INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

# Programming Practical in Earth sciences : River bathymetry estimation from surface measurements (altimetry-like). Part 2.

Spring 2024


FIGURE 1 – The diffusive wave equations are employed in some labs to model Amazonian rivers dynamics. Image extracted from Wikipedia and


INSA Toulouse, Applied Mathematics department
Prof. J. Monnier (Jerome.Monnier@insa-toulouse.fr)


PP initially developed in collaboration with T. Malou, PhD INSA-IMT-CLS 2018-21

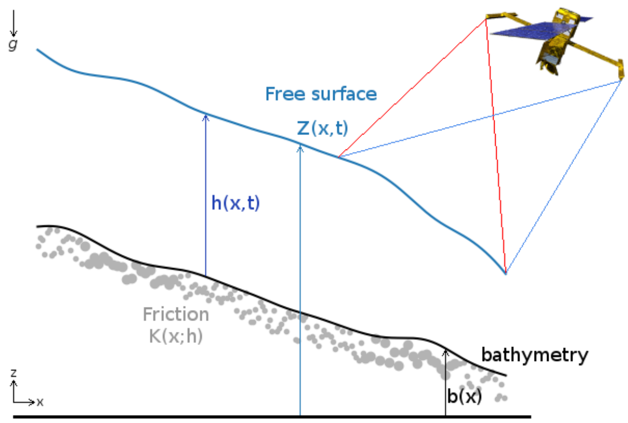


FIGURE 2 – An inverse problem arising in spatial hydrology : estimate the unobserved bathymetry $b(x)$ from altimetry measurements. Measurements are the water surface elevation $H(x,t)$ and the river width $W(x,t)$.

# Table des matières



# 1 The direct flow model and the inverse problem

We here consider the same inverse problem as previously, cf the document Part 1, however based on a non-linear flow model.

The model equation reads :

$$\boxed{-\Lambda_{ref}(\partial_x H_{ref}, b; H)(x)\frac{\partial^2 H}{\partial x^2}(x) + \frac{\partial H}{\partial x}(x) = \frac{\partial b}{\partial x}(x) \quad \forall x \in [0, L]} \tag{1}$$

with Dirichlet boundary conditions, and

$$\Lambda_{ref}(\partial_x H_{ref}, b; H) = \frac{3}{10}\frac{(H-b)(x)}{|\partial_x H_{ref}(x)|}$$

where $\partial_x H^{obs}$ denotes a value of the slope of the water surface elevation deduced from observations ; it is given.

Recall that the considered inverse problem reads : *given some measurements $H^{obs}(x,t)$, estimate the river bathymetry $b(x)$.*

The objectif function measures the discrepancy between the computed water surface elevation and the observation. It reads :

$$J^{obs}(H) = \sum_{m=1}^{M}\left(H(x_m) - H_m^{obs}\right)^2 \tag{2}$$

at $x_m$ where observations are available.

The total cost function $j$ reads :

$$\boxed{j_\alpha(b) = J_\alpha\left(H^b; b\right) = J^{obs}\left(H^b\right) + \alpha_{reg} J^{reg}(b)} \tag{3}$$

where $H^b$ is the (unique) solution of the flow model, given $b$.

The principle of the VDA method is to solve the optimization problem :

$$\boxed{b^*_\alpha = arg \min_{b \in B} j_\alpha(b)} \tag{4}$$

Therefore providing the/an optimal bathymetry $b^*$.

# 2 Your tasks

We here seek to adapt the adjoint method to solve the inverse problem by VDA to the non-linear model above.

## 2.1 Write & code the optimality system equations

— Write the optimality system of the problem, in particular the adjoint equations and the gradient expression.

— Enrich the provided code to solve the inverse problem using the adjoint method.

— Code validation.
Analyse the validation curves of the gradient. Is the computed gradient valid ?
In other words, can you go further by performing gradient-based inversions ?

## 2.2 Inverse problem : inferences of $b(x)$

### 2.2.1 Step 1. Make your own twin experiments and numerical investigations

1. Set up your own geometry case, with quite low observations density, moreover uncertain.
2. Perform the complete inverse problem resolution (VDA process) from different priors.
   Comment, analyse a few solutions your are able to obtain.
3. Consider the case where one (1) in-situ measurement of $h$ is given : $b^*$ is known at one (1) location $x_0$.
   Investigate two different ways to impose this information. Analyse.

   *Important recall.* After each inverse computations, and before analyzing the "physical" results, you must analyse the code numerical outputs : the stopping criteria, the plots cost function vs iterations, control variable vs iterations, and the gradient value vs iterations.

### 2.2.2 Step 2. Change of metrics using a change of variable

You will investigate the results if adopting the change of variable detailed in Appendix below.
All the required routines are already coded, see below and the ReadMe file.

1. - Activate the change of variable in your computations.
   - Validate the gradient (gradient test).

2. Given a background value $b_b$, given $C$ one of the two proposed covariance kernel, analyse the results by comparing the minimization behavior in terms of convergence speed and accuracy for different values of $L$ (e.g. $L = n\ dx$) and by comparing with the case $C = Id$.

**Questions :**

-In the end, is the obtained VDA process more robust and/or faster than if not using covariances-based metrics ?
-How the values of $C$, $L$ and $\sigma$ influence the computed solution $b^*$ ?
You will detail your answers.

