Identifying lateral boundary conditions for the M2 tide in a coastal model using a stochastic gradient descent algorithm

Guillaume Koenig, Clement Aldebert, Cristele Chevalier, Jean-Luc Devenon

Version of Record: https://www.sciencedirect.com/science/article/pii/S1463500320302110
Manuscript_f673980ef423b867cb359cdf3dd26580

Identifying lateral boundary conditions for the M2 tide in a coastal model using a stochastic gradient descent algorithm

Guillaume Koenig[a], Clement Aldebert[a], Cristele Chevalier[a], Jean-Luc Devenon[a]

[a] *Aix-Marseille Universite, Universite de Toulon, CNRS, IRD, MIO, UM 110, 13288 Marseille, France*

Abstract

While lateral boundary conditions are crucial for the physical modelling of ocean dynamics, their estimation may lack accuracy in coastal regions. Data-assimilation has long been used to improve accuracy, but most of the widely-used methods are difficult to implement. We tried a new and an easy-to-implement method to estimate boundary conditions. This method uses data assimilation with a stochastic gradient descent and successive approximations of the boundary conditions. We tested it with twin experiments and a more realistic setting on a tidal model in the lagoon of Ouano, in New-Caledonia. The method proved successful and provided good estimation of the boundary conditions with various settings of subsampling and noise for the pseudo-data in the twin experiments, but there were important oscillations in the experiments with more realistic settings. Here we present those results and discuss the use of our new and easy-to-implement method.

Keywords: Parameters identification Tidal modelling Stochastic algorithms Data assimilation

[☆] Fully documented templates are available in the elsarticle package on CTAN.

1. Introduction

Lateral boundary conditions for physical models are of major importance and very difficult to tune with accuracy in coastal regions (James, 2002).They strongly constrain the water circulation inside the domain, by controlling the exchanges at the boundaries of the domain of interest. In many applications, they are derived from larger scale models, either with nesting techniques or with interpolations. Unfortunately those boundary conditions are not always directly interpolable, especially for coastal regions where the bathymetry, frictional effects and submesoscale variability constrain the small-scale dynamics.

A simple solution can be to implement data-assimilation to adjust the boundary conditions (Taillandier et al., 2004). However, some of those data assimilation techniques require the development of an inverse model to compute the corrections to apply to the boundary conditions (Devenon, 1990). This process is time consuming, and requires a good knowledge both of the direct model and of inverse method discretization theory (Lellouche et al., 1998). Finally, the inverse model can simply be impossible to develop if the code of the direct model is not available.

Some other methods do not require the coding of an inverse model. Some methods use repeated runs of the direct model with perturbations in the parameters to generate approximations of the direct model that can then be inverted, such as the reduced model method of Vermeulen and Heemink (2006), Altaf et al. (2012) or Hoteit and Köhl (2006). Other methods do not use approximations of the direct model and directly use the perturbations of the parameters to determine the best set of parameters, such as the Simultaneous Perturbations Stochastic Approximation (SPSA) algorithm (Spall (1998), Messié et al. (2020),). Those two kinds of methods both require limited coding work, since they only need access to the parameters to be estimated in the model and to outputs of the model, and can therefore be implemented on any model. Their main limitation is that they require large number of iterations of the direct model. And unlike methods relying on an inverse model, this number of iterations grows

exponentially with the number of parameters. Therefore, parameters such as lateral boundary conditions, which are clusters of at least hundreds of grid points, each with independent values, were out of the reach of such methods.

But those methods could be used to estimate approximations of those kinds of parameters. This is done naturally for the reduced model methods by the decomposition in empirical orthogonal functions. But for the SPSA algorithms, finding good approximations is still an open problem. For example, Boutet et al. (2015) used the SPSA algorithm for determining friction coefficients in the Gascogne Gulf, by covering the gulf with a domain of constant friction according to the type of sediments found and outside the domain of oceanography, Tympakianaki et al. (2015) used clustering techniques to generate approximations. The contribution of Altaf et al. (2011) was important in that regard, because they compared the results of the SPSA algorithm with piecewise approximations with the results a reduced model for the calibration of the depth in a North Sea model. They concluded that the SPSA method had the advantage of being able to identify many parameters at once with these approximations.

However the use of piecewise constant functions quickly becomes difficult when the parameters vary continuously, or when the precise location of those corrections is not known in advance. Examples of cases where those corrections are hard to know a priori are coral lagoons, where the steep bathymetry, the presence of a coral reef and the optional connexion with other lagoons can bring unexpected variations in currents and sea level.

Most approximation methods try to identify the coefficients of a given set of basis functions for approaching boundary conditions. This approach is limited because the only way to improve the approximations is to redo the entire approximations with new basis functions. Instead, we chose to use several sets of basis functions whose coefficients could be easily linked from one set to another, so that approximations could be improved without having to redo the whole procedure. This methodology had already been tested (Jahns (1966), Thevenaz et al. (1998)), but never for boundary conditions in oceanography to the authors' knowledge.

We performed twin experiments on a tidal model of the Ouano lagoon, in New-Caledonia to test our method. It is of interest as the boundary conditions are complex to determine, due to the influence of the wave-breaking patterns, the tides and the high bathymetric and friction variations due to the coral barrier (Sous et al., 2017). It enabled us to see how the approximations worked in a complex situation, where boundary conditions sometimes vary very sharply and sometimes are nearly constant. In addition by adding noise and subsampling the data, we tested the robustness of the method in more realistic settings. We finally tested the method in a more realistic setting with no a priori knowledge of the boundary conditions and data collected during survey of the lagoon.

2. Material and methods

2.1. Study zone and parameters

We studied the circulation patterns of the Ouano lagoon. This lagoon is about 30 km long and 10 km wide and is connected to a wider lagoon system located on the south-west coast of New-Caledonia. The coral reef separates the Pacific Ocean and the lagoon, and forms a porous wall with only 2 passes directly linking the ocean and the lagoon. The currents inside this lagoon are dominated by tides, with a spring amplitude of about 0.8 m (Chevalier et al. (2015), Sous et al. (2017)). The wave-breaking occurring on the barrier reef also drives the currents (Chevalier et al., 2015) as it creates a horizontal gradient of radiation stress across the coral barreer that leads to a water flux inside the lagoon (Sous et al., 2017). The wave-breaking and the currents it generates are in addition influenced by the tidal level of the lagoon. Therefore the determination of accurate tidal boundary conditions in the lagoon is necessary for any proper physical modelling.

Tidal boundary conditions were provided by C. Chevalier, so as to be able to test the method under realistic conditions. These consisted in phase and amplitude for the M2 component of the sea level are shown in Figure 1 . Both variables exhibited zones with strong variations, due to the fact that the southern

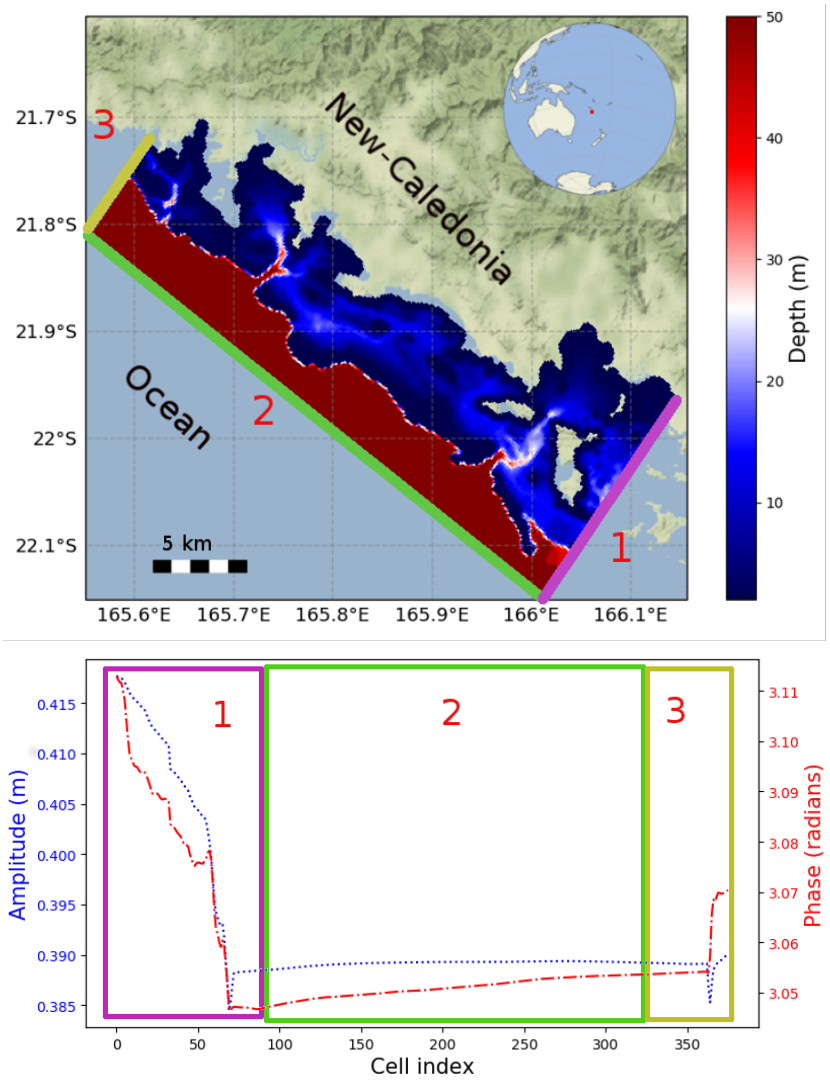

Figure 1: The upper image shows the Ouano lagoon between the ocean and the island of New-Caledonia, with the location of New-Caledonia given by the red dot in the world-map insert. We give the bathymetry of the lagoon and of the nearby ocean that we used in the model. Some water areas of the lagoon do not have bathymetry values on the map because they are very shallow reef or mangrove and we have not considered them here. The different boundaries used in the model are indicated by a colour-scale. Depths over 50 meters are found outside the lagoon. In the lower graphic, we present the boundary conditions for the amplitude and phase of the M2 tide. The colour rectangles indicate the position of the different boundaries.

boundary and the northern boundary passed through the barrier and therefore encompassed both the ocean and the lagoon. Furthermore, boundary conditions in shallow zones may have been influenced by the current coming from the rest of the New-Caledonia lagoon complex (Ouillon et al. (2010),Jouon et al. (2006)).

