Published in Image Processing On Line on 2018–09–03.
Submitted on 2017–02–27, accepted on 2018–05–25.
ISSN 2105–1232 © 2018 IPOL & the authors CC–BY–NC–SA
This article is available online with supplementary materials,
software, datasets and online demo at
https://doi.org/10.5201/ipol.2018.205

Numerical Simulation of Landscape Evolution Models

Marc Lebrun[1], Miguel Colom[1], Jérôme Darbon[2], Jean-Michel Morel[1]

[1] CMLA, ENS Paris-Saclay, France ({lebrun, colom, morel}@cmla.ens-cachan.fr),
[2] Brown University, USA (jerome_darbon@brown.edu)

Communicated by Miguel Colom *Demo edited by* Miguel Colom

Abstract

This paper gives the complete numerical schemes implementing the main physical laws proposed in landscape evolution models (LEMs). These laws can be modeled by a system of three partial differential equations governing water runoff, stream incision, hill slope evolution and sedimentation. The goal of the presented algorithm, code and online demo is to be able to test these equations on digital elevation models (DEMs) of any resolution, and to illustrate its potential to simulate the fine structure of the river network, and to understand the landscape morphology and its causes. The equations simulate plausible evolutions. We illustrate experiments on DEMs of several sites, including one site, La Réunion where the DEM is given at three different resolutions: the SRTM resolution (90m), and then 12m and 4m on DEMs derived from several Pléiades pairs. Other many DEMs are proposed in the online demo, which allows to upload and tests other DEMs.

Source Code

The reviewed source code and documentation for this algorithm are available from the web page of this article[1].

Supplementary Material

Please try on any DEM encoded as a gray level image the online demo (Landscape Evolution Model) http://ipolcore.ipol.im/demo/clientApp/demo.html?id=205. It controls landscape evolution (water runoff, erosion, sedimentation) depending on four main parameters (rain, erosion, creep, and sedimentation).

Keywords: landscape evolution model; partial differential equations; river networks; conservation laws; stream incision law; detachment-limited and transport-limited erosion; Pléiades

[1] https://doi.org/10.5201/ipol.2018.205

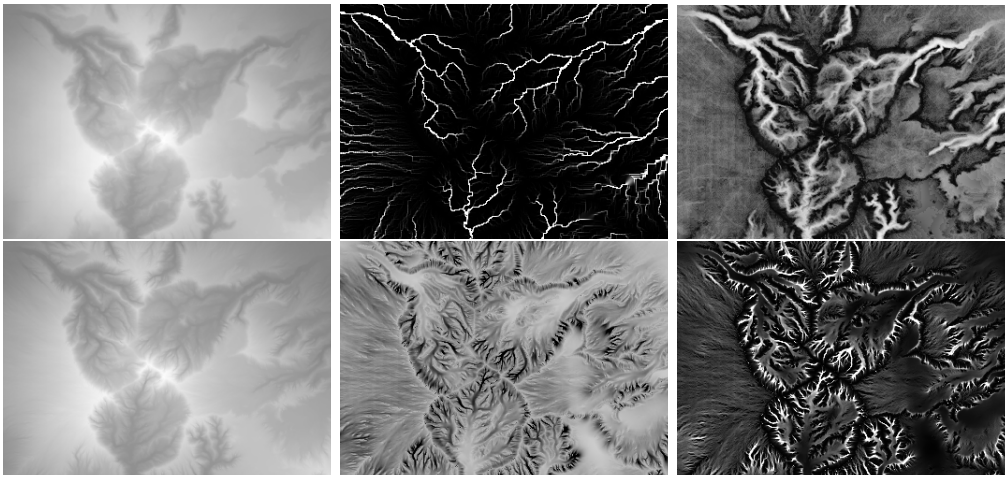

Figure 1: This simulation, on a 90 meter resolution SRTM model of Piton des Neiges, La Réunion, shows all state variables that will be displayed in our numerical LEM. Top left: initial DEM. Top middle: river network obtained by the classic method which identifies it with the drainage basin area. Top right: the steady state value under uniform rain of the water elevation $\theta(x, y)$ when water runs on the landscape without any erosion. Bottom row: evolution for parameters rain $r = 10$, erosion speed $\epsilon = 0.1$, creep $c = 0.1$, after removal by erosion of 5% of the landscape. Bottom left: final landscape. Bottom middle: last evolution of the landscape (landscape derivative) just before stopping, where black denotes large values and white small values. Bottom right: map $\lambda(x, y)$ of the sediment load in water at the beginning of the erosion process.

1 Introduction

The evolution of landscape morphology depends on many factors. Some are tectonic and geologic and aim at explaining the composition of ground layers and their deformation. Yet the actual evolution of landscape on a smaller time and space scale is driven by the conjugated effects of erosion, sedimentation, chemical weathering, creep, tectonic motions, etc. Erosion (runoff) is the removal of sediment from the land surface by a fluid agent such as water, ice or air. Sedimentation is the converse process, in which sediment from the fluid mixture settles onto the land surface. During storm events, rainfall increases the erosion and sedimentation activity. By estimating the amount of erosion and sedimentation, an estimate for the dynamics of the landscape can be obtained. This effect can be complemented by a terrain smoothing effect called *creep*, whose description and causes are multiple (water splash, chemical and thermal weathering, vegetation, gravitational flow of soft soil). All of these effects can be modeled by a *landscape evolution model (LEM)*, summarized in a few partial differential equations. The numerical simulation of a landscape evolution model requires a thorough knowledge of the Earth morphology, at the highest possible resolution. To that effect, increasingly accurate digital elevation models (DEMs) exist. To be useful they must be complete, namely cover large pieces of the landscape, including whole basins, islands or continents. Simulating landscape evolution has several goals. One of them is to identify the key parameters that drive to an observed morphology. Another one might be to restore DEMs by simulating runoff and erosion. A third and very important goal is to detect the drainage network at all scales and, given rainfall data, to evaluate the water flux at all points of the river network. Clearly, flood simulation is also a valid goal given an accurate DEM.

Landscape evolution modeling based on photographs goes back to [14]. The observation of real landscapes and photographs and clever qualitative reasoning led [13] to establish a mix of quantitative and qualitative principles governing all landscapes. The first mathematical explanation of the convexity of hilltops, ascribed to creep, is attributable to [8]. Ever since the Gilbert's analysis, landscape evolution models have involved mainly two competing factors: on the higher slopes, where water currents are weak and dispersive, soil creep dominates and the profiles are convex. On lower slopes, water flow concentrates and profiles become concave leading to the formation of valleys.

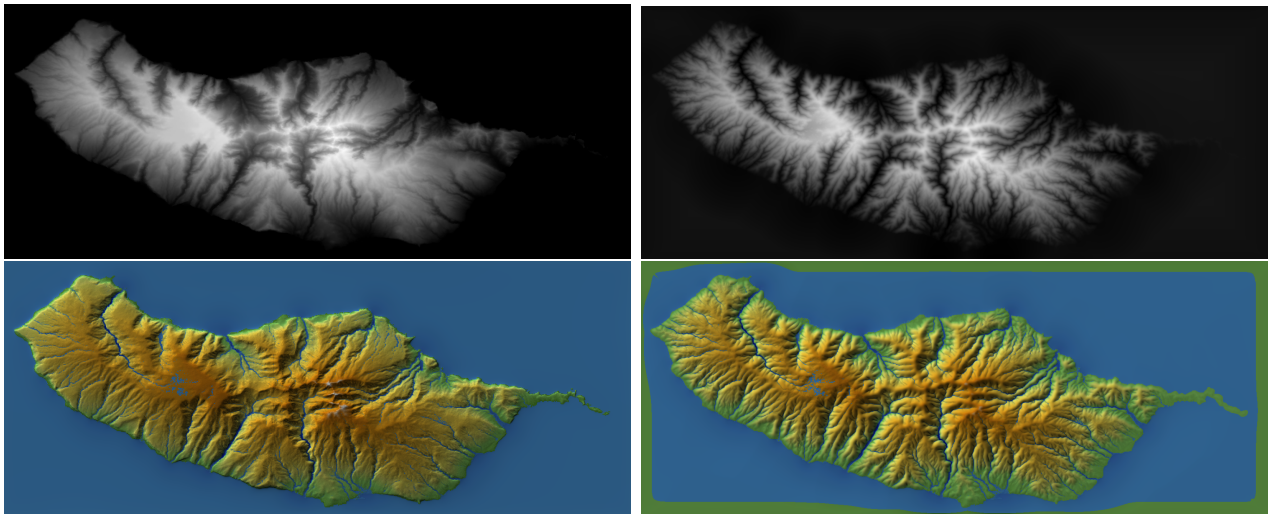

Figure 2: The simulation of the correct equations with the right few parameters should maintain a stable morphology (on stable landscapes!). Here: Madeira, percentage removed: $P = 40\%$, $c = 30$, $e = 2$, $s = 0.2$, $\mathbf{m} = 0.4$, $\mathbf{n} = 0.8$. From top to bottom and left to right, initial landscape, evolved landscape, and their respective topographic renderings. Uplift has compensated the 40% mass reduction. This experiment illustrates (visually) that the morphology of the highland under the complex competition of erosion, sedimentation and creep has not altered significantly, thus confirming the validity of the laws.

We base our equations and numerical experiments on a recent LEM derived through a simplification of a rich list of complex numerical LEMs and software proposed in the past twenty years. The prominent ones are LEGS [21], SIGNUM [22], SIBERIA, CAESAR [15], CASCADE [1], DELIM [16], ZSCAPE, LANDSAP [18], GOLEM [24], APERO [3], EROS [6], and Fastscape [2]. The aforementioned numerical LEMs and software contain dozens and sometimes hundreds of landscape specific parameters. Yet an exhaustive numerical exploration cannot identify more than four or five independent parameters. The model which we follow is introduced in [4] and precised in [5], where it is demonstrated that most features of previous numerical LEMs can be summarized in a system of three partial differential equations governing water runoff, erosion and sedimentation. We shall recall these equations in the next section. The LEM which we shall use involves five parameters. The numerical LEM yields for each initial DEM $h(x, y, 0)$ and each set of parameters an evolved DEM $h(x, y, t)$ under the conjugate effects of erosion, sedimentation and creep. It also yields the final maps of the water height $\theta(x, y, t)$ and of the sediment load $\lambda(x, y, t)$ for each given steady rain rate r. Figure 1 shows such a simulation on an SRTM DEM of the central peak of La Réunion, the Piton des Neiges. It was processed by the test system of three equations (2)-(4) described in Section 2.

The validity of a such a numerical program is sustainable if one can show that a landscape evolution model indeed gives a correct account of landscape evolution on various geological time ranges. Yet, this would require the observation of DEMs on geological time scales incommensurate with human time scales. Fortunately, we shall show in this paper that a simple (but costly) numerical procedure can deliver a sound estimate of the landscape's parameters. The supervised procedure explores potential and reasonable parameter sets, and evolves for each one the landscape until a fixed percentage of the landscape has been scraped and transported by the erosion-sedimentation process. If the landscape morphology is stable and if the right parameters have been found, one can expect that the evolved landscape maintains its morphology (same valleys, similar slopes, etc.). Evaluating that the morphology is stable requires an adequate criterion. A tentative norm for such morphology stability measurements is the Sobolev semi-norm $\int |\nabla(h(x, y, t) - h(x, y, 0))| dxdy$ where $h(x, y, t)$ is the DEM elevation at time t and $h(x, y, 0)$ is the initial landscape. This formula is proposed in Section 5.0.2 describing our automatic parameter estimation method.

With our numerical facility we found the right parameter set on the Madeira island, as illustrated in Figure 2. On a Madeira's SRTM DEM, we tested our three equations model with many different four-parameter sets, until one was found, for which the landscape morphology was stable even after 40% of the emerged ground had been eroded and transported away.

As a teaser we also show numerical experiments on DEMs constructed from images obtained with the Pléiades Earth observation satellite of a very young unstable volcanic landscape, the La Réunion island. In this very recent landscape resulting from successive eruptions, some slopes are still straight and devoid of eroded structure. Yet basins are in formation. We found sets of parameters for which the existing basins seem to extend in a natural way while maintaining their structure. Figure 13 displays such plausible evolution toward a stable landscape, mainly depending on the creep/rain/erosion rate balance.

Plan of the paper Section 2 describes the three equations of the landscape evolution model under consideration. Section 3 focuses on the numerical implementation of the method. The main algorithm is summarized in Section 3.1. Section 3.2 details the water initialization. The core of the algorithm implementing the three main equations of Section 2 is detailed in Section 3.3. Section 4 describes several significant improvements of the experimental setup: Section 4.1 addresses the definition of the right landscape visualization tools and Section 4.2 presents a numerical acceleration tool based on a multiscale formulation, speeding up considerably the initialization of the water runoff scheme. Section 5 describes the three different experimental modes proposed in the code and online demo: 1) the LEM (Landscape Evolution Model, 5.0.1) that corresponds exactly to the algorithms described in Section 3, 2) the LAPE (Landscape Automatic Parameters Estimation, 5.0.2) that tries to automatically estimate the parameter values for a given landscape in such a way that this landscape remains close in shape to the original after the evolution, and 3) the LS (Landscape Synthesis, Section 5.0.3) which purpose is to recreate artificial landscapes from scratch that look like natural ones. This section ends in subSection 5.2 with two experiments for a comparative study of numerical LEM simulation on the same landscape, La Réunion, at three very different resolutions.

