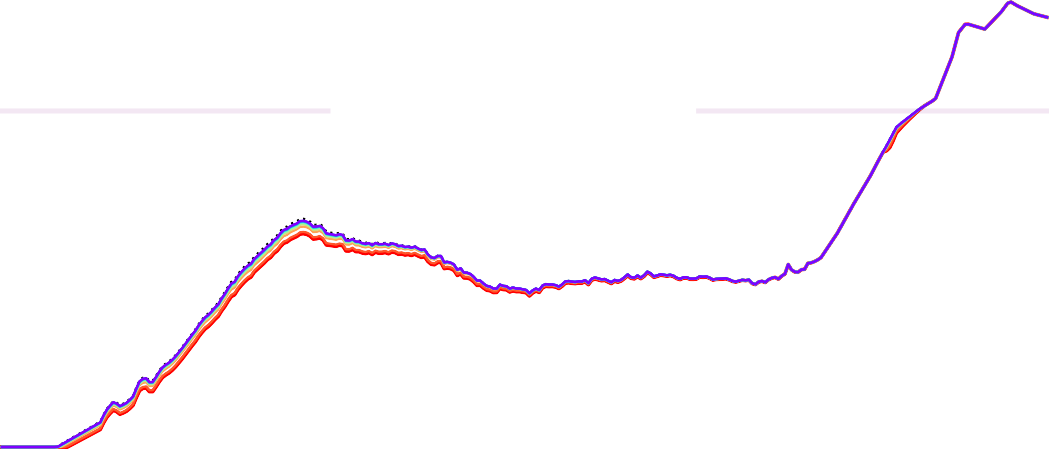
0

0 100 200 300 400 500 600

Distance from deep sea [m]

**Figure B.3 –** Results for different *ε* with the dissipation functional for simulation [**B.1.1**](#_bookmark3).

0.7



Evolution of at the end of the simulation for différents values of epsilon

0.36

0)

3

4

*f* with =0.005

*f* with =0.006

*f* with =0.007

*f* with =0.008

*f* with =0.009

0.30

80

85

90

95 100 105

*f* with =0.00

*f* with =0.00

0.32

0.34

1

2

*f* with =0.00

*f* with =0.00

*h*0

*f* with =0 (CF

0.38

initial

0.6

0.5

Seabed [m]

 0.4

0.3

0.2

0.1

0.0

0 50 100 150 200 250 300

Distance from deep sea [m]

**Figure B.4 –** Results for different *ε* with the dissipation functional for simulation [**B.1.2**](#_bookmark5).

In these cases, the CF0 functional is represented in red. Results from figure [**B.3**](#_bookmark12) suggest that it is possible to move the bar with this functional. However, looking at the figure

[**B.4**](#_bookmark13) we notice that there is no displacement of the sedimentary bar. In fact, the effect of attenuation is merely to attenuate sediment mobility. The conclusion on this functional from a numerical point of view is that it simply attenuates the descent in the gradient descent method. The results with *ε* of the first case are very similar to other results of *ψ* at 90-100% of the simulation time.

To understand what the model does with this functional, it may be interesting to look at the distribution of each term on the domain. That is to say, where does the term J*cinetique* acts and where does the term JH act. We display for the first case on figure

[**B.5**](#_bookmark14), the rapports *Jc* and *J*H at different moments of the simulation: at the beginning, at

*Jtot*

the peak and at the end.

*Jtot*

1.0

0.8

Rapport *JH*/*Jtot*

0.6

0.4

0.2

0.0

1.0

0.8

Rapport *JH*/*Jtot*

0.6

0.4

0.2

0.0

Evolution of *JH*/*Jtot* at the begening of the simulation for différents values of epsilon

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | |  |  |  |  |  |  |
|  |  | |  |  |  |  |  |  |
|  |  | |  |  |  |  |  |  |
|  |  | *JH*/*Jtot* wit *JH*/*Jtot* wit | h epsilon=0 h epsilon=0 | .004  .005 |  |  |  |  |
|  |  |
|  |  | *JH*/*Jtot* wit *JH*/*Jtot* wit | h epsilon=0  h epsilon=0 | .006  .007 |  |  |  |  |
|  |  |
|  |  | *JH*/*Jtot* wit *JH*/*Jtot* wit | h epsilon=0  h epsilon=0 | .008  .009 |  |  |  |  |
|  |  |
|  |  |
|  |  | |  |  |  |  |  |  |