The model used here is a reduced tidal model. It is based on a decomposition of the velocities and elevation in tidal modes in the shallow-water equations. It was used because the equations can then be reduced to a purely boundary conditions problem. The derivation of the model is given in the appendix.

2.2. Gradient descent algorithm

To identify boundary conditions for each of our successive problems, we used a gradient descent algorithm. The one we used is strongly inspired by the Simultaneous Perturbation Stochastic Approximation (SPSA) algorithm (Spall, 1998), that we adapted in order to speed up convergence.

As for any gradient descent algorithm, the objective was to determine the gradient of a cost function according to certain parameters and to follow the gradient to determine an extremal value. In our case, we wanted to minimize the discrepancies between the output of a model and pseudo-data previously generated, by adjusting the boundary conditions of the model. The metrics used are shown in equation 1.

$$J(\theta) = \frac{1}{2}\sum_{k}(F(\theta)_k - \hat{Y}_k)^2 \tag{1}$$

Here J is the cost function, θ represents the set of control parameters, being here the boundary conditions, $\hat{Y}$ represents the data or pseudo-data for twin experiments and F is an operator that represents the action of the model. Those data in our case are the complex values of the sea level. The index k serves to localize the data point and the corresponding model output. We chose this cost function because it is standard in the field and corresponds to an addition of Gaussian uncertainties. Some terms can be added in order to smooth the output

(Sasaki, 1970) or to reduce the size of the search-space (Blum et al., 2009) but those were not used here.

Now we have to determine new parameters incrementally. If we index the iteration of the update of parameters with i, then we can write $\overrightarrow{g}_i$ as the gradient of the cost function according to the parameters and a_i =a parameter used for controlling the size of updates at the iteration i, the gradient descent then update the parameters is written as ,

$$\theta_{i+1} = \theta_i - a_i \overrightarrow{g}_i \tag{2}$$

The modification of parameters is done until a stopping criterion is reached. This criterion can be a measure of the magnitude of the gradient $\overrightarrow{g}_i$ or a number of iterations of the gradient descent, for example. In this procedure, the value of the parameters a_i is crucial, as well as the value of the gradient $\overrightarrow{g}_i$, as it indicates the direction to reach the minimum.

Here, we did not compute the gradient directly but rather an approximation to the gradient. The approximation was the one used in the SPSA (Spall, 1998) algorithm and corresponded to a directional derivative, as shown in system 3 :

$$\begin{cases} \theta_{i\pm} = \theta_i \pm c_i \Delta_i \\ \frac{\partial J(\theta_i)}{\partial \Delta_i} = (J(\theta_{i+}) - J(\theta_{i-})) * \Delta_k * c_i^{-1} \\ \hat{g}_i = \frac{\partial J(\hat{\theta}_i - a_{i-1} \overrightarrow{g}_{i-1})}{\partial \hat{\Delta}_i} \\ \overrightarrow{g}_i = \hat{g}_i + \beta \overrightarrow{g}_{i-1} \end{cases} \tag{3}$$

In this algorithm the direction was determined by a random perturbation of all the parameters, and the perturbation $c_i\Delta_i$ was determined by a predetermined term c_i and a sign term Δ_i given by a Bernouilli law. The perturbation and its symmetric counterpart are used to determine a centered estimate of the derivative of the cost function in this direction. Finally, to provide an estimate closer to the real value of the gradient, we used a momentum technique, as described in Sutskever et al. (2013). This technique allows a smoother descent

by averaging the current estimate of the gradient with the previous estimates of the gradient. Hence the determination of the gradient is given in system 5 :

$$\overrightarrow{g}_i = \overrightarrow{g}_i + \beta \overrightarrow{g}_{i-1} \tag{4}$$

$$\tag{5}$$

where β is a parameter between 0 and 1 according to the importance we want to give to previous estimates. In many applications this sum is normalized, but here, we decided to take another option. This is due to the adjustment we made to our algorithm compared to the standard SPSA. Normally, the gain coefficient a_i and the perturbation coefficient c_i are supposed to be decreasing as in the system 6 :

$$\begin{cases} a_i = \frac{a_0}{(A+i)^\gamma} \\ c_i = \frac{c_0}{i^\delta} \end{cases} \tag{6}$$

where a_0 and c_0 are initial estimates of the perturbations and gain coefficients, A is a regularization term for the decrease and γ and δ are exponents under one Spall (1998). However, we used a different approach: we kept the perturbation amplitude c_i constant but started with a small value for the gain coefficient a_i and increased it in geometric fashion during the gradient descent. This avoided the necessity of choosing the right parameters for the gain of every subproblem. The gain coefficient was updated to take into account the fact that we were going to solve of succession of problems and that we would not be able to tune the gain coefficient a_i for each problem. This is shown in the system 7 :

$$\begin{cases} a_i = a_0(1+q)^i \\ c_i = c_0 \end{cases} \tag{7}$$

where q was a small parameter for the increase.This growth value was chosen so that the algorithm would fit every subproblem. We implemented a line-search

procedure. We used the procedure described in Armijo (1966) that checked whether the parameter modifications effectively reduced the cost functions, and if not, reduced the gain coefficient a_i a few times by a factor depending on the ratio between the estimated cost function and the lowest one. If after some reductions of the gain coefficient a_i the cost function still did not decrease, the gain coefficient a_i was reset to its initial value and momentum terms $\overrightarrow{g}_{i-1}$ were cancelled for the gradient estimation.

In the SPSA algorithm, the gain coefficients and the perturbation coefficients are often decreasing, with the goal of being as small as possible in the last stage of the parameters estimation in order to make only small adjustments around the minimum. The growth procedure used in our case may seem to do the opposite. However, the line-search algorithm tends to reduce the gain coefficient if we are close enough to a minimum. If we combine this with a very small initial gain coefficient, we could expect that close to a minimum, our algorithm and the SPSA would show close behaviour, by only performing small adjustments. In contrast, at the beginning of each subproblem, we expect an important growth of the gain coefficient so that it adjusts to the sub-problem.

Since the gradient descent algorithm relied on few parameters, we report them in table 1.

Table 1: Numerical parameters for the gradient descent algorithm. The units are given for the case were the parameters are related to the phases or to the amplitudes.

a_0	c_0	β	q
m^{-2} or no unit	m or $rad.s^{-1}$	No unit	No unit
1e-2	1e-8	0.8	0.03

2.3. Successive approximations

As mentioned in the Introduction, here we suggested determining approximations of the boundary conditions. The boundary conditions conditions were determined by amplitude and phase at each boundary point to give a complex number, as shown in equation 8

$$\eta_k = |\eta_k| e^{-i\phi_k} \tag{8}$$

where $|\eta_k|$ is the local amplitude and ϕ_k the local phase. One can set the boundary conditions on a 1D vector as shown in Figure 1. It is then possible to approximate boundary conditions with one dimensional functions. In this case, the phase and amplitude can be determined by equation 9 :

$$|\eta_k| \text{ or } \phi_k = \sum_{j=0}^{M} f^N(b_j, x_j, x_k) \tag{9}$$

where f^N were the functions used at the level of approximation N, b_j were their coefficients and M the number of such functions used. Due to the fact that we used nodal functions for the approximations, we also needed a parameter x_k that specifies the position where the value of the function is fixed; those positions are called the nodes.

The nodal functions we used are cubic splines with natural boundary conditions. Those functions are a collection of piecewise cubic polynomial linked by continuity and continuity of the first derivative. Their junctions are situated at nodes x_k and their values at those nodes at the b_j values.

$$\begin{cases} f^N(b_j, x_j, x_k) = \sum_{j=0}^{M} (a_j x^3 + b_j x^2 + c_j x + d_j) \Pi((x_k - x_j)/x_j) \\ f^N(b_j, x_j, x_j) = b_j \\ \frac{df^N(b_j, x_j, 0)}{dx} = c_0 \\ \frac{df^N(b_j, x_j, t)}{dx} = c_M \end{cases} \tag{10}$$

Where here $\Pi((x_k - x_j)/x_j)$ is the door function that is worth 1 between x_j and x_{j+1} and 0 elsewhere. The computation of the coefficients a_j,b_j,c_j and d_j is done following the python implementation of the algorithm of the Scipy package of Python (Sci (2020), De Boor (1978))

Overall, our aim was to determine the parameters b_j that minimized the cost function for a given level of approximation N. Once a minimum is reached

or approached with a given number of nodes, we used this approximation to generate a new set of nodes and node values. Then, we performed another gradient descent to identify the minimum of the cost function relative to the new set of parameters. We thought that reaching the new minimum would require few iterations of the gradient descent, since the last step of minimization had brought us to a good starting point. An example of this approach by successive approximations is shown in Figure 2.

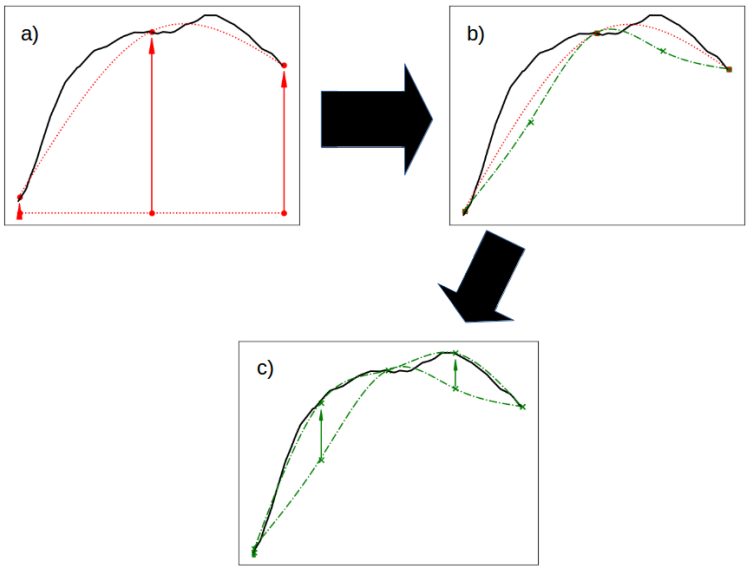

Figure 2: Example of the first stage of boundary conditions identification. The black line here represents the boundary conditions that we are trying to determine. The red dots and green crosses represent the positions of the nodes for the spline interpolations. The red curve represents the spline approximation with only three nodes, while the green curves represent the approximation with five nodes. We begin by optimizing a three-node interpolation (a). Then we use those three-node curves to generate a rough approximation with five nodes (b). Then we adapt the values of the five nodes to optimize the boundary conditions with the green curve (c).