2 The Main Landscape Evolution Equations

The numerical LEM used here was introduced in [4] and extended in [5]. In the forthcoming equations, lowercase letters denote functions of the landscape depending on two geographic coordinates (x, y) and on time t. The function $\theta(x, y, t)$ denotes the water height at (x, y) and time t, and similarly $h(x, y, t)$ is the bedrock surface elevation. For a sake of simplicity, we shall in general omit the triplet (x, y, t) in the equations. We denote by $\mathbf{h} = h + \theta$ the landscape altitude (land surface elevation plus water), $|\nabla(h+\theta)|$ the landscape slope, λ the sediment load in water, so that $\frac{\lambda}{\theta}$ is the sediment density in water. Finally $\mathbf{v}$ is the water velocity. (All of these state variables depend on (x, y, t).) $\nabla\phi$ denotes the gradient of a scalar variable $\phi(x, y)$, $\Delta\phi$ its Laplacian, and $\nabla \cdot \mathbf{v}$ the divergence of a vector field $\mathbf{v}(x, y)$. There are also landscape specific parameters in the equations, that for simplicity we assume constant in time and space on each landscape. These parameters, that can be tuned to find the right set, are c the creep rate, ϵ the erosion rate, r the rain rate, s the sedimentation rate and $\mathbf{m}$, $\mathbf{n}$ the erosion exponents. Altogether, the PDE model depends on six parameters. Among them, only two, $\mathbf{m}$ and $\mathbf{n}$ are structural parameters. All others can be considered as factual parameters. Indeed e, the erosion rate, depends on the ground's hardness and is of course (x, y) dependent. Taking it constant as we do may be an abusive simplification on many landscapes. Similarly, the average rainfall r is also ideally (x, y) dependent, and so is c that may depend on the soil's nature. The main issue of such a model, though, is to identify $\mathbf{m}$ and $\mathbf{n}$, that would be universal for some authors or depend on the particular landscape for others.

Let us now detail the equations. We picked the simplest possible version for the generalized Gauckler-Manning, Saint-Venant law [12] so that the water velocity is simply the negative of the slope,

$$\mathbf{v} = -\nabla(h+\theta). \tag{1}$$

There is no simpler way to express the common sense observation that water runs downs the slope. The system used in all experiments governing the evolution of the landscape elevation h, the water depth θ and the sediment load λ in water is composed of three equations:

1. the water conservation and transport law

$$\frac{\partial\theta}{\partial t} = \nabla\cdot(\theta\nabla(h+\theta)) + r; \tag{2}$$

2. the landscape evolution equation with creep (first term on the right hand of the equation), the incision law (second term), and sediment deposition (last term)

$$\frac{\partial h}{\partial t} = c\Delta h - \epsilon\theta^{\mathbf{m}}|\nabla(h+\theta)|^{\mathbf{n}} + s\frac{\lambda}{\theta}; \tag{3}$$

3. and the conservation of sediment law,

$$\frac{\partial\lambda}{\partial t} = \nabla\cdot(\lambda\nabla(h+\theta)) + \epsilon\theta^{\mathbf{m}}|\nabla(h+\theta)|^{\mathbf{n}} - s\frac{\lambda}{\theta}. \tag{4}$$

Thus simulating this system depends on six parameters, the erosion exponents $\mathbf{m}$ and $\mathbf{n}$, the amount of rain r, the creep rate c, the sedimentation rate s, and the erosion speed ϵ. A stopping time must also be specified and we decided to fix it as the percentage of DEM erosion, namely the ratio p of the average eroded elevation in the DEM to the initial average elevation above its minimal level. One may explore values from 5% to 30% that make the evolution visually conspicuous. The typical values for the exponents are $\mathbf{n} = 2\mathbf{m}$ and $\mathbf{m} = \frac{1}{2}$.

The first Equation (2) in this three equation system is the simplest possible water runoff formulation, which can be viewed as a minimal version of Saint-Venant shallow water equations. In words, it simply states that water runs off at each point (x, y) in the direction opposite to its elevation gradient $\nabla(h+\theta)(x, y)$. The source term r expresses that rain is falling at constant rate all over the landscape. Of course nothing prevents using a space or time variable $r(x, y, t)$ if such rainfall data rates are available. The second Equation (3) contains all of the water-ground interactions as the sum of three terms modifying competitively the ground elevation. The first one is Gilbert's creep evolution

$$\frac{\partial h}{\partial t} = c\Delta h \tag{5}$$

by which a landscape tends to smooth out by a diffusion process. This is nothing but the heat equation. Applied alone, its visual effect is to blur the landscape image. The constant c reflects a diffusion speed depending on soil conditions and on the previously mentioned various perturbing factors (rain splash, wind, chemical weathering, ...). Such a creep term is found, among others, in [11], the GOLEM numerical simulation system (Tucker and Slingerland, 1994) [24], [19], [23], [7], [25].

The second term $-\epsilon\theta^{\mathbf{m}}|\nabla(h+\theta)|^{\mathbf{n}}$ in (3) expresses the stream incision law [20], [10]. It states that the erosion rate in a channel increases with the flux of water in the channel and with the local gradient. The preferred exponents in many models are $\mathbf{m} = \frac{1}{2}$, $\mathbf{n} = 2\mathbf{m} = 1$. The third term in the second Equation (3) $-s\frac{\lambda}{\theta}$ is Exner's sedimentation law by which the sedimentation rate is proportional to the density of sediment $\frac{\lambda}{\theta}$ in water.

To summarize, the second equation expresses that the elevation $h(x, y, t)$ evolves under the conjugate actions of creep, erosion and sedimentation. The third Equation (4) simply expresses the conservation and transport law of the sediment $\lambda(x, y)$ carried by the water. The first term is the run off term, strictly analogous to the one present in (2) for water runoff, as sediment is carried by water at the same horizontal velocity. The second and third terms are the opposite of the terms present in (3). Indeed, the terrain scraped by erosion becomes transported sediment, and can sediment to become terrain again.

The simulations of Figure 13 illustrate the variety of morphologies that can be reached from an initial DEM by varying the parameters of the two-equation model, namely equations (2-3) (in that way, no sedimentation occurs, which makes sense in the sloppy La Réunion). Some of the evolutions maintain a qualitative landscape morphology similar to the original, while others create new basins and rivers and evolve the DEM toward a different morphology. For example the second result and the fourth are obtained by fixing the same erosion percentage, 20%. But the second is still very similar to the original, showing a slow morphological evolution, while the fourth has created or expanded basins, as the landscape evolves to a mature form. In the middle of the last row, the exponent $\mathbf{m} = 0.6$ in the incision law has been changed from $\mathbf{m} = 0.5$ in the other experiments. Observe that this modification is enough to modify valley spacing on the left slope. It seems sound to deduce from this variety of results that, only with five control parameters, a large variety of landscapes morphologies can be modeled by a three-equation model.

3 Implementation

This section will focus on the practical implementation of the method which has been described theoretically in Section 2. First the main algorithm will be described in its entirety in Section 3.1, then Section 3.2 will explain the water initialization step and finally the core of the algorithm that implements the three main equations will be detailed in Section 3.3.

Please note that the images are normalized between $[\![0, 1]\!]$.

Remark: the boundary conditions have to be addressed carefully. Therefore Section 3.4 will be dedicated to this particular problem.

About the rain parameter r, it is a probability mask with the same size of the input image. In order to alleviate the notation, we shall write $r =$ "some constant" to denote the case when all the entries of this matrix are equal to that constant.

3.1 Main Algorithm

The main algorithm can be dissociated into two main steps: 1) the water initialization that fills the depressions of the landscape and initiates the rivers network and, 2) the main iteration that solves the three equations.

This main algorithm is briefly summarized in Algorithm 1. It describes the functions calling. Each and every one of those functions are fully described in Sections 3.2 and 3.3.

3.2 Water Initialization

This section describes the initialization of the water level θ. The main idea is to let the water go through the landscape until the river network takes shape and stabilizes. Therefore only the water evolution scheme is applied (i.e. the landscape and the sedimentation remain untouched). The water initialization is summarized in Algorithm. 2.

Algorithm 1: Main algorithm for the Landscape Evolution Model.

input: original landscape h. **parameter**: N_i, maximal number of iterations,
output: evolved landscape $\tilde{h}$, **parameter**: P, maximal percentage of land to remove.
output: final water network $\tilde{\theta}$,
output: final sediment concentration network $\tilde{\lambda}$.

Part 1: Water initialization
Initialization of the water (Algorithm 2)

Part 2: Run the three equations

Initialization of the variables:
$n = 0$
Compute the initial mass of the landscape m_c (Algorithm 3)
$m_t = m_c(1 - P)$

Main loop
while $n < N_i$ ***and*** $m_c \geq m_t$ **do**
  Run the water and sedimentation evolution (Algorithm 7)
  Run the landscape evolution (Algorithm 5)
  Run the creep evolution (Algorithm 6)
  Update the current mass m_c of the landscape (Algorithm 3)
  $n = n + 1$

Algorithm 2: Water runoff initialization

input : initial landscape h. **parameter**: N_b, number of iterations
output : water initialized θ_o, **parameter**: θ_i, initial water level
parameter: θ_0, ocean level.

Initialization of the water
$\theta_o = \theta_i$

Run the water evolution step
for $n = 1$ *to* N_b **do**
  Set border conditions (Algorithm 8 or 9 or 10)
  Perform one step of the water evolution (Algorithm 4)

Initialize the ocean level
for $(i, j) \in [\![0, NR - 1]\!] \times [\![0, NC - 1]\!]$ **do**
  if $h(i, j) < \theta_0$ **then**
    $\theta_o(i, j) = \max\{\theta_0 - h(i, j), 0\}$

Remark: This initialization needs not to be extremely precise, so both time- and space-multiscale approaches can be used in order to speed up the process, as described in Section 4.2.

3.3 Core Algorithm

Once the water level has been initialized, the three equations of the LEM (water, elevation, and sediment load) may be performed.

In order to determine whether to stop the main loop, one possibility is to check the percentage of mass that remains after the evolution. This is done by computing the total pixel's height above the ocean, as described in Algorithm 3.

The algorithm of the water evolution scheme is described in Algorithm 4. The landscape evolution is described in Algorithm 5, the creep evolution is described in Algorithm 6, and finally the conjoined water and sedimentation evolution is described in Algorithm 7.

Algorithm 3: Integrates the mass of the current emerged landscape (over ocean level).

input: current landscape h. **parameter**: θ_0, ocean level.
output: total mass of the current landscape m_c (non normalized).

$m_c = 0$
for $(i,j) \in [\![0, NR-1]\!] \times [\![0, NC-1]\!]$ **do**
  $m_c = m_c + \max\{h(i,j) - \theta_0, 0\}$

3.4 Border Conditions

To be able to compute landscape, water or sediment derivatives at pixels on the image border, we must attribute initial values to these functions at out-of-domain pixels. Since these out-of-domain pixels are not updated by the main evolution schemes (only the in-domain pixels are), these border pixels must be updated adequately before each iteration of the evolution scheme.

Three options will be available, all equally arguable: the Dirichlet conditions (that set the border equals to zero, see Algorithm 8), the Neumann conditions (constant extension, see Algorithm 9) and the slope extension (see Algorithm 10). In the online demo we indicate our preference as a default value for this boundary option.

Figure 3 illustrates the differences between the three options.

0	0	0	0	0	0	0	0		11	11	11	12	13	14	14	14		1	2	3	4	5	6	7	8
0	0	0	0	0	0	0	0		11	11	11	12	13	14	14	14		5	6	7	8	9	10	11	12
0	0	11	12	13	14	0	0		11	11	11	12	13	14	14	14		9	10	11	12	13	14	15	16
0	0	15	16	17	18	0	0		15	15	15	16	17	18	18	18		13	14	15	16	17	18	19	20
0	0	19	20	21	22	0	0		19	19	19	20	21	22	22	22		17	18	19	20	21	22	23	24
0	0	0	0	0	0	0	0		19	19	19	20	21	22	22	22		21	22	23	24	25	26	27	28
0	0	0	0	0	0	0	0		19	19	19	20	21	22	22	22		25	26	27	28	29	30	31	32

Figure 3: Illustration of the three different options for the border conditions. In grey: original image, in red: borders extension. From left to right: Dirichlet conditions, Neumann conditions, slope extension.

4 Visualization and Speed-Ups

The equations of the LEM have been developed in Section 2 and the algorithm described in details in Section 3. We now discuss several significant improvements of the experimental setup.

Section 4.1 addresses the definition of the right landscape visualization tools.

Section 4.2 presents a numerical acceleration tool based on a multiscale formulation speeding up considerably the initialization of the water runoff scheme.

Algorithm 4: Performs one step of the water evolution.

input : previous water level θ^{n-1}; **parameter**: δ_t, time step;
input : landscape height h^{n-1}. **parameter**: r, rain matrix.
output : updated water level θ^n.