## Supplementary material. Regularization terms based on covariance kernels and change of variable

At the previous stage, you have highlighted that the inverse problem is badly conditioned : reaching the correct minimum independently of the first guess value is hard task. It seems that the functional $J$ presents relatively "flat valleys". We investigate here define different regularization terms that "convexify" the cost function which are based on covariance operators.
For simple linear diffusive equations, the resulting $J_{reg}$ terms are physically consistent with the flow model.

**Regularization term defined from a background value $b_b$ and change of variable** Introducing a background value $b_b$ not too far from the sough optimal value $b^*$ is an excellent way to convexify the cost function therefore to constraint the inverse problem solution. Given a prior $b_b$, we set : $J_{reg}(b) = \|b - b_b\|^2$. However, (the) good background values $b_b$ are a-priori unknown!...

The term $J_{reg}$ above can be defined from a norm characterized by a covariance function $\|\Box\|_{C^{-1}}$, instead of the classical $L^2$ norm. This consists to set :

$$J_{reg}(u) = \|u - u_b\|^2_{C^{-1}} \tag{5}$$

with $\|\Box\|^2_{C^{-1}} = (C^{-1} \cdot \Box, \Box)_2 = (C^{-1/2}\Box, C^{-1/2}\Box)_2 = \|C^{-1/2}\Box\|^2_2$.

Next, instead of directly computing this $C^{-1}$-norm, one naturally introduces the following change of variable :

$$l = C^{-\frac{1}{2}}(u - u_b) \tag{6}$$

However, in practice, the computation of $C^{\pm\frac{1}{2}}$ is CPU-time consuming. A good alternative is to define the following change of variable :

$$\boxed{l = {L_C}^{-1}(u - u_b) \text{ with } C = L_C L_C^T} \tag{7}$$

Indeed, we have :

$$\begin{aligned}\|x\|^2_{C^{-1}} &= (C^{-1}x, x)_2 = ((L_C L_C^T)^{-1}x, x)_2 \\ &= (L_C^{-T} L^{-1}x, x)_2 = (L_C^{-1}x, L_C^{-1}x)_2 = \|L_C^{-1}x\|^2_2\end{aligned}$$

Applied to $x = (u - u_b)$, we get : $\boxed{\|u - u_b\|^2_{C^{-1}} = \|l\|^2_2}$.

The computation of the Cholesky decomposition of $C$ is easy and low CPU time-consuming compared to $C^{\frac{1}{2}}$.

Next, we define the cost function $g$ as :

$$\boxed{g(l) = j(u)} \tag{8}$$

The differential of these two cost functions are related by (easy calculation) :

$$\boxed{\nabla g(l) = L_C^T \ \nabla j(u)} \tag{9}$$

As a consequence, this change of variable change the descent directions along the minimization process.

Another point of view is the following : it acts as a preconditionner of the first order necessary optimality condition. The introduction of $C$ may (hopefully...) provide a faster and/or more robust convergence process.

Finally, the optimization problem is solved using the inverse change of variable : $u = L_C l + u_b$.

**Classical covariance functions** The commonly considered covariance functions $C$ are the following two.

$$C(x,x') = \sigma^2 \exp\left(-\frac{1}{2}\frac{(|x-x'|)^2}{L^2}\right) \text{ and } C(x,x') = \sigma \exp\left(-\frac{1}{2}\frac{(|x-x'|)}{L}\right) \tag{10}$$

In the Gaussian case, the parameter $L^2$ may be viewed as a standard deviation.
In the negative exponential case, $L$ represents a characterize length scale setting the regularization scale ; it is a correlation length scale.
In both cases, $L$ and $\sigma$ are priors of the inverse problem.

The obtained optimal solution $u^*$ depend on these priors $L$ and $\sigma$.
These two parameters may be deduced from physical or mathematical arguments (difficult task) or may be experimentally set up.

**Equivalence between $C^{-1}$-norms and regularization terms** Let us consider the negative exponential covariance kernel $C(x,x') = \sigma^2 \exp\left(-\frac{1}{2}\frac{(|x-x'|)}{L}\right)$.
The covariance operator $\mathbf{C}$ corresponding to this integral kernel is defined by :

$$\mathbf{C}(x) : \hat{k} \mapsto \mathbf{C}\hat{u} = u = \int_{x_1}^{x_2} C(x,x')\hat{u}(x')\ dx' \qquad \forall x \in [x_1, x_2]$$

Then, one can show for example the following results (see e.g. A. Tarantola book Chapter 7).

$$\|u\|_2^2 \approx \frac{1}{2\sigma^2}\left(\frac{1}{L}\int_{x_1}^{x_2} \hat{u}^2(x)\ dx + L\int_{x_1}^{x_2} (\partial_x \hat{u}(x))^2\ dx\right)$$

That is the $C^{-1}$-norm above is equivalent to introduce a second order regularization term weighted by coefficients depending on $L$.
A very few equivalences as above have been stated. Moreover, the expression of the equivalent regularization term depends on the dimension (here 1d).

**Coding aspects** The Gaussian and decreasing exponential covariance functions above are already coded (routines in the file class vda.py). The change of variable is coded too (in the file class vda.py).