The new nodes were created midway between existing nodes. Also, the

values of the new nodes were determined by linear interpolation between the values of the previously existing nodes.

2.4. Implemented tests

2.4.1. Twin experiments and pseudo-data

We performed twin experiments to test our parameters identification algorithm. We began by running our reduced model with known boundary conditions for the M2 tide. Those boundary conditions were boundary conditions taken from the previous work of Chevalier et al. (2015). We used the output of this run as references for our first assimilation. We will refer to them as pseudo-data in the rest of the paper, as they will be used as data for our assimilation but were initially produced by the model. That ensures a total compatibility of this data with the physics of the model and avoids discrepancies between the model outputs and in-situ data.

We then used pseudo-data to try to identify the boundary conditions that served to produce them. Since those pseudo-data were outputs of the model, we knew we would not have any compatibility issues between the model and the data. The model was a linear one, we knew that we had only one set of boundary conditions corresponding to the pseudo-data, if we restricted the domain of the phase from 0 to 2π radians.

We also tested conditions closer to a real data assimilation by adding noise and decimating the pseudo-data, so that they would be closer to real in-situ data. It enabled us to know how the noise on data acted on the boundary conditions that were determined and how the position and number of data could affect the shape of the boundary conditions.

Finally, we performed a final experiment with in-situ data and our reduced model.

2.4.2. Parameter identification methods

We wanted to prove the efficiency of the successive approximations method was not due only to a good gradient descent algorithm but also to a well-

chosen set of basis functions for the approximation. We then compared the results obtained with this method with results obtained by two other methods. Those methods did not use successive approximations, and only one of them used approximations with bicubic splines. But they used the same SPSA-like gradient descent algorithm.

We ran the three methods with the same number of iterations of the gradient descent algorithm. We also ensured that for the methods that relied on spline approximations the approximations presented the same number of nodes at the end of the parameter identification. Finally, we started with the same constant values of parameters on the entire boundary for each method.

For the successive approximations method, we began with a small number of nodes and generated new ones after a certain number of iterations of the gradient descent algorithm. Those numbers of iterations were determined by trial and error because they generally presented a sufficient decrease of the cost function.

The second method used only one level of approximation and is referred to as the high-order approximation method. The number of nodes used in the high-order approximation method is the same as the number of nodes in the final stage of the successive approximation method.

The third method did not use approximation, and directly identified the values of phase and amplitude for each grid cell that is situated at the open boundary. We refer to this method as the no-approximation method.

We decided to split the parameter identification for amplitude and phase for the three methods. In the successive approximations methods, we alternated the identification of amplitude and phase. Also, in the successive approximations method we used more gradient descent iterations for the phase than for the amplitude.

The parameter for the implementation of the three methods in our specific case can be found in Table 2.

Table 2: Numerical parameters for the different methods. We refer to those parameters as "NA" (Not Applicable) when the method mentioned does not use the procedure involving those parameters.

Parameter	Successive Approximations	High-order Approximation	No Approximation
Number of stages	5	NA	NA
Increment of iterations between stages	20	NA	NA
Increment of iterations between amplitude and phase	10	NA	NA
Number of iterations for the first stage	20	NA	NA
Initial number of nodes	3	33	397
Final number of nodes	33	33	397
Total number of iterations	1261	1260	1260

2.4.3. Noise and subsampling

Parameter identification was also performed in the presence of noise and subsampling. For this purpose, we added noise to the pseudo-data by adding a uniformly distributed noise to each data point. We used different levels of noise but always paired the level of noise of amplitude with that of phase. We tested respectively noise levels of 0.0001 m and 0.0001 rad, 0.001 m and 0.001 rad, 0.01 m and 0.01 rad and 0.1 m and 0.1 rad. In the Results section, those levels of noise are labelled with their numerical value only and not the unit; we will use "Noise of 0.1" and not "Noise of 0.1 m and 0.1 rad". Those levels of noise were chosen to show the increase of impact of noise in the data. In addition, the middle values of $0.01m$ and $0.01rad.s^{-1}$ are of the order of uncertainties that

Table 3: List of the numerical experiments presented in the article.

Methods	Sampling	Noise m and rad
Successive Approximations	All grid cells	No
High-order Approximations	All grid cells	No
No Approximation	All grid cells	No
Successive Approximations	1/4, 1/64, 1/256 and 1/1024 of all grid cells	No
Successive Approximations	All grid cells	0.0001, 0.001, 0.01 and 0.1
Successive Approximations	66 grid cells in the southern part, the ocean or the lagoon	No
Successive Approximations	4 sensors in the lagoon	No

we have when we treat our data of the tidal elevation of the lagoon with the Python implementation of Codiga (2011). The values of $0.1m$ and $0.1rad.s^{-1}$ correspond to extremely high noise value, since we expect tidal amplitudes of the order of $0.4m$, and they serve to test the robustness of the method.

The subsampling was performed by only keeping some pseudo-data. We removed them in the same way in the two directions and with no special consideration for the land. We respectively kept one pseudo-data on 2, 4, 16 and 32 in each direction for each case, which ended up with only one peudo-data on 4, 64, 256 and 1024, 4. We chose levels which respectively correspond to 16275, 4143, 240, 55 and 12 available data points. This number of data will be use as a label in the experiments.

Finally, we also performed a few tests with localized pseudo-data. This was done in order to see how unequally distributed data would influence the parameter determination method and how robust it was. All those experiments

are reported in table 3.

2.5. Computational cost of the algorithm

The code of the reduced model took about eight seconds to run on our local computer with a Core i7 processor from Intel, while an entire assimilation loop, comprising three runs of the reduced model, an estimation of the cost function and the update of the parameters took 28 seconds on average with our Python implementation.

3. Results

3.1. Results obtained with successive approximations

We first present the boundary conditions determined with the successive approximations method in Figure 3. We see different stages of the successive approximations method. If those stages are all around the reference boundary conditions, the 5^{th} stage has less oscillations around the true value. Those oscillations are bigger between the boundary index 0 and 70 and around the boundary index 370, where variations are high. Between the boundary index 0 and 370 all the variations of the phase are not captured even at the 5^{th} stage. Furthermore, we see that the variations in this region are less steep at the 3^{rd} stage than at the 5^{th}.

In Figures 4 and 5 we present the output of the model with the boundary conditions determined with the successive approximations method. The output for the parameters determined with the successive approximations method are close to the original, the only variations being in some oscillations close to the southern and northern boundaries. Apart from those, variations are of the order of one millimeter for amplitude and 1/1000 of a radian for the phase.

3.2. Results obtained with the other two methods

We present the amplitude and phase determined with the no approximation and the high-order approximation method in Figure 6, so that a comparison

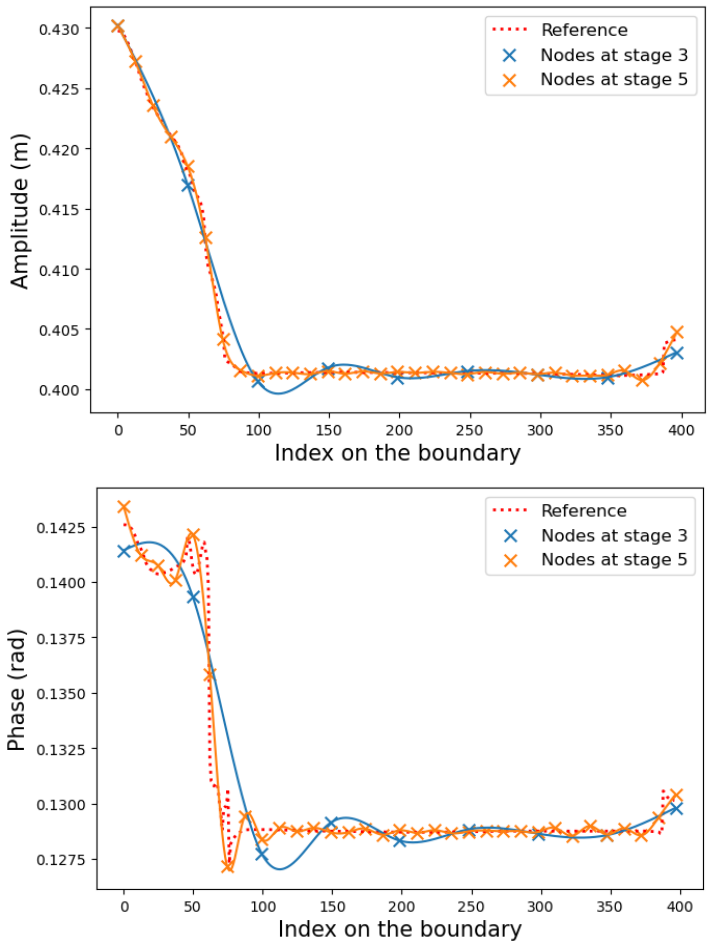

Figure 3: Comparison of the reference boundary conditions (dotted red lines) and those determined by the successive approximations method and the SPSA, at two different stages of approximation. The green dotted lines represent the boundary conditions determined at the second stage of approximation, and the blue crosses the position of the nodes during that stage. The blue semi-dotted lines represent the boundary conditions determined at the fourth stage of approximation, and the orange crosses the position of the nodes during that stage. In the upper plot we show the boundary conditions for amplitude and in the lower plot the boundary conditions for phase.

with the results of the successive approximations method can be made. We see
320 that both methods exhibit wide oscillations. Those oscillations are less marked and of larger scale for the high-order approximation method than for the no

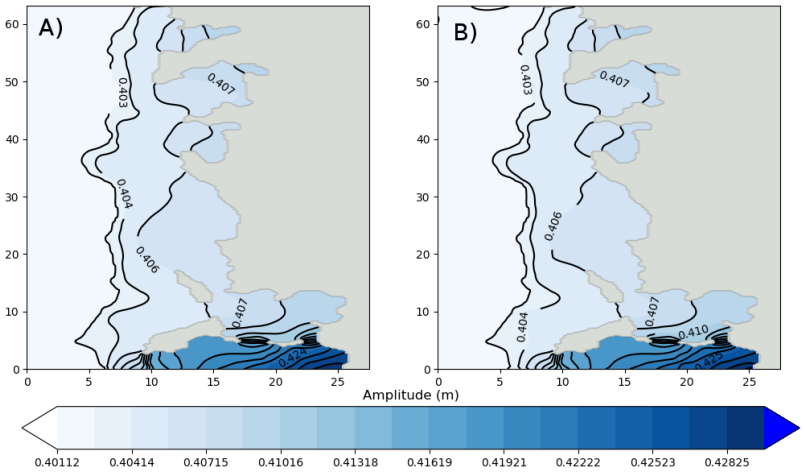

Figure 4: Amplitude output of the model in the Ouano lagoon for boundary conditions provided from a) The reference boundary conditions b) The boundary conditions determined by the successive approximations method.