Set the border conditions (Algorithm 8, or 9, or 10)

For convenience, declaration of a temporal image
$\tau = (0)_{\{NC,NR\}}$

for $(i,j) \in [\![0, NR-1]\!] \times [\![0, NC-1]\!]$ **do**
  Compute the transfer over the 8 neighbors
  for $(i',j') \in \{(0,\pm1),(\pm1,0),(\pm1,\pm1),(\pm1,\mp1)\}$ **do**
    Normalization factor
    if $(i',j') = (0,\pm1)$ ***or*** $(\pm1,0)$ **then**
      $\omega = 1$
    else
      $\omega = \frac{1}{\sqrt{2}}$
    Compute the steep
    $\text{steep} = \theta^{n-1}(i',j') + h^{n-1}(i',j') - \theta^{n-1}(i,j) - h^{n-1}(i,j)$
    Update the transfer
    if *steep* > 0 **then**
      $\tau(i,j) = \tau(i,j) + \omega \times \delta_t \times \text{steep} \times \theta^{n-1}(i',j')$
    else
      $\tau(i,j) = \tau(i,j) + \omega \times \delta_t \times \text{steep} \times \theta^{n-1}(i,j)$
  The water level can't be negative
  $\tau(i,j) = \max\{-\delta_t \times r(i,j) - \theta^{n-1}(i,j), \tau(i,j)\}$

Update the water level
for $(i,j) \in [\![0, NR-1]\!] \times [\![0, NC-1]\!]$ **do**
  $\theta^n(i,j) = \theta^{n-1}(i,j) + \tau(i,j) + \delta_t \times r(i,j)$

Algorithm 5: Performs one step of the landscape evolution scheme.

input	: previous landscape height h^{n-1}	**parameter**:	**m**, power exponent;
input	: current water level θ^{n-1};	**parameter**:	**n**, power exponent;
input	: previous sedimentation concentration λ^{n-1}	**parameter**:	e, erosion coefficient;
output	: updated landscape height h^n;	**parameter**:	s, sedimentation coefficient;
output	: updated sediment concentration λ^n.	**parameter**:	δ_t, time step;
		parameter:	θ_0, ocean level.

Set the border conditions (Algorithm 8, or 9, or 10)

Compute the erosion

for $(i,j) \in [\![0, NR-1]\!] \times [\![0, NC-1]\!]$ **do**

$\delta_e = 0$

Horizontal

$d_l = h^{n-1}(i,j) + \theta^{n-1}(i,j) - h^{n-1}(i,j-1) - \theta^{n-1}(i,j-1)$

$d_r = h^{n-1}(i,j) + \theta^{n-1}(i,j) - h^{n-1}(i,j+1) - \theta^{n-1}(i,j+1)$

$\delta_e = \delta_e + \left(\max\{d_l, d_r, 0\}\right)^2$

Vertical

$d_t = h^{n-1}(i,j) + \theta^{n-1}(i,j) - h^{n-1}(i-1,j) - \theta^{n-1}(i-1,j)$

$d_b = h^{n-1}(i,j) + \theta^{n-1}(i,j) - h^{n-1}(i+1,j) - \theta^{n-1}(i+1,j)$

$\delta_e = \delta_e + \left(\max\{d_t, d_b, 0\}\right)^2$

Diagonals

$d_{tl} = h^{n-1}(i,j) + \theta^{n-1}(i,j) - h^{n-1}(i-1,j-1) - \theta^{n-1}(i-1,j-1)$

$d_{br} = h^{n-1}(i,j) + \theta^{n-1}(i,j) - h^{n-1}(i+1,j+1) - \theta^{n-1}(i+1,j+1)$

$\delta_e = \delta_e + \frac{1}{2}\left(\max\{d_{tl}, d_{br}, 0\}\right)^2$

$d_{tr} = h^{n-1}(i,j) + \theta^{n-1}(i,j) - h^{n-1}(i-1,j+1) - \theta^{n-1}(i-1,j+1)$

$d_{bl} = h^{n-1}(i,j) + \theta^{n-1}(i,j) - h^{n-1}(i+1,j-1) - \theta^{n-1}(i+1,j-1)$

$\delta_e = \delta_e + \frac{1}{2}\left(\max\{d_{tr}, d_{bl}, 0\}\right)^2$

Get the final amount to erode

$\delta_e = -e\left(\theta^{n-1}(i,j)\right)^{\mathbf{m}} \left(\delta_e\right)^{\mathbf{n}/2}$

No erosion below ocean level:

if $h^{n-1}(i,j) < \theta_0$ **then**

$\delta_e = 0$

Compute the sedimentation

if $\theta^{n-1}(i,j) > 0$ **then**

$\delta_s = s\frac{\lambda^{n-1}(i,j)}{\theta^{n-1}(i,j)}$

else

$\delta_s = 0$

Landscape to increment:

$\delta = \delta_t(\delta_e + \delta_s)$

The transported sediment can't be negative:

$\delta = \min\{\delta, \lambda^{n-1}(i,j)\}$

No submarine erosion:

$\delta = \max\{\delta, -h^{n-1}(i,j)\}$

Update the landscape height

$h^n(i,j) = h^{n-1}(i,j) + \delta$

Update the sediment concentration

$\lambda^n(i,j) = \lambda^{n-1}(i,j) - \delta$

Algorithm 6: Performs one step of the creep evolution scheme.

input : previous landscape height h^{n-1}. **parameter**: c, creep coefficient;
output : current landscape height h^n. **parameter**: δ_t, time step.

Set the border conditions (Algorithm 8, or 9, or 10)
for $(i,j) \in [\![0, NR-1]\!] \times [\![0, NC-1]\!]$ **do**
Compute the creep over the 8 neighbors

$$\begin{aligned}\Delta =&[h^{n-1}(i,j-1) + h^{n-1}(i,j+1) + h^{n-1}(i-1,j)\\ &+ h^{n-1}(i+1,j) - 4h^{n-1}(i,j)]+\\ &\frac{1}{\sqrt{2}}[h^{n-1}(i-1,j-1) + h^{n-1}(i-1,j+1) + h^{n-1}(i+1,j-1)\\ &+ h^{n-1}(i+1,j+1) - 4h^{n-1}(i,j)]\end{aligned}$$

Diffusion term: $\delta = \min\{\delta_t c\Delta, \frac{1}{4}\Delta\}$
No submarine creep: $\delta = \max\{\delta, \theta_0 - h^{n-1}(i,j)\}$
Update the landscape height
$h^n(i,j) = h^{n-1}(i,j) + \delta$

4.1 Visualization

As illustrated in Figure 4, it is quite difficult to discern how a landscape has evolved, particularly in its low (and therefore dark) regions. The visual difference between the original landscape and the final one are not conspicuous.

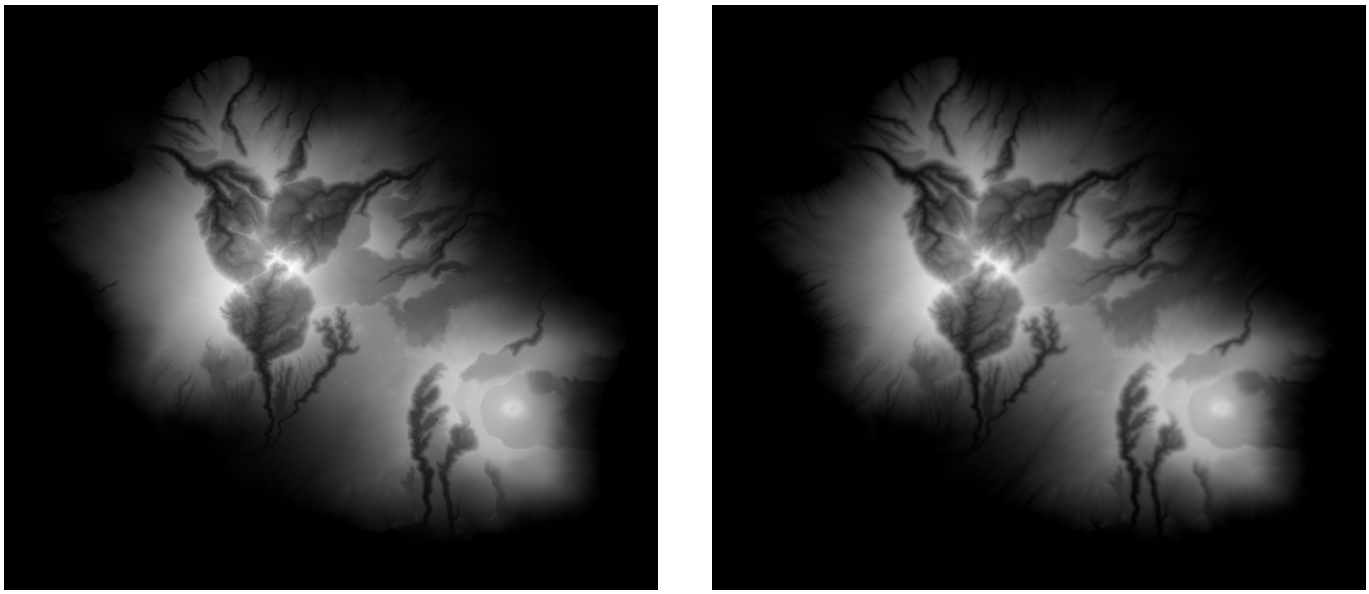

Figure 4: From left to right: original landscape of the island "La Réunion', evolved one with 5% ($\mathcal{N}_g = 0.337$, $\mathcal{N}_i = 0.047$) of the landscape removed ($r = 1.0 \times 10^{-7}$, $e = 5.0$, $c = 10.0 \times 10^{-3}$, $s = 0.2 \times 10^{-6}$, $\mathbf{m} = 0.5$, $\mathbf{n} = 1.0$, no uplift, Dirichlet conditions for θ, Neumann conditions for h).

Therefore the visualization is improved with two tools: 1) a false color visualization, that uses

Algorithm 7: Performs one step of the water and sedimentation evolution scheme.

input : previous water level θ^{n-1}; **parameter**: δ_t, time step;
input : landscape height h^{n-1}; **parameter**: r, rain matrix;
input : previous sediment concentration λ^{n-1}. **parameter**: θ_0, ocean level.
output : current water level θ^n;
output : current sediment concentration λ^n.

Set the border conditions (Algorithm 8, or 9, or 10)
For convenience, declaration of a temporary image
$\tau_\theta = (0)_{\{NC,NR\}}$ $\tau_\lambda = (0)_{\{NC,NR\}}$
for $(i,j) \in [\![0, NR-1]\!] \times [\![0, NC-1]\!]$ **do**
  Compute the transfer over the 8 neighbors
  for $(i',j') \in \{(0,\pm1),(\pm1,0),(\pm1,\pm1),(\pm1,\mp1)\}$ **do**
    Normalization factor
    if $(i',j') = (0,\pm1)$ ***or*** $(\pm1,0)$ **then**
      $\omega = 1$
    else
      $\omega = \frac{1}{\sqrt{2}}$
    Compute the steep
    $\text{steep} = \theta^{n-1}(i',j') + h^{n-1}(i',j') - \theta^{n-1}(i,j) - h^{n-1}(i,j)$
    Update the transfer
    if *steep* > 0 **then**
      $\tau_\theta(i,j) = \tau_\theta(i,j) + \omega \times \delta_t \times \text{steep} \times \theta^{n-1}(i',j')$
      $\tau_\lambda(i,j) = \tau_\lambda(i,j) + \omega \times \delta_t \times \text{steep} \times \lambda^{n-1}(i',j')$
    else
      $\tau_\theta(i,j) = \tau_\theta(i,j) + \omega \times \delta_t \times \text{steep} \times \theta^{n-1}(i,j)$
      $\tau_\lambda(i,j) = \tau_\lambda(i,j) + \omega \times \delta_t \times \text{steep} \times \lambda^{n-1}(i,j)$
  The water level and sediment in water should always be non-negative
  $\tau_\theta(i,j) = \max\{-\theta^{n-1}(i,j) - \delta_t \times r(i,j), \tau_\theta(i,j)\}$
  $\tau_\lambda(i,j) = \max\{-\lambda^{n-1}(i,j), \tau_\lambda(i,j)\}$
Update the water and water times concentration level
for $(i,j) \in [\![0, NR-1]\!] \times [\![0, NC-1]\!]$ **do**
  The ocean level shouldn't change if $h^{n-1}(i,j) < \theta_0$ **then**
    $\theta^n(i,j) = \theta_0 - h(i,j)$
  else
    $\theta^n(i,j) = \theta^{n-1}(i,j) + \tau_\theta(i,j) + \delta_t \times r(i,j)$
  $\lambda^n(i,j) = \lambda^{n-1}(i,j) + \tau_\lambda(i,j)$

Algorithm 8: Initializes the out-of-domain pixels of an image (Dirichlet conditions).

input : inner image u.
output : $\tilde{u}$, with its out-of-domain pixels initialized and the inner pixels untouched.

Top and bottom lines
for $j = 0$ ***to*** $NC - 1$ **do**
  $\tilde{u}(-1, j) = 0$
  $\tilde{u}(NR, j) = 0$
Left and right lines
for $i = -1$ ***to*** NR **do**
  $\tilde{u}(i, -1) = 0$
  $\tilde{u}(i, NC) = 0$

Algorithm 9: Initializes the out-of-domain pixels of an image (Neumann conditions).

input : inner image u.
output : $\tilde{u}$, with its out-of-domain pixels initialized and the inner pixels untouched.