0 100 200 300 400 500 600

Distance from deep to see [m]

Evolution of *JH*/*Jtot* at the peak of the simulation for différents values of epsilon

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | |  |  |  |  |  |  |
|  |  | |  |  |  |  |  |  |
|  |  | |  |  |  |  |  |  |
|  |  | *JH*/*Jtot* wit *JH*/*Jtot* wit | h epsilon=0 h epsilon=0 | .004  .005 |  |  |  |  |
|  |  |
|  |  | *JH*/*Jtot* wit *JH*/*Jtot* wit | h epsilon=0  h epsilon=0 | .006  .007 |  |  |  |  |
|  |  |
|  |  | *JH*/*Jtot* wit *JH*/*Jtot* wit | h epsilon=0  h epsilon=0 | .008  .009 |  |  |  |  |
|  |  |
|  |  |
|  |  | |  |  |  |  |  |  |

0 100 200 300 400 500 600

Distance from deep to see [m]

Evolution of *JH*/*Jtot* at the end of the simulation for différents values of epsilon

1.0

0.8

Rapport *Jc*/*Jtot*

0.6

0.4

0.2

0.0

1.0

0.8

Rapport *Jc*/*Jtot*

0.6

0.4

0.2

0.0

Evolution of *Jc*/*Jtot* at the begening of the simulation for différents values of epsilon

*JH*/*Jtot* wit *JH*/*Jtot* wit

*JH*/*Jtot* with epsilon=0.006 *JH*/*Jtot* with epsilon=0.007 *JH*/*Jtot* wit

*JH*/*Jtot* wit

.008

.009

h epsilon=0

h epsilon=0

.004

.005

h epsilon=0

h epsilon=0

0 100 200 300 400 500 600

Distance from deep to see [m]

Evolution of *Jc*/*Jtot* at the peak of the simulation for différents values of epsilon

*JH*/*Jtot* wit *JH*/*Jtot* wit

*JH*/*Jtot* with epsilon=0.006 *JH*/*Jtot* with epsilon=0.007 *JH*/*Jtot* wit

*JH*/*Jtot* wit

.008

.009

h epsilon=0

h epsilon=0

.004

.005

h epsilon=0

h epsilon=0

0 100 200 300 400 500 600

Distance from deep to see [m]

Evolution of *Jc*/*Jtot* at the end of the simulation for différents values of epsilon

1.0 1.0

*JH*/*Jtot* w *JH*/*Jtot* w

*JH*/*Jtot* with epsilon=0.006 *JH*/*Jtot* with epsilon=0.007 *JH*/*Jtot* w

*JH*/*Jtot* w

0.008

0.009

ith epsilon=

ith epsilon=

0.004

0.005

ith epsilon=

ith epsilon=

*JH*/*Jtot* wit *JH*/*Jtot* wit

*JH*/*Jtot* with epsilon=0.006 *JH*/*Jtot* with epsilon=0.007 *JH*/*Jtot* wit

*JH*/*Jtot* wit

.008

.009

h epsilon=0

h epsilon=0

.004

.005

h epsilon=0

h epsilon=0

0.8 0.8

0.6 0.6

Rapport *JH*/*Jtot*

Rapport *Jc*/*Jtot*

0.4 0.4

0.2 0.2

0.0

0 100 200 300 400 500 600

Distance from deep to see [m]

0.0

0 100 200 300 400 500 600

Distance from deep to see [m]

**Figure B.5 –** Rapports  *Jc* and *J*H

at different times of the simulation: at the beginning, at the peak and at the end:

*Jtot*

according to different values of *ε*.

*Jtot*

Not surprisingly, we notice that in all cases, the JH cost function is predominant in far from the coast and then the J*c* cost function takes over for the break-up. This can be explained because there are changes in velocities when the wave breaks and therefore the gradient is more likely to evolve. Although this work is interesting in understanding the physics behind the model despite this functional does not help us to address the weaknesses of our model. We therefore reject this functional in order to move towards other functions.