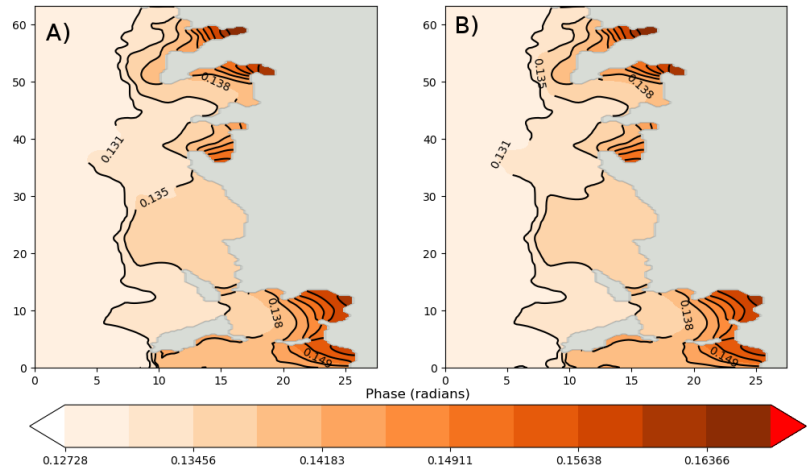

Figure 5: Phase output of the model in the Ouano lagoon for boundary conditions provided from a) The reference boundary conditions b) The boundary conditions determined by the successive approximations method.

approximation method. It is interesting to note that the boundary conditions obtained with the successive approximations method are also plotted in Figure 6 but are undistinguishable from reference boundary conditions.

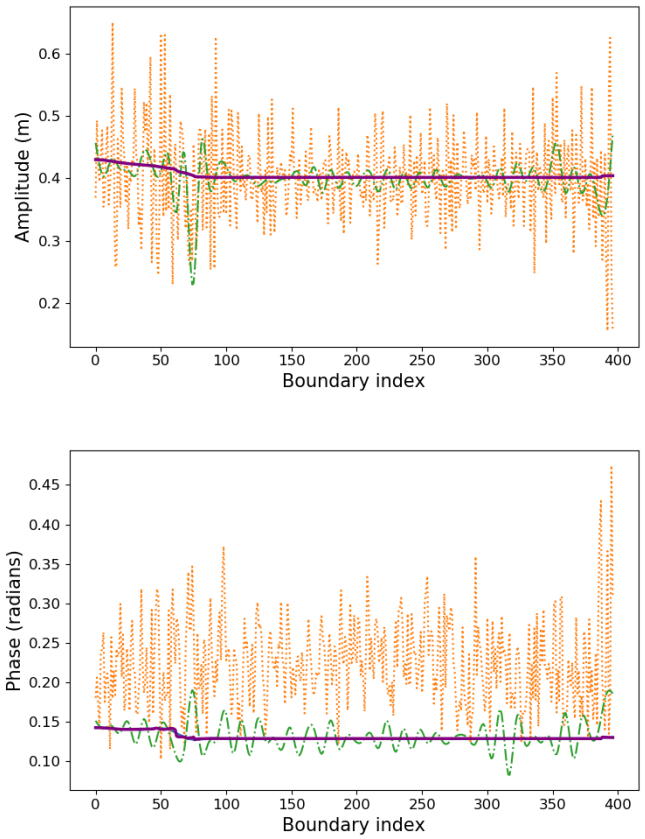

Figure 6: Comparison of the reference boundary conditions (in dotted blue) and those determined by the high-order approximation (in green) or the no approximation method (in orange). In the upper plot we show the boundary conditions for amplitude and in the lower plot the boundary conditions for phase.

3.3. Comparison of the three methods during the identification

The three methods take different paths to identify parameters. To show those different paths we present the changes of the cost function during the gradient descent for each the three methods in Figure 7. All methods give a roughly equivalent cost function value up to the 50^{th} iteration. After that point, the successive approximation method gives the lowest cost function value and the steepest slope. The curves for the cost function values of high-order approximation and no-approximation methods only separate around the 700^{th} iteration, which is the moment of the variable shift.

The three methods end up with different cost function values, with around 10^{-4} for the successive approximations method and around 1 for the high-order approximation method and 10 for the high-order approximation method. If we use the definition of the cost function, we realize that this cost function corresponds to a maximum error of 10^{-2} rad or m for the successive approximations method, if all the error was on only one grid point.

The different behaviour for the three methods is even more explicit if we take a closer look at the patterns of change in the cost function. The high order approximation method and no-approximation method have similar shapes. They have a very steep descent at first and then continue to decrease slowly. There is another steep descent when the parameter changes from amplitude to phase. For the gradient descent of the method with successive approximations, we see that we have a very steep part before the second node creation. After that, the gradient descent is smoother with a series of steep descents right after node creations that slowly turn into plateaus, forming a stairway-like overall shape. We also see that after each node creation we have an increase in the cost function.

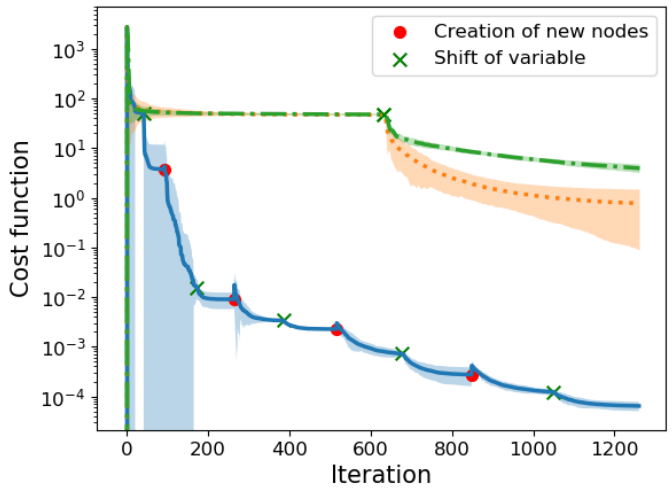

Figure 7: Evolution of the cost function value for the different methods. The filled lines correspond to the mean values observed for ten runs, and the shaded zones to two standard deviations around this value. The blue line represents the successive approximation method, the orange line the high-order approximation method and the green line the no-approximation method. The green crosses represent the shift between optimization of the amplitude and optimization of the phase. The red dots represent the iteration at which we increase the number of coefficients used in the approximation.

3.4. Noisy and subsampled pseudodata

Here we performed some boundary conditions identification where we used only a fraction of the available pseudo-data. Since those pseudo-data were the results of a reference run, they already corresponded to the position of the model output and no interpolation was necessary. To extract a fraction of those pseudo-data for the subsampling, we simply kept one data out of a certain number in both the Y and X direction. This means that if we began with 114 pseudo data in the X direction for each Y level, we end up with only 57 pseudo-data in the X direction with a subsampling level of two, and that those pseudo-data are still equally spaced. The different subsamplings used are illustrated in figure 8.

Since the decimation of the pseudo-data was done in the X and Y direction at the same time, we refer to the different subsampling experiments by the

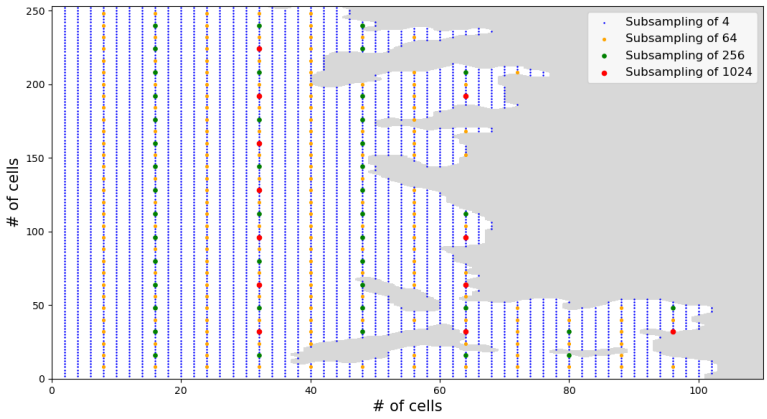

Figure 8: Map of the pseudo-data for the different levels of subsampling.

fraction of pseudo-data used in the X direction multiplied by the fraction of pseudo data considered in the Y direction. If we still use our example where we used one pseudo-data out of two in the X and Y direction, we will refer to it as the experiment with a subsampling of one data point over 4. We chose levels of subsampling of one pseudo data over 4, 64, 256 and 1024.

If we compare the number of data points available for each subsampling level, of 16275, 4143, 240 and 12 available data points and the 33 nodes of the final splines, we see that the cases with one pseudo-data out of 4 and 64 still have more pseudo-data than the final number of nodes of our procedure. The case with one pseudo data over 256 would still have more pseudo-data (55)than the final number of nodes, but the land cells actually makes it close, while the case with one pseudo-data over 1024 has less pseudo-data points than the final number of nodes. At the end of the procedure, we therefore expect it to behave as an underdetermined problem.