Top and bottom lines
for $j = 0$ ***to*** $NC - 1$ **do**
  $\tilde{u}(-1, j) = u(0, j)$
  $\tilde{u}(NR, j) = u(NR - 1, j)$
Left and right lines
for $i = -1$ ***to*** NR **do**
  $\tilde{u}(i, -1) = u(i, 0)$
  $\tilde{u}(i, NC) = u(i, NC - 1)$

Algorithm 10: Initializes the out-of-domain pixels of an image (slope extension).

input : inner image u.
output : $\tilde{u}$, with its out-of-domain pixels initialized and the inner pixels untouched.

Top and bottom lines
for $j = 0$ ***to*** $NC - 1$ **do**
  $\tilde{u}(-1, j) = 2u(0, j) - u(1, j)$
  $\tilde{u}(NR, j) = 2u(NR - 1, j) - u(NR - 2, j)$
Left and right lines
for $i = -1$ ***to*** NR **do**
  $\tilde{u}(i, -1) = 2u(i, 0) - u(i, 1)$
  $\tilde{u}(i, NC) = 2u(i, NC - 1) - u(i, NC - 2)$

both the water level image and the landscape image and 2), a false 3D rendering based on shape from shading. This fast technique was invented by Eduard Imhof in 1925.

Colorization: two different color palettes are used for both the water level and the landscape altitude. Both palettes are applied to both images, where the normalized values of the pixel are stretched into the palette range (see Algorithm 12). For instance the highest pixel of landscape will be colored in white, and the lowest one in dark brown.

Then both images are merged into one, in order to see the river network directly on the landscape image. The threshold below which a pixel appears as belonging to the water is fixed as follows:

- If the landscape elevation is below ocean level, then the current pixel appears as water;
- If the water level is above τ_θ times the average water level over all the landscape, then the current pixel appears as water. By default $\tau_\theta = 1$;
- Otherwise, the current pixel appears as ground.

False 3D rendering: First, a shadow image is obtained by computing the scalar product between an hypothesized direction of the sun (-1, -1, 1) and the normal to the landscape. Then a Cauchy blur is applied to this shadow image, before stretching its dynamic between 0 and 1. This stretching uses the simplest color balance technique [17], where the image is normalized between the 1% and 90% percentile, which saturates a small proportion of the pixels, but provides a better contrasted result.

Second, a negative Laplacian is computed over the landscape image, blurred and stretched in the same way.

In continuation, a convex combination is performed between the shadow image (80%) and the Laplacian image (20%).

Finally the color image is simply the product of the colorized map and the 3D rendered image. Figure 5 shows that the shadowed parts of the landscape become much more visible, and the differences between the original and final landscape more discernible. Algorithm 11 summarizes the steps to obtain the false 3D rendering.

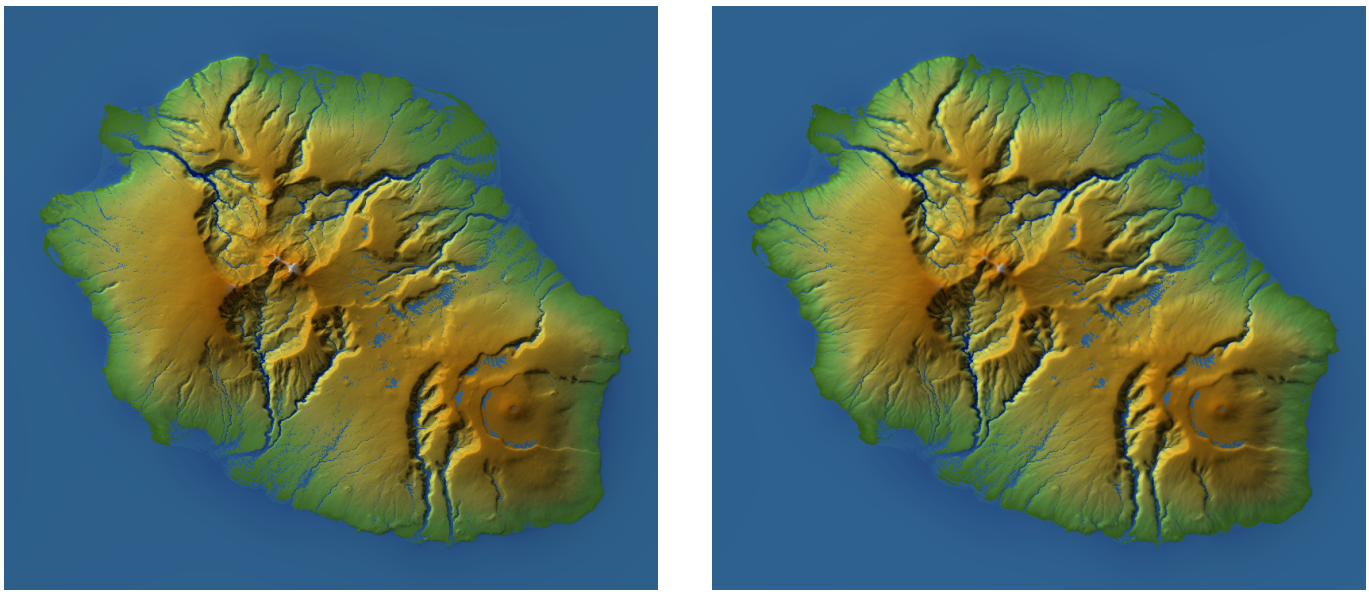

Figure 5: Result of the false color visualization over the images presented in Figure 4. From left to right: original landscape (with initialized water level), final landscape (with final water level).

Algorithm 11: Applies the false color and 3D visualization.

input : landscape h;
input : water level θ;
output : color image u.

Get the colorized map u_c from h and θ (Algorithm 12)
Get the shadow image u_s from h (Algorithm 13)
Apply a Cauchy blur on u_s
Stretch $u_s \in [0, 0.9]$ (Algorithm 14)
Get the opposite of the Laplacian u_L from h (Algorithm 15)
Apply a Cauchy blur on u_L
Stretch $u_L \in [0, 0.9]$ (Algorithm 14)
Apply the blend
for $c \in [\![1, 3]\!]$ **do**
  for $(i, j) \in [\![0, NR - 1]\!] \times [\![0, NC - 1]\!]$ **do**
    $u(i, j, c) = u_c(i, j, c) \left[0.1 + 0.8u_s(i, j) + 0.2u_L(i, j)\right]$

Algorithm 12: Applies the color palettes of the ground and the ocean.

input : h, landscape; **parameter:** θ_0, ocean level;
input : θ, water level; **parameter**: τ_θ, threshold coefficient for water visualization.
input : θ_0, ocean level;
input : p_h, normalized color palette for the ground of size N_h;
input : p_θ, normalized color palette for the ocean of size N_θ;
output : u_c, colorized map.

Initialization
$\theta_m = \text{mean}(\theta)$, $h_{min} = \min(h)$, $h_{max} = \max(h)$, $\theta_{min} = \min(\theta)$, $\theta_{max} = \max(\theta)$
$\delta_h = \frac{(N_h - 1)}{h_{max} - h_{min}}$, $\delta_\theta = \frac{(N_\theta - 1)}{\theta max - \theta min}$.
Apply the colorization
for $(i, j) \in [\![0, NR - 1]\!] \times [\![0, NC - 1]\!]$ **do**
  $v_h = h_{max} \left(\frac{h(i,j) - \theta_0}{h_{max}}\right)^{1/4}$
  if $h(i, j) \leq \theta_0$ ***or*** $\theta(i, j) \geq \tau_\theta \theta_m$ **then**
    $u_c(i, j) = p_\theta \left([\theta(i, j) - \theta_{min}]\, \delta_\theta\right)$
  else
    $u_c(i, j) = p_h \left([h(i, j) - h_{min}]\, \delta_h\right)$

4.2 Multi-Scale Approach

The main issue of LEM algorithms is their complexity. Even if the algorithm presented here can be efficiently vectorized (for example by using SSE approaches in the case of Intel CPUs) and parallelized (using OpenMP), the number of iterations is quite high. In particular, to get an acceptable water network in the initialization, it requires up to 40 000 iterations.

Since this network needs not really get a high resolution at the beginning, a multi-scale approach can be used to speed up considerably the water initialization process. For both space and time multi-scale procedures, the common idea is the following: to start coarse and fast, and to slow down

Algorithm 13: Obtains the shadow image from the landscape.

input : landscape h;
output : shadow image u_s.

for $(i,j) \in [\![0, NR-1]\!] \times [\![0, NC-1]\!]$ **do**
  $i_T = \max(0, i-1)$, $i_B = \min(NR-1, i+1)$,
  $j_L = \max(0, j-1)$, $j_R = \min(NC-1, j+1)$.
  Compute the vector for the horizontal gradient
  $V_x = \left\{1, 0, \frac{h(i,j_R)-h(i,j_L)}{2}\right\}$
  Compute the vector for the vertical gradient
  $V_x = \left\{1, 0, \frac{h(i_B,j)-h(i_T,j)}{2}\right\}$
  Vector for the sun (lighting)
  $V_s = \{-1, -1, 1\}$
  Get the normal of the surface
  $V_n = \{V_x(2)V_y(3) - V_x(3)V_y(2), V_x(3)V_y(1) - V_x(1)V_y(3), V_x(1)V_y(2) - V_x(2)V_y(1)\}$
  Compute the scalar product to get the shadow
  $u_s(i,j) = V_n(1)V_s(1) + V_n(2)V_s(2) + V_n(3)V_s(3)$

Algorithm 14: Stretches an image.

input : image u to stretch;
output : stretched image $\tilde{u}$

Sort the values of u
$u_s = \text{sort}(u)$
Get the quantiles at 1% **and** 90%
$q_{min} = u_s\left(\lfloor \frac{(NR-1)(NC-1)}{100} \rfloor\right)$
$q_{max} = u_s\left(\lfloor (NR-1)(NC-1)\frac{90}{100} \rfloor\right)$
Stretch the values between $[0, 0.9]$
for $(i,j) \in [\![0, NR-1]\!] \times [\![0, NC-1]\!]$ **do**
  if $u(i,j) \le q_{min}$ **then**
    $\tilde{u}(i,j) = 0$
  else if $u(i,j) \ge q_{max}$ **then**
    $\tilde{u}(i,j) = 0.9$
  else
    $\tilde{u}(i,j) = \frac{0.9}{q_{max}-q_{min}}(u(i,j) - q_{min})$

to refine the process in the last phase.

Space Multi-Scale Approach

1. Apply a zoom-out of a factor 4 in each direction over the original landscape image h_0 to get h_2;
2. Run $\frac{N_b}{2}$ iterations of the water transport equation over h_2 to get θ_2;
3. Apply a zoom-in of a factor 2 in each direction over this coarse water level image to get θ_1;

Algorithm 15: Computes the opposite of a discrete Laplacian over an image.

input : image u;
output : Laplacian image $\tilde{u}$

for $(i,j) \in [\![0, NR-1]\!] \times [\![0, NC-1]\!]$ **do**
  $i_T = \max(0, i-1)$, $i_B = \min(NR-1, i+1)$,
  $j_L = \max(0, j-1)$, $j_R = \min(NC-1, j+1)$.
  $u_L(i,j) = 4u(i,j) - u(i_T,j) - u(i_B,j) - u(i,j_L) - u(i,j_R)$

4. Apply a zoom-out of a factor 2 in each direction over the original landscape image to get h_1;

5. From the zoomed-in coarse water level image θ_1, continue the water transport equation over h_1 and apply $\frac{N_b}{4}$ iterations;

6. Apply a zoom-in of a factor 2 in each direction over θ_1 to get θ_0;

7. Apply $\frac{N_b}{32}$ iterations of the water transport equation over h_0 from θ_0.

This approach is summarized in Algorithm 16.

Algorithm 16: Space Multi-Scale Approach.

input : original landscape h;
input : Number of water initialization iterations N_b;
output : initialized water level θ.

Scale 2
Apply a zoom-out over h to get h_1
Apply a zoom-out over h_1 to get h_2
Apply $\frac{N_b}{2}$ iterations of the water evolution over h_2 to get θ_2 (Algorithm 4)
Scale 1
Apply a zoom-in over θ_2 to get θ_1
Apply $\frac{N_b}{4}$ iterations of the water evolution over h_1 and θ_1 (Algorithm 4)
Scale 0
Apply a zoom-in over θ_2 to get θ_0
Apply $\frac{N_b}{32}$ iterations of the water evolution over h_0 and θ_0 to get θ (Algorithm 4)

Thanks to this approach, and because it requires less and less iterations to get the finest details, we get a theoretical speed-up factor of 8, without any substantial precision loss.

The zoom-out is obtained by first applying a Gaussian blur of factor 1.4 over the image; and second by simply extracting one pixel over two in both directions in order to get a four times smaller image. The zoom-in is obtained by bilinear interpolation.

Time Multi-Scale Approach Only used during the water initialization phase in the online demo, this acceleration can be activated in the provided code. It is based on the same basic idea as the space multi-scale approach: going fast in the beginning, then progressively slowing down when approaching the desired result:

1. Multiply the time step parameter δ_t by a given factor of acceleration, say 4 for instance;

2. Use this new δ_t to reach half the wanted number of iterations (or half of the percentage to remove);

3. Divide by two δ_t and reach half of the remaining (i.e. $\frac{3}{4}$ of the wanted result);

4. Finish the evolution with the original δ_t.

Thanks to this approach, we get another theoretical speed-up factor of 2, without noticeable precision loss. However, as this acceleration sometimes lets δ_t become larger than 1, it may cause some numerical instability. Therefore it has been removed from the online demo.