### Functional in Terms of Representing Work

Other functions were tested in the same way at the [**B.2.1**](#_bookmark10)part. The idea here is to consider a functional that would represent the notion of work. This approach aims to "artificially" adds the notion of current thank to the group velocity. The functionals tested are the following:

J*CF*10 = *εCg H*2 (B.5a)

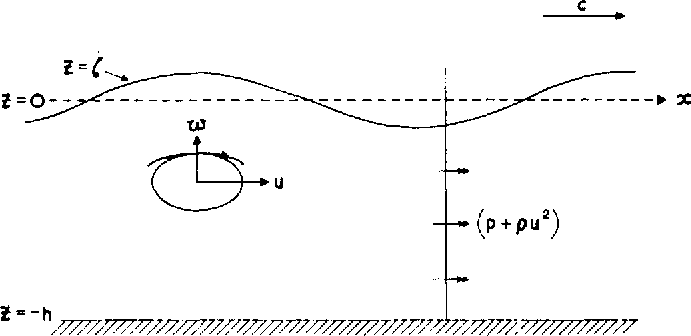
J*CF*11 = *Cg H* (B.5b)

J*CF*12 = *ε*(*Cg H*)2. (B.5c)

The differentiation of these functional has been done in a similar way to the previous part [**B.2.1**](#_bookmark10). The results coming from these functionals are very similar to those produced by the JH functional. These functions don’t add significant value to the model, so we’ve decided not to keep them.

### Functional with Radiation Stress *Sxx*

In the momentum balance on a wave (equation ([**B.1**](#_bookmark9))), there is a radiation stress term. This one represents an excess of flow is present as a result of the orbital movement of a wave. This can be observed on the figure [**B.6**](#_bookmark15). This excess dissipates mainly on the bottom friction.



**Figure B.6 –** Diagram of momentum balance on a wave.

Longuet-Higgins et al. (1962) have sought to quantify this excess flow name the radi- ation stress *Sxx*. To quantify this variable, the following expression (equation ([**B.6**](#_bookmark16))) was established.

*Sxx*

*η*

= (*p* +

∫

−*h*

*u*˜2) *dz* 0

−*h*

∫−

*ρ*

*p*0*dz* (B.6)

The first term of this expression is the total flux of momentum of a wave averaged. It is then subtracted from it the average flow in absence of a wave. This quantity *Sxx* is thus to be seen as the difference between the time-averaged flux of momentum and the average

flux in absence of wave. The work of Longuet-Higgins et al. (1962) was to simplify the expression of *Sxx* by a simpler expression namely:

*Sxx* = 2 *Cg* − 1 *E* = 1 *ρg* 2*kh* + 1 *H*2 [*J*.*m*−1] (B.7)

*C*

2

8

sinh 2*kh*

2

with *Cg* the group velocity [*m*], *ρ* the water density [*kg*.*m*−3], *g* the gravitational constant [*m*.*s*−2], *H* the significant height and *E* the wave energy. The new approach would be to postulate that *the system will try to minimize its energy in the sense of momentum:*

*minimize the slopes and currents of it*. The idea transmitted through this is to suppose that the system tries to *minimize the mechanisms by which this energy is transmitted to it*. This would mean in our case to reduce the spatial gradient of the radiation stress. To be clearer, we would try to minimize the following functional:

J = *ε*∇*x*(*Sxx*) [*J*.*m*−2] (B.8)

with *ε* in [*m*] chosen arbitrarily. The differentiation of these functions will be done in a similar way to the previous part [**B.2.1**](#_bookmark10). Comparing the results obtained with this functional with the others, we notice strong oscillations on the morphodynamics. This can be explained by the fact that the gradient is calculated numerically: this can induce many numerical biases. Moreover, calculating the gradient will necessarily lead to oscillations since the values of *Sxx* (equation ([**B.8**](#_bookmark17))) are terms in sinh which oscillate a lot. We can try for example to calculate a functional based only the radiation stress of the following form (equation ([**B.9**](#_bookmark18))):

J = *Sxx* [*J*.*m*−2]. (B.9)

Like the previous part, this functional gives results very close to those obtained by the functional JH. This is because the expression equation ([**B.8**](#_bookmark17)) shows that this is simply a slightly more complex form of energy function. It is therefore normal to obtain results very close to those obtained by the JH. We also reject these functional. All previous work on functional amounts to performing calculations from the same linear theory of physics with variables often very close to the calculation of JH.