We show the results with different levels of subsampling in Figure 9. We see that results are fairly close until the subsampling reaches one data point over 256. With so few data some oscillations begin to appear, especially in the phase. Also, the conditions determined close to the northern boundary differ

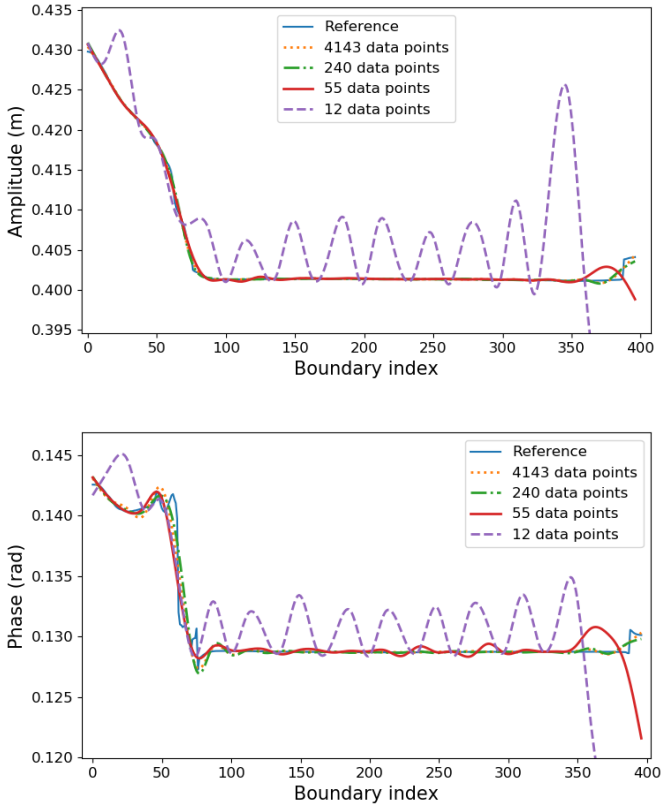

Figure 9: Variation of the boundary conditions determined for different levels of subsampling in the pseudo-data.

strongly. Those oscillations are stronger with one data point over 1/1024. Still, those oscillations are around the reference boundary conditions.

We also present the results with different levels of noise added to the pseudo-data in Figure 10. The noise level of $0.0001m$ and $rad.s^{-1}$ produces results that are indistinguishable from those obtained without noise. The noise levels of $0.001m$ or $rad.s^{-1}$ and $0.01m$ or $rad.s^{-1}$ produce very comparable results, where small scale oscillations start to appear. Finally, the noise level of 0.1 exhibits very strong oscillations around the reference boundary conditions.

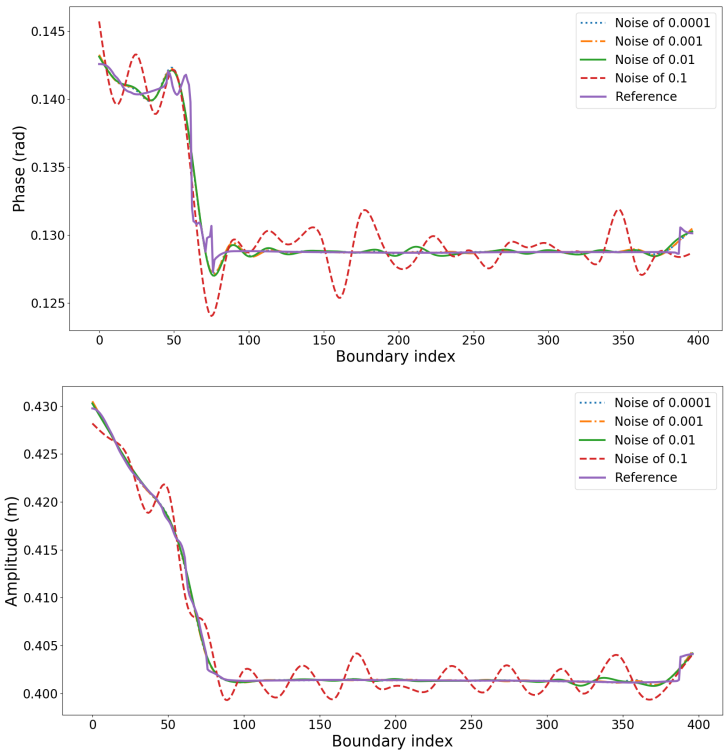

Figure 10: Variation of the boundary conditions determined for different levels of noise in the pseudo-data.

3.5. *Grouped data*

We also performed a few experiments in which we used only 60 pseudo-data located in a precise part of the domain. We tried with data in the lagoon part, data in the deeper ocean and data in the shallow part in the south-east of the domain. Those locations are visible in Figure 11.

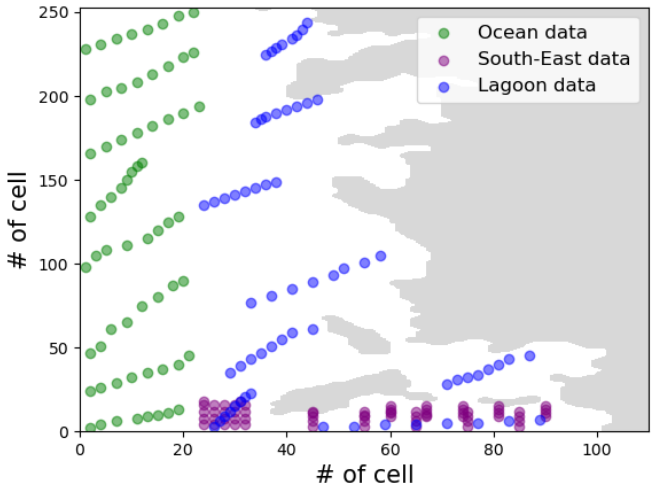

Figure 11: Position of the data for the assimilation with grouped data.

As we can see in Figure 12, the data placed in the lagoon have oscillations of the order of a centimeter relative to the true value. In the cases where the data were in the ocean, we have larger oscillations in the oceanic part of the boundary and both the phase and the amplitude of the boundary conditions are underestimated in the south-eastern boundary. When the data are located in the south-eastern shallow part of the lagoon, the boundary conditions are reconstructed with an accuracy of a few centimeters, but the estimates for the oceanic boundary conditions oscillate strongly with an overestimation of nearly 10 centimeters followed by an underestimation of nearly 20 centimeters.

3.6. Assimilation with in-situ data

We also tried to determine boundary conditions from in-situ data gathered during the 2016 campaign in the lagoon. In this situation, however, we had to reduce the number of nodes of the splines because we only had four data points. Those are shown on the figure 13.

It seems that the variations in the boundary conditions were large, as is shown in Figure 14. The mean trend of the true phase and amplitude is caught, but the oscillations are very large, with $10cm$ for the amplitude and $0.1rad$ for the phase. In addition, the maxima do not seem to be correlated with any variation in depth close to the boundaries. Those strong variations are damped in the lagoon to have outputs that are around the data, with discrepancies of around 0.05 m and less than 0.01 rad, as shown in Figure 15.

4. Discussion

4.1. Choice of the functions shape

The results obtained with the successive approximations method were very close to the original boundary conditions. Compared to the other two methods, not only was convergence speeded up but accuracy was also increased. This is interesting for development of methods for determination of boundary conditions in a coastal model. Furthermore, this method does not depend on a specific gradient descent method or type of parameters, and could be extended to other problems, such as the determination of friction coefficients, or with other algorithms, such as genetic algorithms.

We can say that the increase in accuracy and speed of convergence were indeed linked to the successive approximations, as neither the no-approximation or the high-order approximation method could approach the reference boundary conditions. The slow convergence was expected for the no-approximation method, as we knew that our algorithm had a speed of convergence close to that of the SPSA, which depends on the number of parameters, even though the computing cost for gradient estimate does not. For the high-approximation method, we expected the convergence to be faster since the number of parameters was reduced. But by comparing it with the successive approximation method, it seems that the starting point also plays an important role in the convergence rate. And the successive approximation method effectively sped up convergence by first reducing the number of parameters and next by providing good starting points for the successive problems.

This method also contrasts with what had already been done with a stochastic gradient descent algorithm. Other examples exist, such as that of Altaf et al. (2011) and Boutet et al. (2015), and Hoang and Baraille (2011) also used this algorithm for determining a reduced number of coefficients of approximation for a covariance field of a Kalman filter. In all those cases, they reached an acceptable level of precision. However, they worked with limited number of actual parameters, less than 20 each time, and with parameters whose spatial

and temporal variations were known before the assimilation.

This number of parameters is close to the number of parameters we had in our realistic application. In this case, we realized that our method led to strong oscillations. This may be linked to the positions of the nodes of the splines, or to the use of the spline approximations. This could be solved by trying a dynamic placement of the nodes or by trying other basis functions such a akima interpolation or p-chip that do not have such a curvature.

Lastly, the algorithmic implementation of those approximations is also to be considered. For example, the odd behaviour that we observed at the very beginning and the very end of our determined parameters in Figure 3 was due to the way a condition of zero normal gradient was imposed. The algorithm we used places null-derivative conditions at the extremities of the splines instead of forcing the values of the nodes at those points and extending the function with null derivative before and after those points. This illustrates that not only the functions used, but the way they are implemented must be carefully taken into account for good parameter identification.

4.2. Impact of sampling

The effects of noise and of subsampling were also investigated. In Figures 9 and 10, we show that discrepancies stayed low for noise up to 10% of the amplitude and phase value, and up to $1/16^{th}$ of the pseudo-data taken into account. With less data we expect a less precise definition of the boundary condition. Therefore, the problem may derive for the fact that we are looking for a precision that is beyond what is achievable by any method for this combination of model and data.

In that case, it may be that our problem was poorly constrained even with $1/32^{nd}$ pseudo-data. The distance and the position of the data points relative to the boundaries may play on how they are affected by them. Furthermore, some regions are nearly isolated from the rest of the domain, especially in the south-eastern part of the lagoon. This appeared more clearly in the experiments with the different locations of data. We saw that data located in the lagoon provided

the best estimates of boundary conditions, showing that the lagoon zone was more sensitive to variations in the boundary conditions than the deeper oceanic zone.