5 Three Different Experimental Modes On Line

In this section, three different modes will be described: 1) the LEM (Landscape Evolution Model, 5.0.1) that corresponds exactly to the algorithms described in Section 3, 2) the LAPE (Landscape Automatic Parameters Estimation, 5.0.2) that tries to automatically estimate the parameter values for a given landscape in such a way that this landscape remains close in shape to the original after the evolution, and 3) the LS (Landscape Synthesis, Section 5.0.3) which purpose is to recreate artificial landscapes from scratch that look like natural ones.

5.0.1 Landscape Evolution Model

This mode follows exactly the theory described in Section 2. Its main purpose is to propose an evolution where each parameter can be controlled and chosen by the experimenter. Due to the impossibility to get a ground truth over a few thousand years of evolution the only way to estimate the validity of the results is to look at them.

Parameters: for this mode, the available set of parameters is composed of e, c, s, r, $\mathbf{m}$ and $\mathbf{n}$ associated to the main equations. In addition, two optional parameters are proposed: the initial ocean level θ_0 and the original water level θ_i that sets a uniform layer of water all over the landscape.

Stopping criteria: in addition to those parameters, the two stopping criteria that determine the percentage of landscape to remove P, which is the ratio of difference between the initial and final landscape mass, divided by the total initial mass, and the maximum number of iterations for the main equations (N_i).

Refinement: as described in Section 5.1, it is possible to get enhanced results by first applying a zoom-in on the input landscape, and obtain the final result with a zoom-out. Especially, as described in Figure 12, this option provides results with thinner valleys and details.

Automation: Finally, two more options are proposed. The first one allows for an automatic estimation of the erosion rate e in such a way that for a given number of iterations the landscape evolution should reach the desired percentage of landscape to remove P. This automatic estimation of e is really important to compare comparable evolutions as described in Section 5.0.2. An example of the use of these parameters is shown in Figure 6.

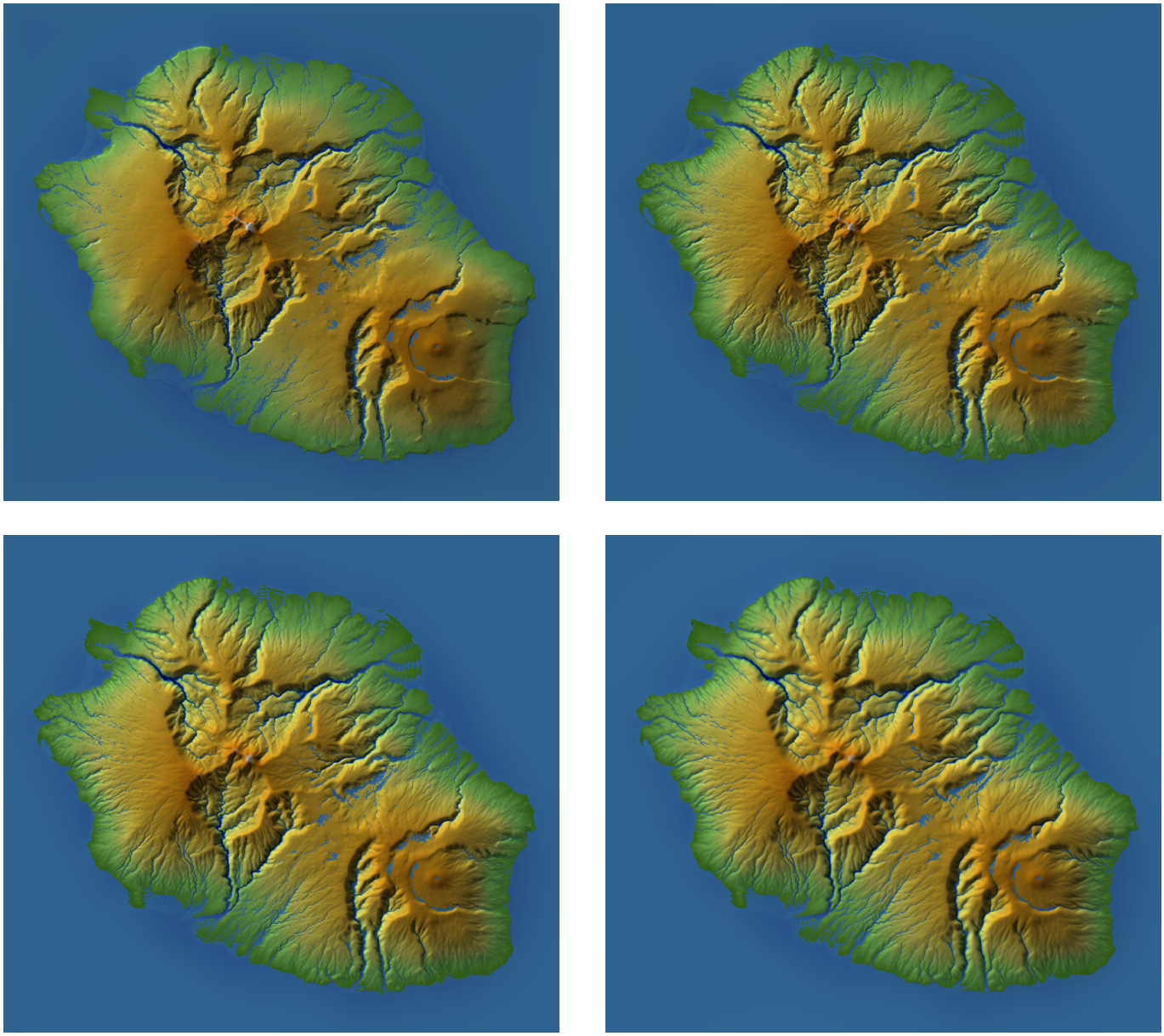

Figure 6: Example of the automatic estimation of the erosion rate. Common parameters values: $r = 1.0\times10^{-7}$, $c = 2\times10^{-3}$, $s = 0.2 \times 10^{-6}$, $\mathbf{m} = 0.5$, $\mathbf{n} = 1.0$. From left to right, top to bottom: original landscape, ($e = 8.44$ (estimated), $P = 9.9\%(reached)$, $N_i = 100$), ($e = 1.8$ (estimated), $P = 9.2\%(reached)$, $N_i = 500$), ($e = 0.92$ (estimated), $P = 9.0\%(reached)$, $N_i = 1000$). This experiment illustrates how we can fix an arbitrary number of iterations and automatically adapt the erosion parameter e to reach an erosion percentage target (here, the target was 9%).

The second option is the automatic uplift to compensate the erosion of the landscape. Since the landscape must not have negative values, when it gets close to zero due to the erosion it is flattened artificially. This occurs quite easily for large values of P and e. In order to avoid undesired behavior, an automatic uplift is added at each iteration to maintain the average altitude of the landscape constant.

For the automatic uplift, two options are available: uniform and local. For the uniform one, the same altitude is added to all pixels. For the local one, the uplift term is proportional to $(h(i,j))^{\frac{1}{4}}$. Figure 7 shows the importance of this option when the percentage of removed landscape is important, and the difference between both uplifts.

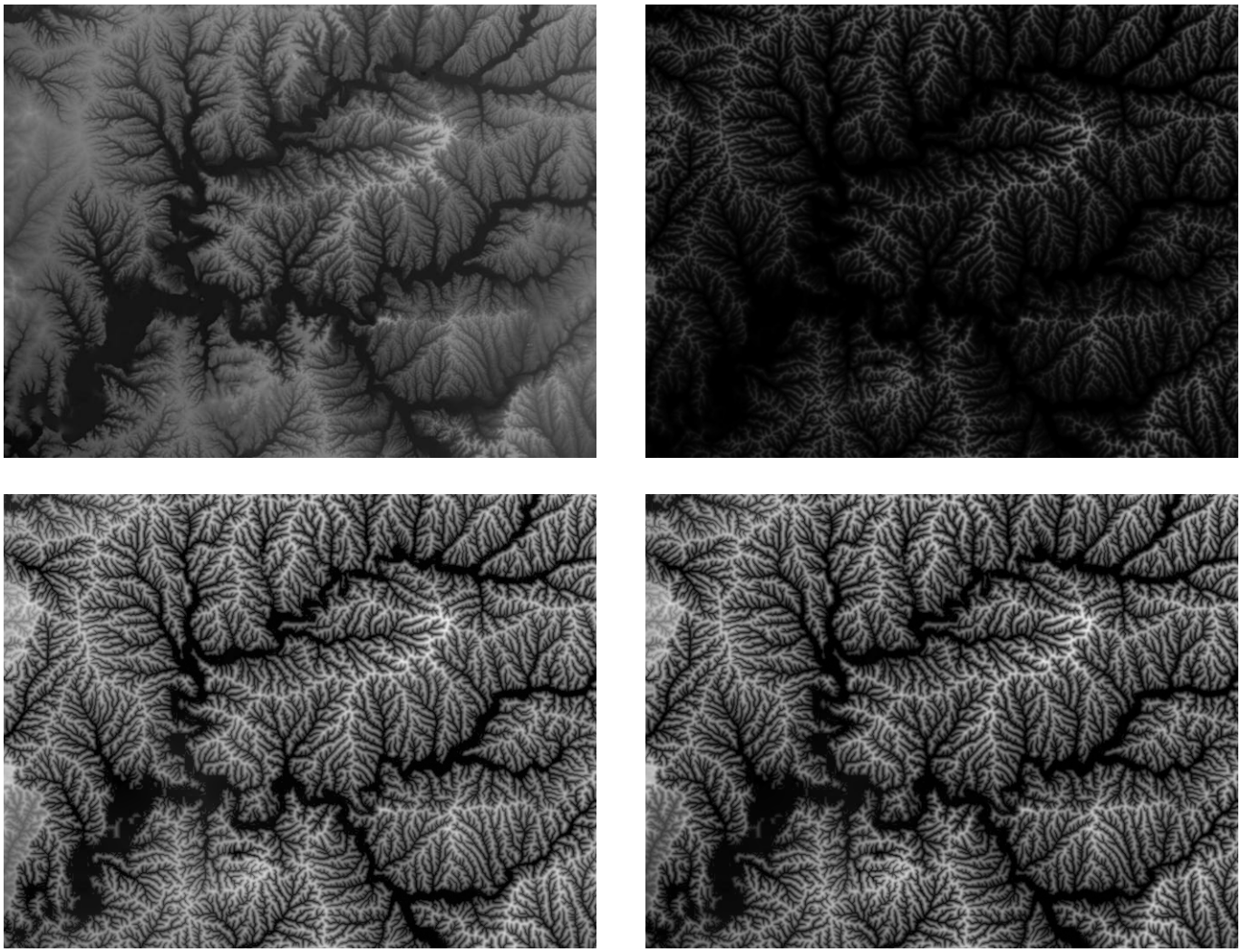

Figure 7: Example of the automatic uplift. Common parameters values: $r = 1.0 \times 10^{-7}$, $c = 2 \times 10^{-3}$, $e = 15$, $s = 0.2 \times 10^{-6}$, $\mathbf{m} = 0.5$, $\mathbf{n} = 1.0$, $N_i = 1000$, Dirichlet conditions for θ, Neumann conditions for h. From left to right, top to bottom: original landscape, final landscape ($P = 79.3\%$, $\mathcal{N}_g = 0.823$, $\mathcal{N}_i = 1.057$), final landscape with uniform uplift ($P = 92.4\%$, $\mathcal{N}_g = 1.015$, $\mathcal{N}_i = 0.337$), final landscape with local uplift ($P = 88.2\%$, $\mathcal{N}_g = 1.060$, $\mathcal{N}_i = 0.367$.)

5.0.2 Landscape Automatic Parameters Estimation

Even if the algorithm described in Section 3 with the simple mode (Section 5.0.1) works as expected, it would be an overwhelming task to estimate correctly the set of parameters by hand, one by one.

In order to automate this estimation, a norm has been developed to translate the following assumption: *for a mature landscape, the best set of values for the parameters should favor an evolution in such a way that the evolved landscape's shape remains closest in shape to the original one.*

This assumption requires the use of a norm comparing the shapes of landscapes. Clearly, landscapes are similar if theirs slopes are parallel. This means that for a stable landscape, both original and final landscapes should keep similar gradients after some small evolution. Therefore the norm that will be minimized during the estimation will be

$$\mathcal{N}_g\left(h,\tilde{h}\right)=\sqrt{\frac{2\int_{h>\theta_0}\left(\left(\frac{\partial h}{\partial x}-\frac{\partial \tilde{h}}{\partial x}\right)^2+\left(\frac{\partial h}{\partial y}-\frac{\partial \tilde{h}}{\partial y}\right)^2\right)}{\int_{h>\theta_0}\left(\left(\frac{\partial h}{\partial x}\right)^2+\left(\frac{\partial h}{\partial y}\right)^2+\left(\frac{\partial \tilde{h}}{\partial x}\right)^2+\left(\frac{\partial \tilde{h}}{\partial y}\right)^2\right)}} \tag{6}$$

We also provide for comparison the more trivial L^2 norm $\mathcal{N}_i$ in the demo, defined as

$$\mathcal{N}_i\left(h,\tilde{h}\right)=\sqrt{\frac{2\int_{h>\theta_0}\left(h-\tilde{h}\right)^2}{\int_{h>\theta_0}\left(h^2+\tilde{h}^2\right)}} \tag{7}$$

Our first endeavor is to validate the existence of a set of parameters of (e, s) that performs better than a pure creep. Figure 8 shows that for ($r = 1 \times 10^{-7}$, $N_i = 500$, $c = 10.0 \times 10^{-3}$) fixed, the erosion and the sedimentation can counter-balance the effect of the creep to remain closer to the original landscape in the sense of the Sobolev norm, Equation (6).