### Functional with Memory Term

A new approach, quite different from the previous ones, would be to incorporate "mem- ory" into the functional. This approach was inspired by (Mohammadi et al. 2014) where we add a constraint on the movement of the sand requiring a minimum of bathymetric changes over the time interval [*t* − *τ*, *t*] with *τ* chosen so that *τ* >> *T*0 and *T*0 is the wave

period. This gives us the following functional:

= 1 *ρwg*

8

J ∫

Ω*S*

*H*2*dx* + *ρsg*

Ω*S*

∫

(*ψ*(*t*) − *ψ*0(*τ* − *t*))2*dx*, (B.10)

with *ρs* the density of sand [*kg*.*m*−3], *ρw* water density [*kg*.*m*−3], *H* significant water height [*m*], *g* the gravitational constant [*m*.*s*−2], *ψ* bathymetry [*m*] and *ψ*0 initial bathymetry [*m*].

By differentiating the equation ([**B.10**](#_bookmark19)), we obtain:

∇*ψ J* = ∇*ψ JH* + 2*ρsg*(*ψn* − *ψn*−1) (B.11) and so we can expand using the descent equation (**5.16**) to arrive at:

*n*+1 = *n dt*ΛΥ ∇*ψ*JH , (B.12)

*ψ ψ* −

1 − *dt*ΛΥ*ρsg*

in an unconstrained configuration. This equation is very similar to the equation ([**B.10**](#_bookmark19)). The results obtained with this functional are still very similar to those obtained with JH. They are almost identical when we use *τ* = *dt* presented in [**B.1.1**](#_bookmark3). The idea of acting on the numerical scheme can be an interesting approach. We could add more physics, add this notion of transport by adding a transport term in the numerical descent scheme as we will see in the next part.

## Adding Transport in the Descent Scheme

By discretizing the descent equation (**5.16**), we obtain the following basic descent equation without constraints:

*ψn*+1 = *ψn* − ΥΛ∇*ψ*J . (B.13)

*i*

*i*

We can easily add a term representing a horizontal transport according to a speed *V*

[*m*.*s*−1] which would transform the equation ([**B.13**](#_bookmark20)) into a new equation:

*ψn*+1 = *ψn* + ΥΛ∇*ψ*J (*ψn*) − *ρVn*∇*xψn*. (B.14)

This equation can be easily implemented by calculating ∇*xψn* by finite differences. For our tests, we take a constant speed *V* = 0.001*m*.*s*−1 and the orbital velocity *V* = *Uorb*. For the case 1 ([**B.1.1**](#_bookmark3)), we obtain the following final bathymetry results *ψf* on figure [**B.7**](#_bookmark21).

Evolution of seabed *f* for differents velocitys with descent transport equation

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  | |  |
|  |  |  |  |  |  |  | |  |
|  |  |  |  |  |  |  | |  |
|  |  |  |  |  |  |  | |  |
|  |  |  |  |  |  |  | |  |
|  |  |  |  |  |  |  | *h*0 |  |
|  |
|  |  |  |  |  |  |  | 0: initial  V=0 *m*. *s* | 1 |
|  |
|  |  |  |  |  |  | V=10e-3 | | *m*. *s* 1  *m*. *s* 1] |
|  |  |
|  |  |  |  |  |  |  | |  |

7

V=*UOrb* [

6

5

Seabed [m]

4

3

2 

1

0

0 100 200 300 400 500 600

Distance from deep to sea [m]

**Figure B.7 –** Evolution of *ψf* bathymetry for different speeds with the descent transport equation.

The case where *V* = 0.001*m*.*s*−1 (green) shows us that transport works well. However, it makes no physical sense. This is not realistic because we should not have any sand displacement in deep-water. The case where *V* = *Uorb* seems quite realistic. Moreover, it

shows a displacement of the bar without velocity towards the side. This could possibly lead to better results on case 2 [**B.1.2**](#_bookmark5). Performing analogous simulations, we obtain the following results figure [**B.8**](#_bookmark22).