We emphasize the need for both a good sampling strategy and a good choice of the node positions for reconstructing the function we are trying to approach. Perhaps the hardest part in parameter identifications with regard to determining the boundary conditions is to identify not the value but the shape of those boundary conditions. Indeed, methods using an inverse model (Devenon, 1990) and the methods with empirical orthogonal functions of Vermeulen and Heemink (2006) can provide such a shape, but requires the development of a second numerical model or many iterations of the model before parameters can be determined. With the successive approximations, this shape is determined by the functions used for approximations and the position of the nodes. Hence, both factors should be carefully considered with the physical knowledge of the problem to tackle in mind.

Furthermore, the successive approximation methodology presents a filtering effect on the reconstruction of the shape of the boundary conditions, as is visible in Figures 9 and 10. This effect is inherent to the spline interpolation method and has already been observed by Unser (1999). The filtering effect could also be an advantage, since depending on the number of observations that we dispose or the nature of the problem, we could also not want to deal with small scale variations. In addition, small scale variations can be destabilizing in some applications and therefore filtering them out can be an important point, as was discussed in Sasaki (1970).

4.3. Practical implementation

The implementation of the successive approximation strategy with an algorithm based on an SPSA-like algorithm is not necessary. The pyramidal approach presented in Thevenaz et al. (1998) or the "stepwise approach" presented in Jahns (1966) used other algorithms. However, in our case, the SPSA-like algorithm presented many advantages as it was easy to implement and presented

good convergence properties (Spall, 1998), which are important in geophysics for models that tend to be quite complex. Still, the reduction of the number of variables could open up the way to other algorithms based on perturbations of the parameters, such as genetic algorithms for example. The choice of the gradient descent algorithm would of course be highly dependent on the problem to be tackled.

Our tests showed us some issues with the implementation of successive approximations. As different gradient descent are performed one after another, a robust and versatile tuning of the gradient descent algorithm is important. As already pointed out, there is no reason for the algorithm parameters to remain constant for two successive problems. With some gradient-descent algorithms, the problem of the gain coefficient could be tackled by the use of algorithms that use hessian approximation as step-size. With the SPSA algorithm, there are some existing second order methods (Zhu and Spall, 2002) (Reddy et al., 2016). Some work should also be devoted to the stopping criterion, as it is difficult to estimate the number of iterations necessary for different stages. The number of iterations necessary depends on the adjustments needed at each stage, which it would be difficult to estimate in advance. Some work has been done with the SPSA in order to determine a stopping criterion (Wada and Fujisaki, 2013) but are so far not directly transposable to complex applications.

In a similar way, a line-search procedure was used because we knew the problem was convex and because we were not able to compute a correct step-size with a second order algorithm. However, the use of line-search algorithm is unwieldy and not suited to non-convex problems. Thus, some other methods should be used for non-linear problems, perhaps second order approximations could be more appropriate.

One of the advantages of this method is its computational cost. We saw that the time required for each step of the gradient descent was largely dominated by the model runs. Since this method only requires estimation of the cost function and generation of vectors of parameters, the ratio of the time taken by the model runs to the generation of perturbations by the algorithm is likely to

increase with larger models with more data. The time necessary for parameter identification is dominated by the time of the model runs. This time coud be reduced by parallelizing the model runs of the same iteration.

4.4. Perspectives

Here, we successfully identified boundary conditions for a tidal linear model. The method described in this article could be used directly for other coastal situations or other tidal components with in-situ data instead of doing twin experiments. In addition, we could extend the work presented here to other situations.

In fact, the problem of identifying lateral boundary conditions shares some features with other problems of interest in geophysics. The high-number of parameters and the continuity of the field of interest can also be found in initial conditions or friction coefficient, for example. The methodology of approximation with splines can also be used directly for two-dimensional or even higher-dimensional fields of parameters and therefore the problem addressed here can be seen as a first step towards a more general parameter identification method. As a complement to these advantages, our method do not require an inverse model, owing to the gradient descent algorithm used.

Also on the basis of this gradient descent algorithm, we could identify parameters in non-convex problems or with non-linear models. To perform this identification we should remove some of the adaptations that we made to the algorithm in the present work but those modifications are minor, and non-linear optimization has already been performed with the SPSA algorithm (Spall, 1998) outside the field of oceanography, or even in the field of oceanography by Boutet et al. (2015).

Furthermore, the simplicity of implementation of the method enables us to build more sophisticated tools for the study of the circulation patterns. We can consider the use of the reduced tidal model as a special case of linearization of a more complex circulation model. Hence, we could use this model in a data-assimilation loop for optimizing the tidal boundary conditions for a more

complex model, such as CROCO, for example.

5. Conclusions

We performed, with a linear tidal propagation model, a series of twin experiments and a more realistic experiment. We investigated whether a SPSA-like gradient descent algorithm could be used to identify boundary conditions, as they represented a large number of parameters. We showed it was possible with a special method involving several stages of approximations. We used approximations that have coefficients whose magnitude is close to the magnitude of the function to be approximated. The fact that the magnitude of the coefficients was close to the magnitude of the boundary conditions helped us in setting up the successive approximations used.

We were able to determine the boundary conditions of our tidal model up to a tenth of centimeter for the twin experiments. The novelty here was that we succeeded with tens of effective parameters, and with a methodology that would enable us to go even further.

However, the first try-outs with more realistic data locations and with in-situ data showed us a more contrasted picture. As for any inverse methods, the precision and location of data is crucial and the successive approximations do not change that. The issue that is more specific to this method are the oscillations occuring in the boundary conditions in the realistic experiments. This may be improved by trying other functions for approximations, with p-chip or akima interpolation which reduces the oscillations or by playing with the positions of the nodes.

To conclude, we have developed a method that was able to identify an arbitrary number of parameters and tested it with a tidal model in a lagoon setup. The results were affected both by the noise in data and their sampling, but also by the number of parameters of the approximation used in the method. We saw an effect of filtering smaller scales of the boundary conditions. There is no theoretical drawback to the use of other gradient descent algorithms or even

other optimization methods with the successive approximations method, but there are still some issues relative to its use with reduced numbers of data. The use of approximations and the successive problems solved are fully compatible with any method. However, we had to modify the gain sequence of our gradient descent algorithm to perform efficiently the successive gradient descents.

Future work will focus on the implementation of this method in realistic applications. Specifically, we could use this method for non open-source models or for models which do not have an adjoint. The CROCO model is a good example but hardly the only one, and we are sure that modellers will find many creative applications for the successive approximations method.

Declaration of Competing Interest

The authors declare that they no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This research has been made possible thanks to a PhD thesis funded by the French ministry of Research and Higher Education. This work was done on computing resources provided by the OPLC team of the MIO. The authors would personally like to thank Remi Pages, Melika Baklouti, Denis Bourras and Claire Marc for their support during the drafting of the paper.

References

, 2020. Splines implementation in scipy. `http://pageperso.lif.univ-mrs.fr/~francois.denis/IAAM1/scipy-html-1.0.0/generated/scipy.interpolate.CubicSpline.html`. Accessed: 2020-07-02.

Altaf, M., Heemink, A., Verlaan, M., Hoteit, I., 2011. Simultaneous perturbation stochastic approximation for tidal models. Ocean Dynamics 61, 1093–1105.

Altaf, M., Verlaan, M., Heemink, A., 2012. Efficient identification of uncertain parameters in a large-scale tidal model of the european continental shelf by proper orthogonal decomposition. International journal for numerical methods in fluids 68, 422–450.

Andersen, O.B., Egbert, G.D., Erofeeva, S.Y., Ray, R.D., 2006. Mapping nonlinear shallow-water tides: a look at the past and future. Ocean Dynamics 56, 416–429.

Armijo, L., 1966. Minimization of functions having lipschitz continuous first partial derivatives. Pacific Journal of mathematics 16, 1–3.

Blum, J., Le Dimet, F.X., Navon, I., 2009. Data assimilation for geophysical fluids, in: Handbook of numerical analysis. Elsevier. volume 14, pp. 385–441.

Boutet, M., Lathuilière, C., Hoang, H., Baraille, R., 2015. Bottom friction optimization for a better barotropic tide modelling, in: EGU General Assembly Conference Abstracts.

Chevalier, C., Sous, D., Devenon, J.L., Pagano, M., Rougier, G., Blanchot, J., 2015. Impact of cross-reef water fluxes on lagoon dynamics: a simple parameterization for coral lagoon circulation model, with application to the ouano lagoon, new caledonia. Ocean Dynamics 65, 1509–1534.

Codiga, D.L., 2011. Unified tidal analysis and prediction using the UTide Matlab functions. Graduate School of Oceanography, University of Rhode Island Narragansett, RI.

Csanady, G.T., 1981. Circulation in the coastal ocean, in: Advances in Geophysics. Elsevier. volume 23, pp. 101–183.

De Boor, C., 1978. A practical guide to splines. volume 27. springer-verlag New York.

Devenon, J.L., 1990. Optimal control theory applied to an objective analysis of a tidal current mapping by hf radar. Journal of Atmospheric and Oceanic Technology 7, 269–284.

Hoang, H., Baraille, R., 2011. On efficiency of simultaneous perturbation stochastic approximation method for implementation of an adaptive filter. Computer Techniques and Applications 2, 948–962.

Hoteit, I., Köhl, A., 2006. Efficiency of reduced-order, time-dependent adjoint data assimilation approaches. Journal of oceanography 62, 539–550.

Jahns, H., 1966. A rapid method for obtaining a two-dimensional reservoir description from well pressure response data. Society of Petroleum Engineers Journal 6, 315–327.

James, I., 2002. Modelling pollution dispersion, the ecosystem and water quality in coastal waters: a review. Environmental Modelling & Software 17, 363–385.

Jouon, A., Douillet, P., Ouillon, S., Fraunié, P., 2006. Calculations of hydrodynamic time parameters in a semi-opened coastal zone using a 3d hydrodynamic model. Continental Shelf Research 26, 1395–1415.