When looking for the structural parameters that give the best stability, simulation parameters like r and c can be fixed to focus on the main unknown structural parameter $\mathbf{m}$, $\mathbf{n}$ and the sedimentation rate s.

Stopping criteria: The number of iterations N_i determines the desired precision and the amount of creep, whereas the percentage of landscape to remove P determines the erosion parameter. As described in Section 5.0.1, e will be automatically tuned according to N_i and P for each tested set $(s, \mathbf{m}, \mathbf{n})$.

Estimation: The initial value for the triplet $(s, \mathbf{m}, \mathbf{n})$ is selected by the user. Then for each parameter, one value on the left and one on the right are tested, and the best triplet over the 27 tested that provides the smallest norm will be kept.

We found a local minimum around $(s, \mathbf{m}, \mathbf{n}) = (0.2\times10^{-6}, 0.5, 1.0)$ for almost all tested landscapes.

5.0.3 Landscape Synthesis

Once the Landscape Evolution Model has been validated visually leading to a natural evolution, and that a set of parameters can be automatically estimated on every landscape (see Section 5.0.2), it becomes possible to recreate artificial landscapes from scratch. As it will be discussed in the following, the aim of this method is to provide a set of tools to create artificial landscape by using the parameters estimated in Section 5.0.2, starting from a rough version of any landscape.

Parameters: for this mode, the available set of parameters is composed of e, c, s, r, $\mathbf{m}$ and $\mathbf{n}$ associated to the main equations. In addition, the initial ocean level θ_0 can be adjusted as well.

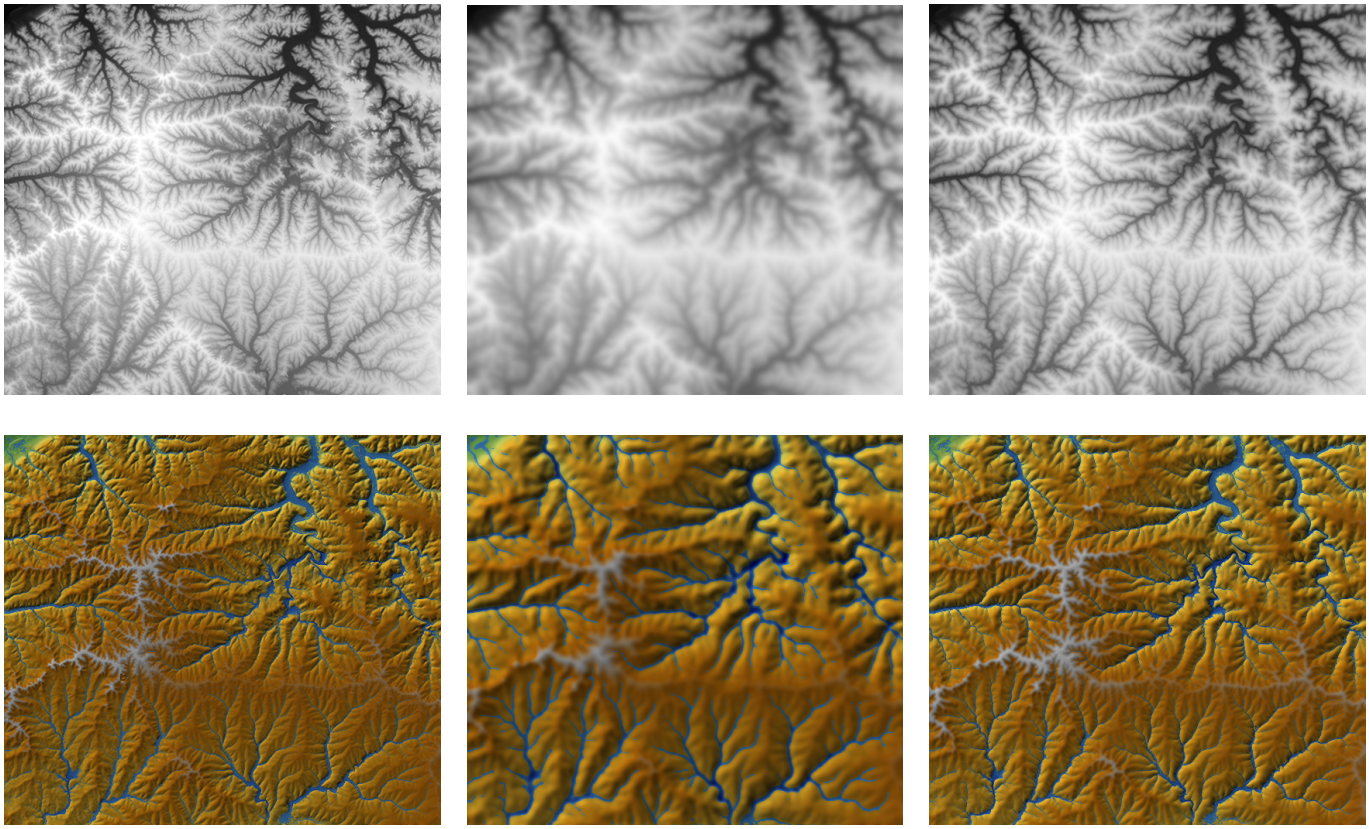

Figure 8: Common boundary conditions: Neumann for θ and h, uniform uplift. From left to right: initial landscape, final landscape obtained with ($r = 1 \times 10^{-7}$, $N_i = 500$, $c = 10.0 \times 10^{-3}$, $e = 0$, $s = 0$) $\mathcal{N}_g = 1.206$, $\mathcal{N}_i = 0.087$, final landscape obtained with ($r = 1 \times 10^{-7}$, $N_i = 500$, $P = 5\%$, $c = 10.0 \times 10^{-3}$, $\mathbf{m} = 0.6$, $\mathbf{n} = 1.05$, $e = 15$, $s = 0.2 \times 10^{-6}$) $\mathcal{N}_g = 0.818$, $\mathcal{N}_i = 0.087$. From top to bottom: normalized visualization, false color visualization. One can be convinced that landscape stability is improved with the set of optimized parameters. The optimized landscape is closer to the original than the one obtained by applying only a creep. This gives evidence that it is possible to compensate the creep effect by carefully tuning the erosion and the sedimentation coefficients.

Stopping criteria: in addition to those parameters, only the maximum number of iterations for the main equations (N_i) can be adjusted. Indeed, as we seek for a landscape creation, the amount of landscape to remove is irrelevant to stop the evolution.

Automation: as described in Section 5.0.1, the automatic uplift option is proposed with two different modes: local and uniform.

Synthesis: the core part of this mode can be divided into two main parts. The first one is to get a rough/young version of an input landscape by blurring it ($\mathcal{B}_i$) and adding Gaussian noise ($\mathcal{G}_i$), as shown in Figure 9. By blurring the reference landscape, all the finest water network and smaller valleys disappear. Then a noise is added to help initiate channel creation.

Refinement: as described in Section 5.1, it is possible to get enhanced results by first applying a zoom-in on the input landscape, and obtain the final result with a zoom-out. Especially, as described in Figure 12, this option provides results with thinner valleys and details.

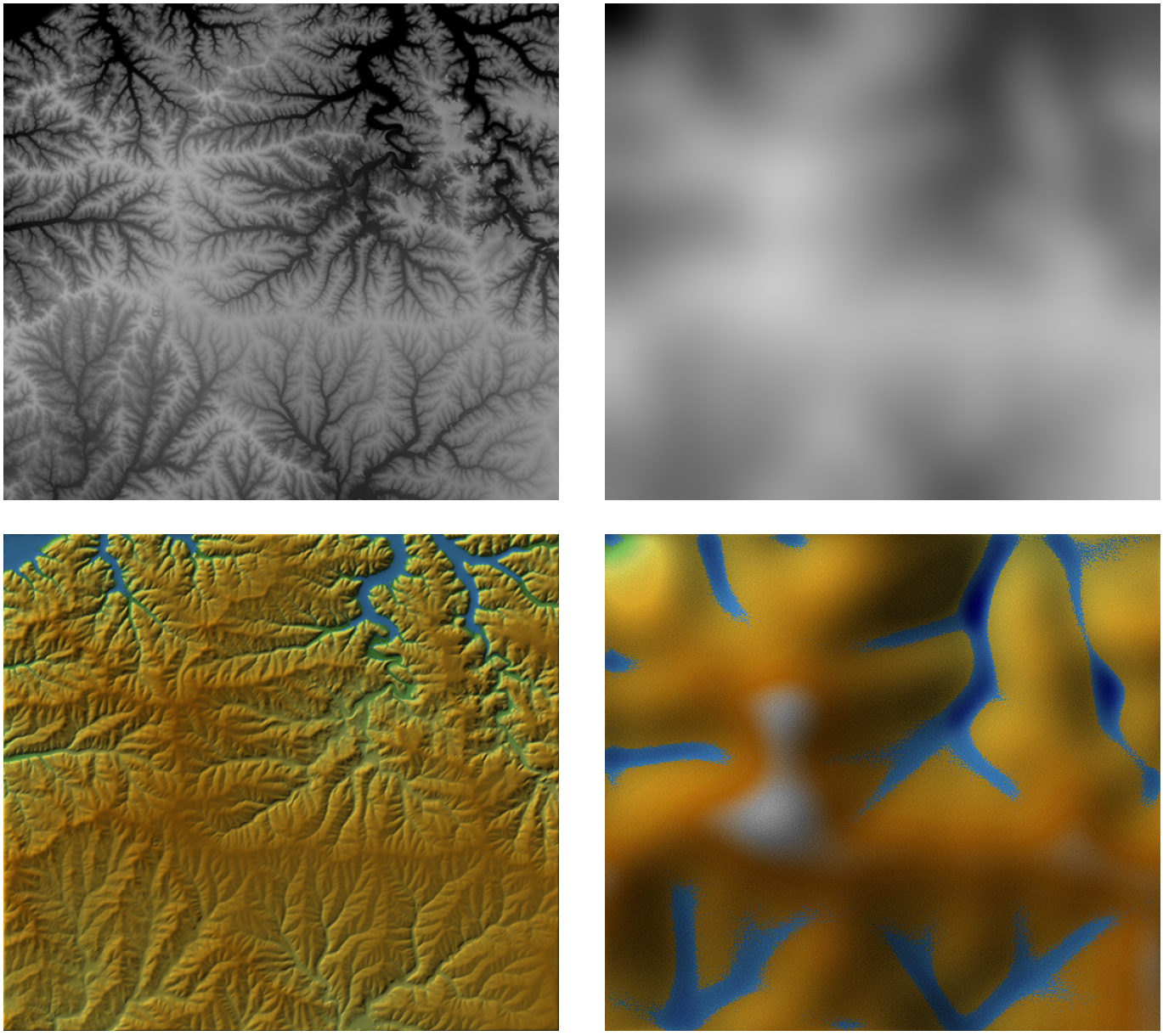

Figure 9: Illustration of the rejuvenation of a landscape by adding blur and noise. From left to right: input original landscape, output landscape with initial Gaussian blur of standard deviation 25 and normalized Gaussian noise of standard deviation 10. From top to bottom: normalized visualization, false color visualization.

Starting from the rough landscape, two additional simulation tools improve the realistic aspect of landscape synthesis during the main part of the algorithm. First, an additional noise is added every few hundreds iterations. Second, at each iteration of the three equations (erosion, creep, water and sediment transport), more iterations of only the water and sediment transport are run to handle the excess of water.

Figure 10 shows an example of landscape synthesis with parameter values learned from Section 5.0.2.