Evolution of seaved *f* for differents velocitys with descent transport equation

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  | |
|  |  |  |  |  |  |  |  | |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | |
|  |  |  |  |  |  |  |  | |
|  |  |  |  |  |  | *h* |  | |
|  |  |  |  |  |  | 0  0: initi | al | |
|  |  |  |  |  |  | V=0 *m*.  V=*UOrb* | *s* 1  [*m*. *s* 1] | |
|  |  |  |  |  |  |  |  | |

0.7



0.6

0.5

Seabed [m]

0.4

0.3

0.2

0.1

0.0

0 50 100 150 200 250 300

Distance from deep sea [m]

**Figure B.8 –** Evolution of *ψf* bathymetries for different speeds with the descent transport equation.

*C. Configuration File of XBeach*

In this case, we don’t see the displacement of the sedimentary bar we’d hoped for. Although the results are not promising, this approach is still very interesting for defining a current in our model. Assuming we have a wave-to-wave resolution model, we could obtain the actual current *u*. This would be much more relevant than orbital velocity.

## Conclusion

This part focused on a functional approach to account for a better physics by trying to solve the limitations stated by case 1 ([**B.1.1**](#_bookmark3)) and 2 ([**B.1.2**](#_bookmark5)). Many other functional have been tested, such as some with currents (from SWAN / XBeach models), bottom stress,

.... Some were interesting but none of them was conclusive enough to lift the limits with this functional approach. However, as indicated in the chapter **3** in the section **5.2**, the velocity remains to be defined cleanly in order to be robust on this transport.

# Configuration File of XBeach

params.txt

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%% XBeach parameter settings input file %%%

%%% %%%

%%% case DISCOVER1 %%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%% Bed composition parameters %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% rhos = 2650

D90 = 0.0002

%%% Grid parameters %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

depfile = bathy\_psi.dep posdwn = -1

nx = 179

ny = 0

alfa = 0

vardx = 1 xfile = x.grd yfile = y.grd

|  |  |  |
| --- | --- | --- |
| xori | = | 0 |
| yori | = | 0 |
| thetamin | = | -180 |
| thetamax | = | 180 |

dtheta = 360

%%% Model time %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

tstop = 151

CFL = 0.900000

%%% Morphology parameters %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

morstart = 0

morfac = 0

%ne\_layer = bathy\_D1\_b.dep

%%% Waves %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

instat = 0

Hrms = 0.42

Trep = 8.0

dir0 = 270

%%% Output %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

tint = 1

tstart = 150

nglobalvar = 6 zs

zb H

k u

taubx

# Configuration File of SWAN

params.txt

PROJECT 'maupiti1D' 'HOE'

$ Setting the configuration

$##########################

*D. Configuration File of SWAN*

SET LEVEL 0.0 NOR 90 INRHOG 1 MODE STATIONARY ONEDIMENSIONAL COORDINATES CARTESIAN

$ Definition of the grid/ bottom condition

$#########################################

$ REGULAR [xpc] [ypc] [alpc] [xlenc] [ylenc] [mxc] [myc] [mdc] [flow] CGRID REGULAR 0 0 0 180 0 179 0 CIRCLE 12 0.02 0.4 36

$ INPGRID BOTTOM REGULAR [xpinp] [ypinp] [alpinp] [mxinp] [myinp] [dxinp] [dyinp] INPGRID BOTTOM REGULAR 0 0 0 179 0 1 0

READINP BOTTOM -1 'psi.dat' 3 0 FREE

$ Setting physical quantities for the simulation

$###############################################

$ DIFFRACtion 1 0.2 OFF QUAD

GEN3

BREAKING BKD 1 0.73 7.59 -8.06 8.09 TRIAD

$ Definition of forcing conditions

$#################################

BOUND SHAPESPEC JONSWAP 3.30 PEAK DSPR DEGREES

$ PAR [hs] [per] [dir] [dd]

BOUNDSPEC SIDE West CON PAR 0.42 8.0 0 20

$ Si on met 20, on réduit le nombre de basse fréquence qui est à Nan -999.

$ Definition of output features

$##############################

CURVE 'profil' 0 0 179 179 0

TABLE 'profil' HEADER 'swan\_output\_HSIG.dat' HSIGN TABLE 'profil' HEADER 'swan\_output\_T0.dat' TM01

$ Calculating waves

$####################

COMPUTE STOP

$

$ Fin du calcul

$############################# STOP