Le Provost, C., Genco, M., Lyard, F., Vincent, P., Canceil, P., 1994. Spectroscopy of the world ocean tides from a finite element hydrodynamic model. Journal of Geophysical Research: Oceans 99, 24777–24797.

Lellouche, J.M., Devenon, J.L., Dekeyser, I., 1998. Data assimilation by optimal control method in a 3d coastal oceanic model: The problem of discretization. Journal of Atmospheric and Oceanic Technology 15, 470–481.

Messié, M., Petrenko, A., Doglioli, A.M., Aldebert, C., Martinez, E., Koenig, G., Bonnet, S., Moutin, T., 2020. The delayed island mass effect: How islands can remotely trigger blooms in the oligotrophic ocean. Geophysical Research Letters 47, e2019GL085282.

Ouillon, S., Douillet, P., Lefebvre, J.P., Le Gendre, R., Jouon, A., Bonneton, P., Fernandez, J.M., Chevillon, C., Magand, O., Lefèvre, J., et al., 2010. Circulation and suspended sediment transport in a coral reef lagoon: The south-west lagoon of new caledonia. Marine pollution bulletin 61, 269–296.

Reddy, D.S.K., Prashanth, L., Bhatnagar, S., 2016. Improved hessian estimation for adaptive random directions stochastic approximation, in: 2016 IEEE 55th Conference on Decision and Control (CDC), IEEE. pp. 3682–3687.

Sasaki, Y., 1970. Numerical variational analysis formulated under the constraints as determined by longwave equations and a low-pass filter. Mon. Wea. Rev 98, 884–898.

Sous, D., Chevalier, C., Devenon, J.L., Blanchot, J., Pagano, M., 2017. Circulation patterns in a channel reef-lagoon system, ouano lagoon, new caledonia. Estuarine, Coastal and Shelf Science 196, 315–330.

Spall, J., 1998. An overview of the simultaneous perturbation method for efficient optimization. Johns Hopkins apl technical digest 19, 482–492.

Sutskever, I., Martens, J., Dahl, G., Hinton, G., 2013. On the importance of initialization and momentum in deep learning, in: International conference on machine learning, pp. 1139–1147.

Taillandier, V., Echevin, V., Mortier, L., Devenon, J.L., 2004. Controlling boundary conditions with a four-dimensional variational data-assimilation method in a non-stratified open coastal model. Ocean Dynamics 54, 284–298.

Thevenaz, P., Ruttimann, U., Unser, M., 1998. A pyramid approach to subpixel registration based on intensity. IEEE transactions on image processing 7, 27–41.

Tympakianaki, A., Koutsopoulos, H., Jenelius, E., 2015. c-spsa: Cluster-wise simultaneous perturbation stochastic approximation algorithm and its application to dynamic origin–destination matrix estimation. Transportation Research Part C: Emerging Technologies 55, 231–245.

Unser, M., 1999. Splines: A perfect fit for signal and image processing. IEEE Signal processing magazine 16, 22–38.

Vermeulen, P., Heemink, A., 2006. Model-reduced variational data assimilation. Monthly weather review 134, 2888–2899.

Wada, T., Fujisaki, Y., 2013. A stopping rule for simultaneous perturbation stochastic approximation, in: 2013 European Control Conference (ECC), IEEE. pp. 644–649.

Zhu, X., Spall, J.C., 2002. A modified second-order spsa optimization algorithm for finite samples. International Journal of Adaptive Control and Signal Processing 16, 397–409.

Appendix A. Reduced tidal model

Development of the model

For the numerical modelling, we can start with the shallow water equations. It seems justified because previous reports of the patterns of circulation in the lagoon did not show stratification (Sous et al. (2017), Chevalier et al. (2015)).

$$\begin{cases} \frac{\partial \overrightarrow{u}}{\partial t} + (\overrightarrow{u} \cdot \overrightarrow{\nabla})\overrightarrow{u} + \overrightarrow{f} \wedge \overrightarrow{u} + \overrightarrow{P} = \frac{K\overrightarrow{u}}{H+\eta} & \text{Momentum conservation} \\ \frac{\partial \eta}{\partial t} + \overrightarrow{\nabla}((H+\eta)\overrightarrow{u}) = 0 & \text{Mass conservation} \end{cases} \tag{A.1}$$

Here, $\overrightarrow{u}$ is the velocity in $m.s^{-1}$, H is the local depth in m, η is the sea surface elevation in m, f is the Coriolis frequency at the Ouano lagoon in s^{-1} and K is the local friction coefficient in $m.s^{-1}$. The linear form of the friction can be used for an average over a tidal cycle (Csanady (1981), Devenon (1990)).

We can simplify those equations with a few observations in the lagoon. First, the depth in the lagoon is a few meters, and the tidal amplitude is of the order of 0.4 m for the M2 component. We expect velocities of the order of a most 0.2 $m.s^{-1}$ in the passes, and see that the frequency of the M2 component is of the order of the Coriolis frequency of around 5.10^{-4} s^{-1}. In the lagoon, we expect strong friction coefficients and can work with an estimate of K of 10^{-2}

$m.s^{-1}$. If we add to that we expect typical flow length of a few kilometers, we can estimate the magnitude of the different terms.

$$\begin{cases} \overbrace{\frac{\partial \overrightarrow{u}}{\partial t}}^{10^{-5} m.s^{-2}} + \overbrace{(\overrightarrow{u} \cdot \overrightarrow{\nabla})\overrightarrow{u}}^{10^{-6} m.s^{-2}} + \overbrace{\overrightarrow{f} \wedge \overrightarrow{u}}^{10^{-5} m.s^{-2}} + \overbrace{\overrightarrow{P}}^{?m.s^{-2}} = \overbrace{\frac{K\overrightarrow{u}}{H}}^{10^{-4} m.s^{-2}} & \text{Momentum conservation} \\ \underbrace{\frac{\partial \eta}{\partial t}}_{10^{-5} m.s^{-1}} + \underbrace{\overrightarrow{\nabla}(H\overrightarrow{u})}_{10^{-3} m.s^{-1}} = 0 & \text{Mass conservation} \end{cases} \tag{A.2}$$

We see that the only remaining term is the pressure gradient that must be at equilibrium with the friction. And this equilibrium gives a linear system at first order. In the passes and over the reef, where velocities are more important and depth reduced, non-linear terms can be bigger. However, the advection and continuity terms generate compound tides and overtides and do not affect the M2 coefficient when no constant current is present Andersen et al. (2006) ! The linear friction term that we used was fitted for the M2 circulation but does not allow for the generations of overtides and compound tides. We therefore can neglect those interactions at first order. Therefore, we can end up with the following system, that is close to those used previously (Devenon (1990),Le Provost et al. (1994)).

$$\begin{cases} (-i\omega_k - r/H)\overrightarrow{U}_k + g\overrightarrow{\nabla}\eta_k = \overrightarrow{0} & \text{Momentum conservation} \\ -i\omega_k \eta_k + \overrightarrow{\nabla} \cdot (H\overrightarrow{U}_k) = 0 & \text{Mass conservation} \end{cases} \tag{A.3}$$

Here k is the index corresponding to a given tidal wave, M2 in this case, ω_k denotes the frequency of the considered tidal wave in s^{-1}. Note that we have kept the acceleration term and the Coriolis term which were of the same order of magnitude and two orders of magnitude smaller than the friction and pressure gradient terms. However, the acceleration term was kept because it is required for the oscillatory dynamics of the tide.

The usual argument for considering or not the Coriolis term is the Rossby number. If we consider the tidal velocities of 0.10 $m.s^{-1}$, characteristic length

of 5 km and a Coriolis of the order of factor of 5.10^{-5} s^{-1} (Sous et al., 2017), we have a Rossby number of 0.4. But the Rossby number is mainly used for comparing the advection terms and the Coriolis term in open sea. For tides in coastal situation, the advection terms are neglected because they neglect compound and overtides and do not contribute directly to the M2 component. It is more appropriate to compare the Rossby radius of deformation, which here would be :

$$\sqrt{gh}/f = \sqrt{10m.s^{-2} * 10m}/5.10^{-5}s^{-1} = 200km \tag{A.4}$$

If we compare this value with the 30 km of the largest dimension of the lagoon, we can consider that the corrections brought by the Coriolis term would be negligible at first order. We can then neglect it and with some algebraic manipulations, we obtain a complex Helmholtz equation for the amplitude. And the velocities can be deduced from the first derivative of the sea level. The associated equation is the equation A.5.

$$\begin{cases} i\omega_k\eta_k + \overrightarrow{\nabla}(\beta_k\overrightarrow{\nabla}\eta_k) = 0 \\ \beta_k = \frac{gH}{(r/H - i\omega_k)} \end{cases} \tag{A.5}$$

To discretize this equation, we used a second order finite-difference centered scheme, the first order Neumann condition was implemented directly in the solving matrix. Finally, we solved it with an iterative Jacobi method with a thousand iterations for ease of implementation with our programming language. The number of iterations was fixed because variations were below a tenth of millimeters after a thousand iterations, but the speed of convergence depended on the boundary condition.

The main goal here was to provide a model that could be run fast, so that we could do the number of tests necessary for our method. We still wanted something accurate enough to reproduce qualitatively what we observe with the complete non-linear models such as CROCO, and that was consistent with

the precision of the pressure sensors that were used in the 2016 campaign in the lagoon (Sous et al., 2017), with discrepancies of at most a few centimeters and a phase lag of a few degrees. However, we expected the precision to be lowest in the shallower parts of the lagoons because that would be the most sensitive to non-linear effects, and therefore we would expect our model to be less efficient in those parts.

Comparison with a reference model

We therefore compared the field of η obtained with this tidal model with the output of the fully non-linear CROCO (available on the site `http://www.croco-ocean.org`) model in 2D for the M2 tidal component in a case that corresponded to the model outputs presented in Chevalier et al. (2015). The output held less than 10% discrepancies with the CROCO output. The shallowest region exhibited wider discrepancies in phase, as is presented in Figure A.16. Those discrepancies may be attributed to the friction in those regions, which was set as quadratic in a previous modelling effort Chevalier et al. (2015) but has been linearized in the tidal model, and to the challenging handling of boundary conditions and of advection in those regions.