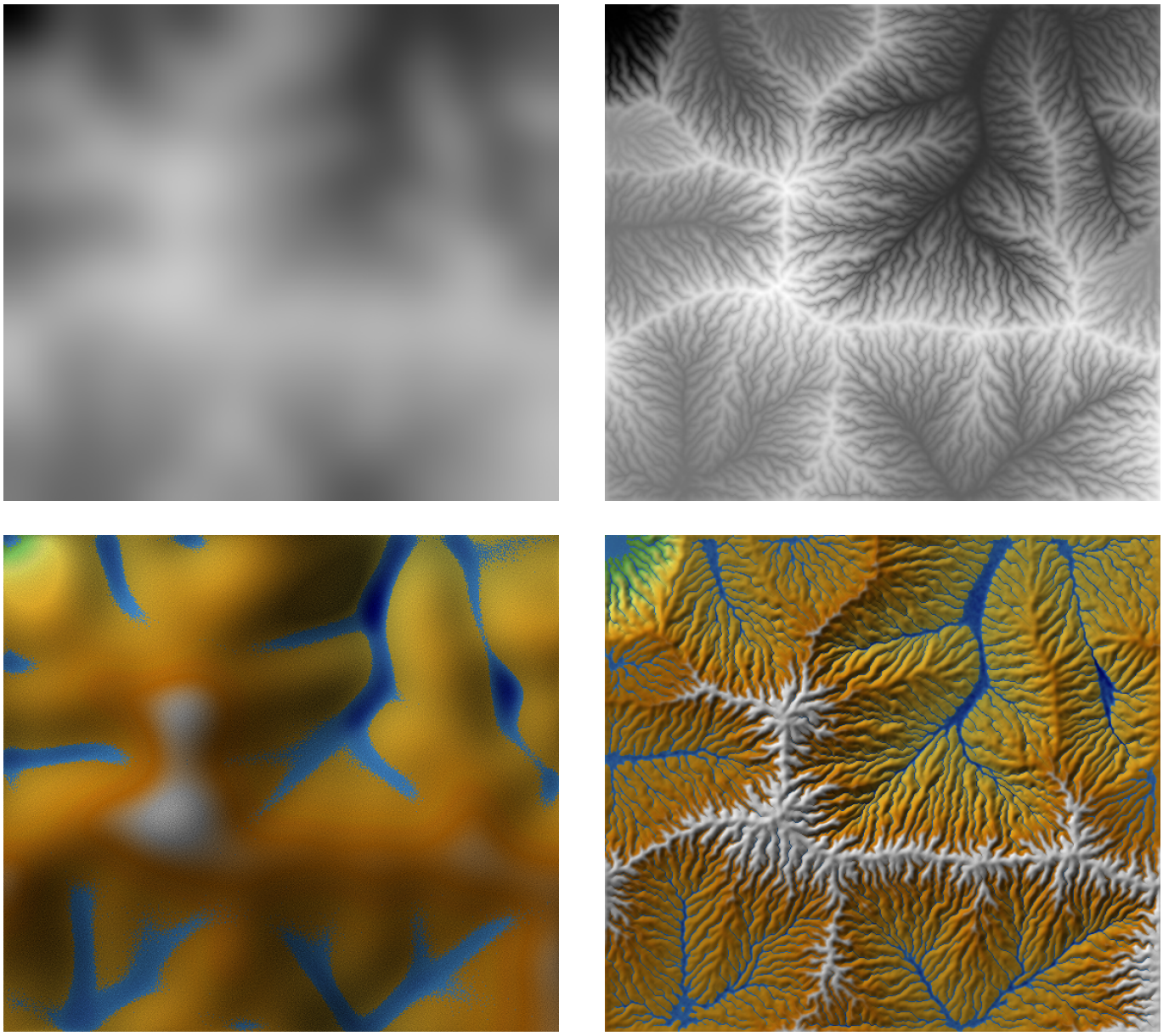

Figure 10: Example of landscape synthesis, with the following set of parameters: $r = 1.0 \times 10^{-7}$, $e = 5.0$, $c = 5.0 \times 10^{-3}$, $s = 0.2 \times 10^{-6}$, $\mathbf{m} = 0.5$, $\mathbf{n} = 1.0$, $N_i = 2000$, initial Gaussian noise: 10×10^{-2}, initial Gaussian blur: $\sigma = 25$, add 1.0×10^{-2} Gaussian noise every 100 iterations, run 10 iterations of water and sediment transport with $r = 1.0 \times 10^{-7}$, with an automatic local uplift. From left to right: initial rough landscape, final landscape synthesized. From top to bottom: normalized visualization, false color visualization.

5.1 Refinement

One issue generally raised in the Landscape Synthesis mode, and in a lesser extent in the Landscape Evolution Model mode, is the scale of the details and the spacing between the valleys, as illustrated in Figure 11. One simple solution to address this problem is to apply the following steps:

1. Apply a zoom-in of a given factor f_z to the input landscape image;
2. Apply the whole algorithm over this zoomed image;
3. Apply a zoom-out of a factor $\frac{1}{f_z}$ over all the output images.

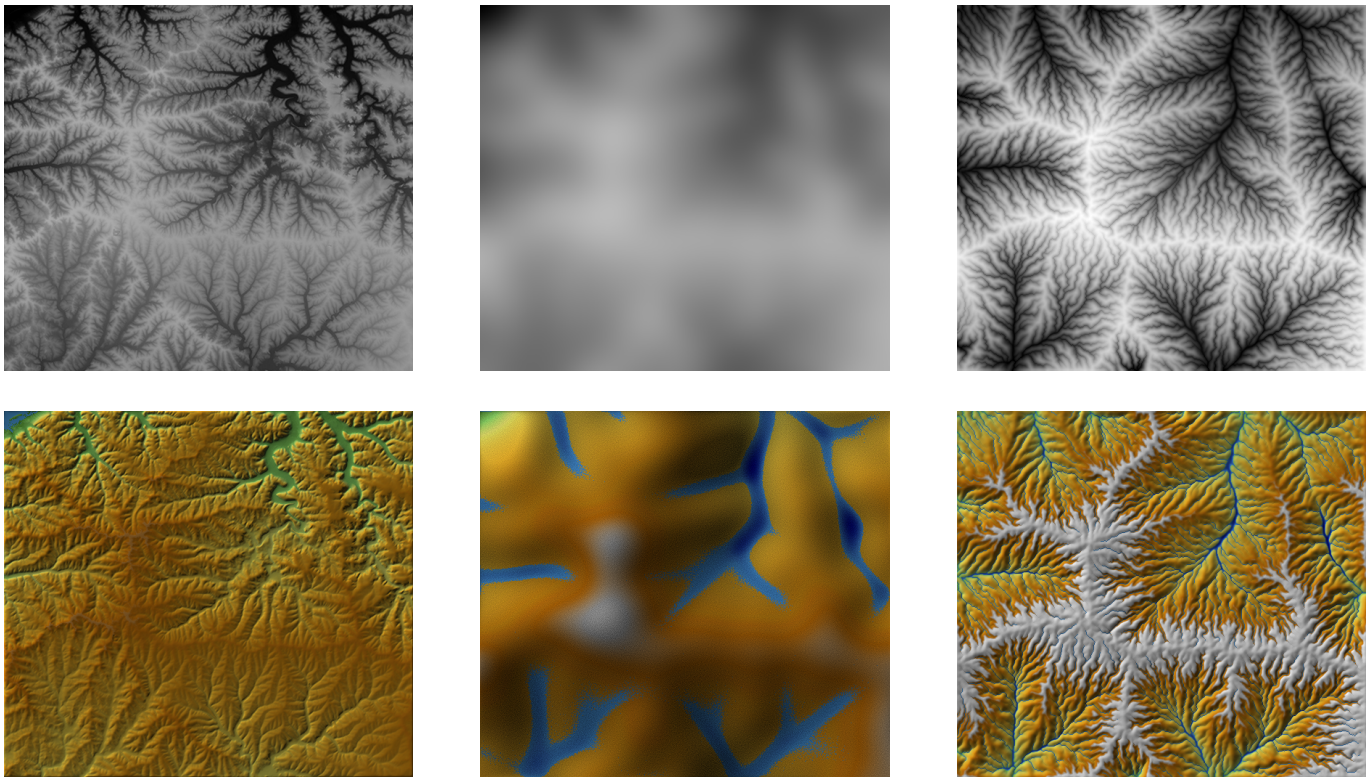

Figure 11: Illustration of the non-reducible gap between valleys for the Landscape Synthesis mode. From left to right: original landscape, blurred and noisy landscape, evolved landscape with the following set of parameters :$\{N_i = 2000, r = 1.0 \times 10^{-7}, e = 5.0, c = 5.0 \times 10^{-3}, s = 0.2 \times 10^{-6}, \mathbf{m} = 0.5, \mathbf{n} = 1.0, \mathcal{G}_i = 10.0 \times 10^{-2}, \mathcal{B}_i = 25.0\}$, Dirichlet conditions for water and Neumann conditions for landscape. From top to bottom: normalized visualization, false color visualization.

By using this technique, a visible gain is obtained on the details of the evolved landscape, in particular the gap between the valleys is reduced. Figure 12 shows the advantage of using this zoom-based technique for different values of zoom.

5.2 Multiscale Experiments: La Réunion at Three Resolutions

This section presents two experiments for a comparative study of numerical LEM simulation on the same landscape, La Réunion, at three very different resolutions. The results show a coherence

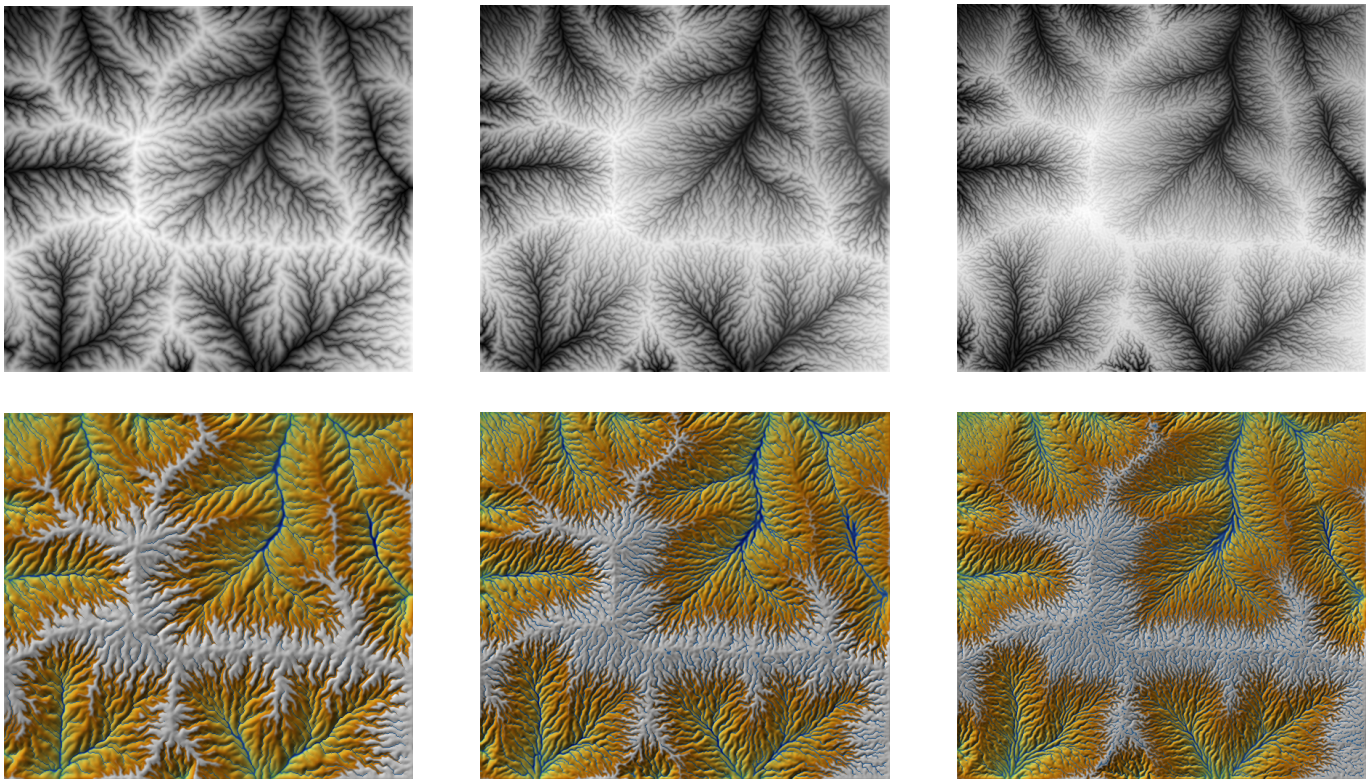

Figure 12: Illustration of the non-reducible gap between valleys for the Landscape Synthesis mode. The following set of parameters has been used for all images :$\{N_i = 2000, r = 1.0 \times 10^{-7}, e = 5.0, c = 5.0 \times 10^{-3}, s = 0.2 \times 10^{-6}, \mathbf{m} = 0.5, \mathbf{n} = 1.0, \mathcal{G}_i = 10.0 \times 10^{-2}, \mathcal{B}_i = 25.0\}$, Dirichlet conditions for water and Neumann conditions for landscape. From left to right: zoom = 1.0, zoom = 1.5, zoom = 2.0. From top to bottom: normalized visualization, false color visualization.

of the aspect of the evolved landscapes at all resolutions. But they also show that the numerical landscape evolution leads to the emergence of a very fine resolution network. So the better the initial resolution, the better the prediction. We have shown the results of LEM simulations on an SRTM 90m La Réunion DEM in Figure 13. Figures 14 and 15 show analogue results at much higher resolutions on DEMs obtained from Pléiades stereo pairs by applying a stereo reconstruction algorithm yielding respectively DEMs at 12 meters/pixel and 3m/pixel using the method sketched in [9].