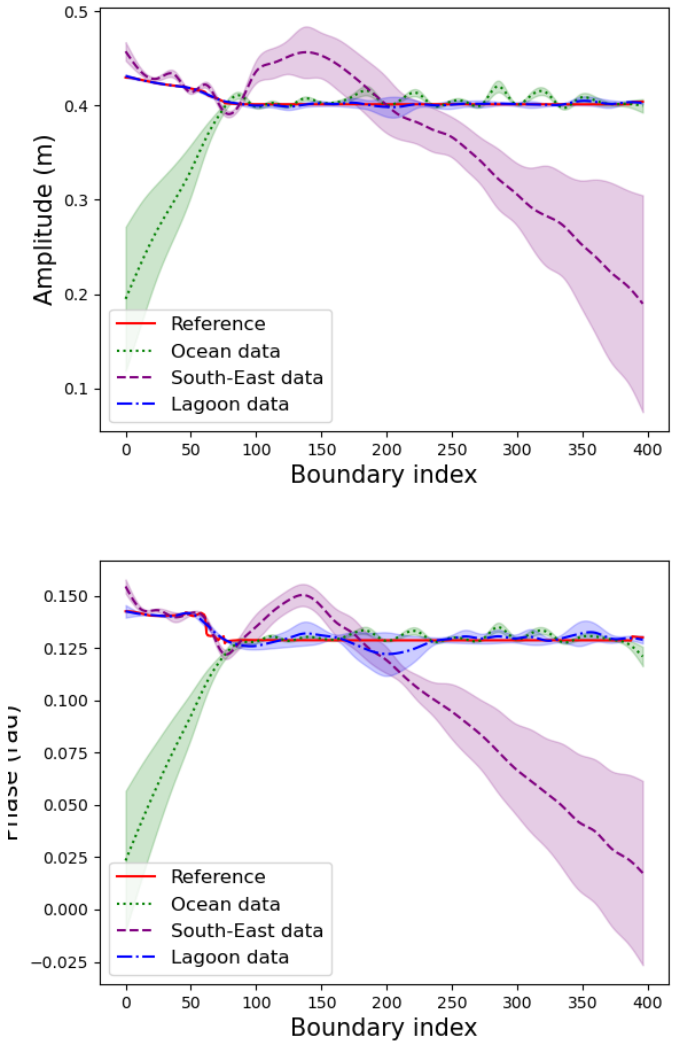

Figure 12: Boundary conditions determined with the data located in specific parts of the lagoon. The lines represent mean values for ten determinations of boundary conditions while the shaded zone represents one standard deviation from that mean value.

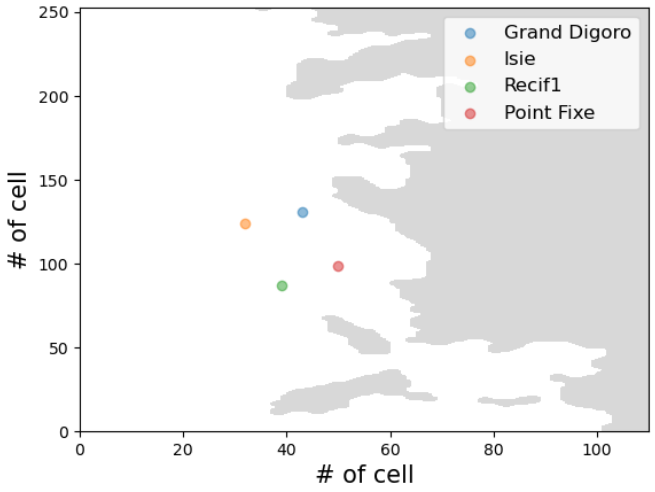

Figure 13: Position of the sensors for the real data assimilation experiment.

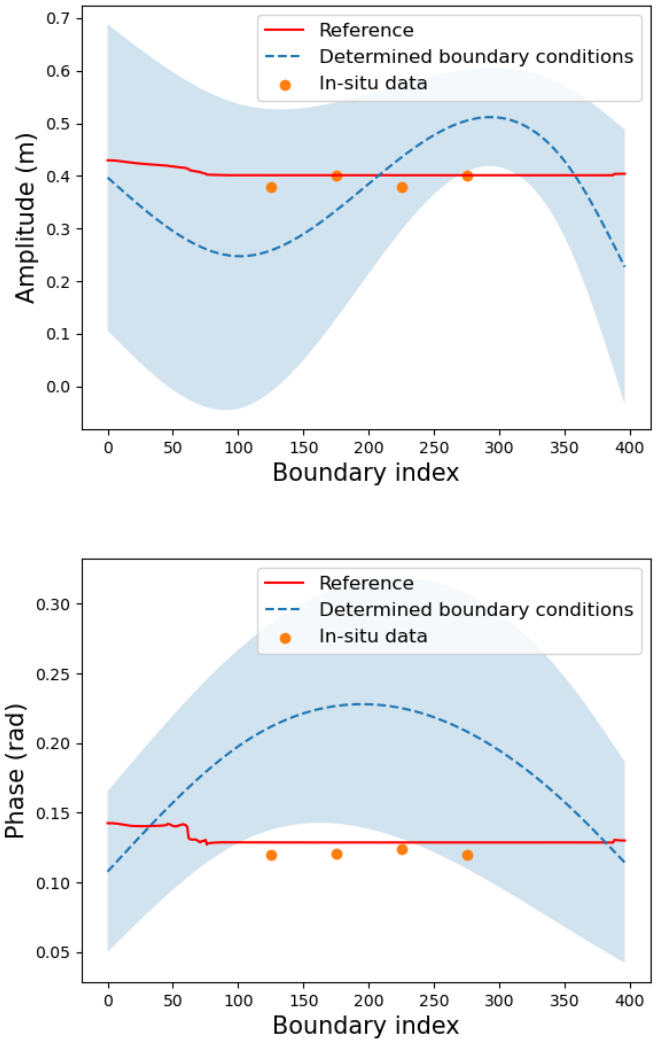

Figure 14: Boundary conditions determined with the in-situ data only. The reference curves are here to show what would be the first order expected results. The dots correspond to the data of the sensors, but their positions are arbitrary.

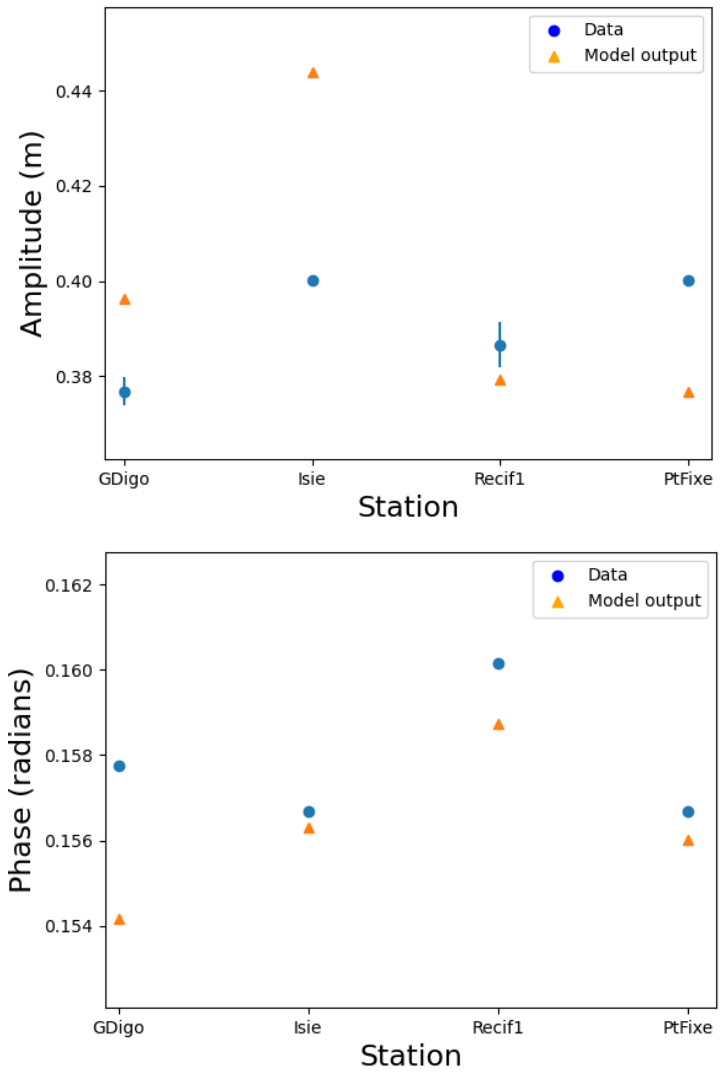

Figure 15: Comparison of the M2 tide amplitude and phase at the different stations for the data and the outputs of the reduced model.

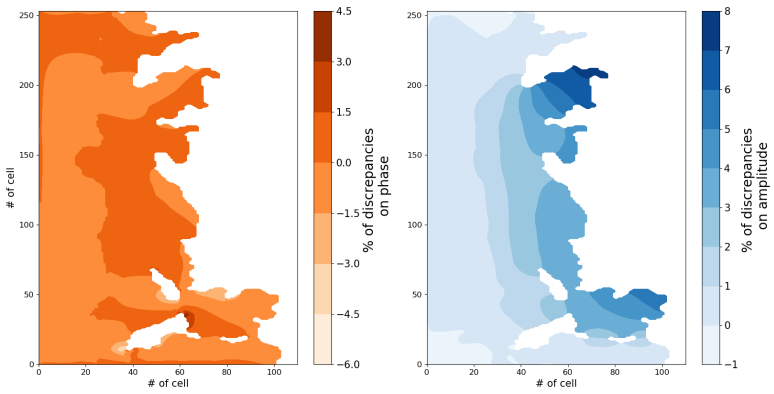

Figure A.16: Differences in phase (left) and amplitude (right) in percentage of the M2 component of the tide determined in the Ouano lagoon by the CROCO model and the Serpent de Mer model, with the same boundary conditions