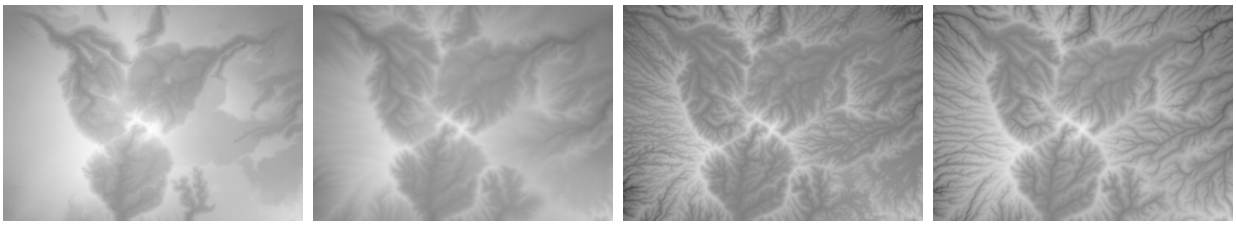

Figure 13: This experiment illustrates the variety of morphologies that can be reached from an initial DEM by varying the parameters of a three-equation model. From top to bottom and from left to right, we have the initial landscape and then the evolved landscape for $(r, \epsilon, c, p) = (5, 1, 10, 20)$, $(1, 1, 2, 20)$, $(1, 1, 0.5, 30)$ with $\mathbf{m} = 0.5$, $\mathbf{n} = 2\mathbf{m}$. Notice how some of the evolutions maintain a similar landscape, while others create new basins and rivers and evolve it toward a different morphology. For example the second result and the third are obtained with the same final amount of scrapped land, 20%. But the first is still very similar to the original, showing a slow morphological evolution, while the second has created or expanded basins, as the landscape evolves to a mature form. In the last experiment, the exponents are changed to $\mathbf{m} = 0.6$, $\mathbf{n} = 2\mathbf{m}$. Observe that this modification seems to modify valley spacing on the left slope. Data: SRTM La Réunion, Piton des Neiges. See Section 2 for more details on the equations and parameters.

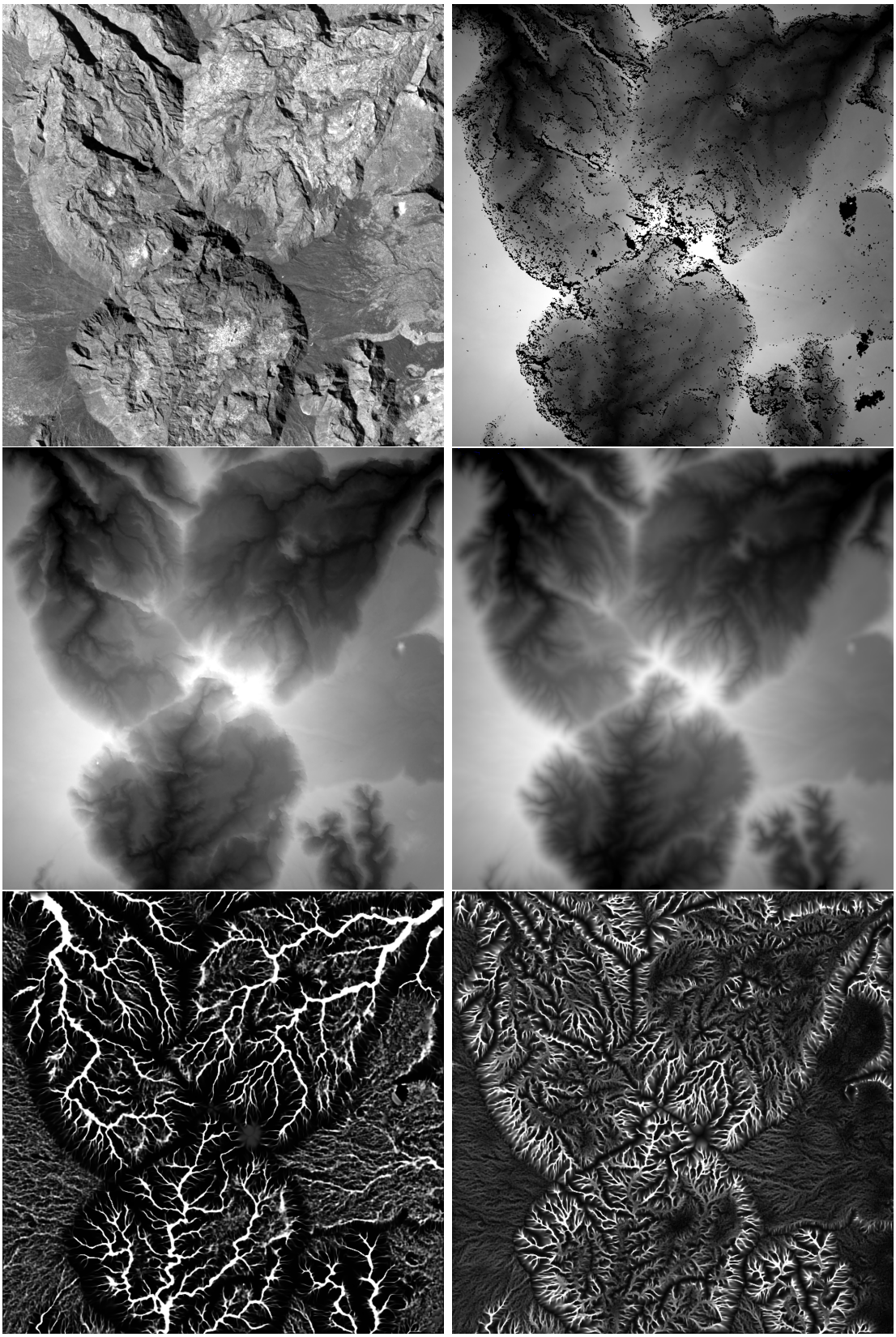

Figure 14: From top left to bottom right: Piton des Neiges, La Réunion, image at circa 12 meters/pixel by numerical zoom out, obtained from Pléiades (CNES 2014, Distribution Astrium Services); DEM of the same; same DEM, interpolated to remove holes left by the stereomatching in dark regions; a plausible evolution, where 10% of the terrain has been removed by erosion. The parameters in the equation are rain $r = 1$, erosion $\epsilon r = 1$, creep $c = 16$, sedimentation $s = 1$; water network at the end of the evolution $\theta(x, y)$; sediment contained in water $\lambda(x, y)$, revealing the fine structure of the hydrological network.

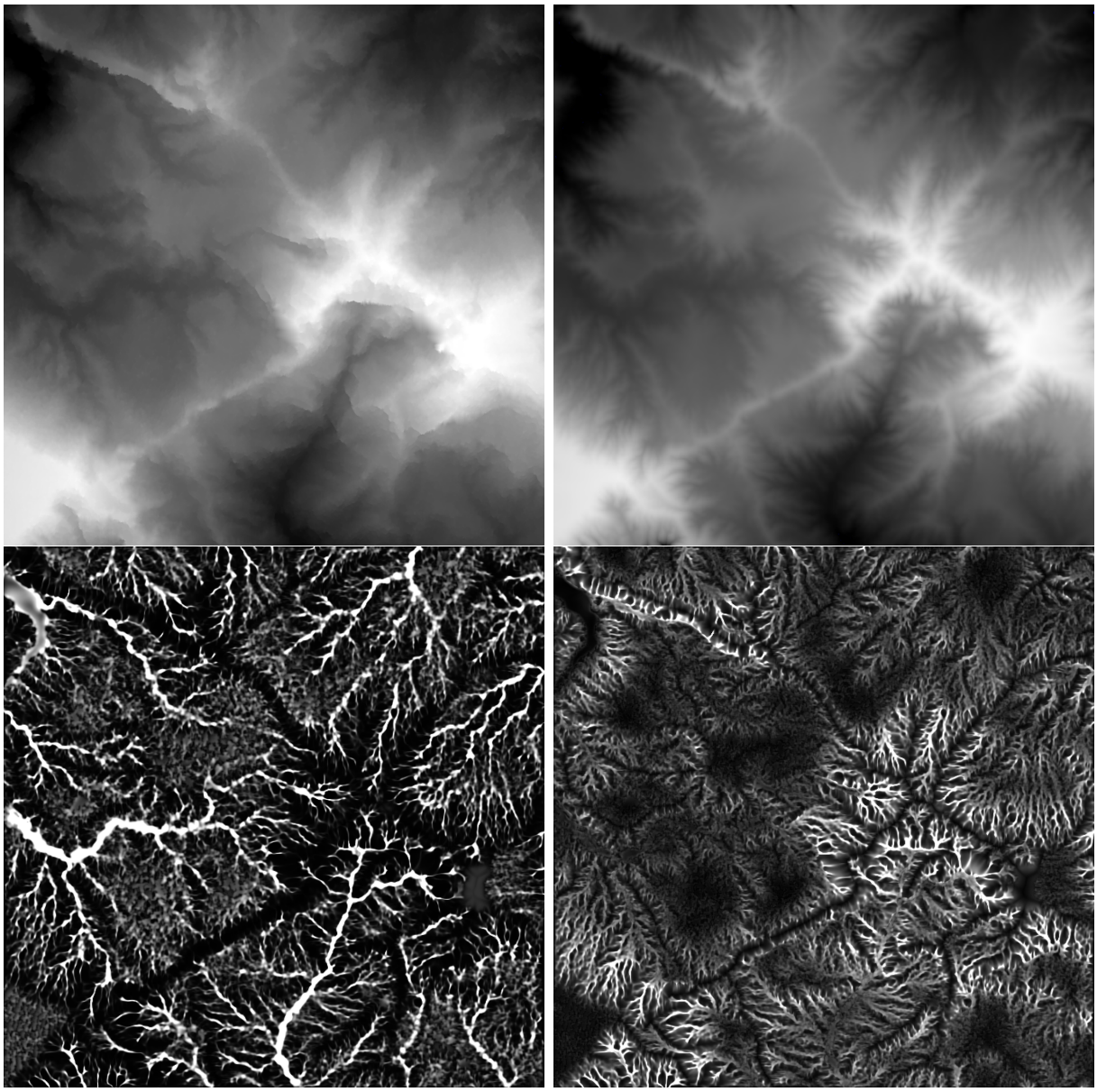

Figure 15: From top left to bottom right: Piton des Neiges, La Réunion, DEM at circa 4 meters/pixel obtained from Pléiades (CNES 2014, Distribution Astrium Services); a plausible evolution toward a stable landscape, after 10% of the terrain has been removed by erosion. The parameters in the equation are rain $r = 1$, erosion $\epsilon r = 1$, creep $c = 32$, sedimentation $s = 1$; water network at the end of the evolution $\theta(x, y)$; sediment contained in water $\lambda(x, y)$, revealing the fine structure of the hydrological network.

In Figure 14, we display first one image from a stereo pair of Piton des Neiges, La Réunion. This image at about 12 meters/pixel is obtained by numerical zoom out from Pléiades (CNES 2014, Distribution Astrium Services). With this zoom out the image stereo pair has become virtually noiseless. Thus the automatic DEM reconstruction chain S2P [9] yields a fairly dense reconstruction. Nevertheless, interpolation remains necessary to remove holes left by the stereo-matching in dark regions (not lit directly by the sun) and in zones with very fine texture, which both correspond to a low SNR, making block matching fail.

The next image shows a plausible DEM evolution, which is stopped when 10% of the terrain has been removed by erosion. The parameters in the equation are rain $r = 1$, erosion $e = 1$, creep $c = 16$, sedimentation $s = 1$; The images of the last row of the figure show the water network at the end of the evolution $\theta(x, y)$ and the sediment density contained in water $\lambda(x, y)$, revealing still better the fine structure of the hydrological network, as sediment has high concentration in the fine network.

This experiment illustrates the sensitivity of numerical landscape evolution to the initial resolution; clearly the simulated network tends to increase the DEM resolution by recreating a fine channel network that was missing in the original DEM. This suggests that much will be gained by increasing still the resolution, as it is by now possible with Pléiades or Worldview Earth stereo imaging, for example. The nominal resolution of Pléiades being 0,7m, nothing prevents from simulating landscape evolution at this resolution. This makes sense, because water runoff is very shallow except in big rivers, and therefore definitely affected by landscape roughness at the very scale of Pléiades observation.

We shall be contented here to compare our previous landscape evolution at 12m to the result of the same LEM applied this time to a circa 4 meter resolution stereo pair (Figure 15). The second stereo pair of Piton des Neiges, La Réunion, DEM was obtained by zoom out from the same original stereo Pléiades pair (CNES 2014, Distribution Astrium Services). The DEM therefore gets a three times finer resolution and was again interpolated to remove the holes where stereo matching failed. Again, a plausible evolution toward a stable landscape is shown, after 10% of the terrain has been removed by erosion. The parameters in the equation are rain $r = 1$, erosion $\epsilon r = 1$, creep $c = 32$, sedimentation $s = 1$. We display the landscape evolution results in the same format as in Figure 14: water network at the end of the evolution $\theta(x, y)$; sediment contained in water $\lambda(x, y)$, revealing the still finer structure of the hydrological network.

While both networks look compatible, it is clear that the landscape evolution at a finer scale is different. If follows from this observation, as we already anticipated, that applying LEMs at 50 centimeters scale is probably necessary to obtain a realistic simulation and get to the critical scales at stake in landscape evolution. This without any doubt requires a huge but feasible numerical machinery, and it requires stereo pairs at a Pléiades resolution, or finer.

Notation

Model parameters

- c: creep rate;
- r: amount of rain added at each iteration;
- e: erosion rate;
- s: sedimentation rate;
- $\mathbf{m}$: exponent for the landscape evolution scheme;
- $\mathbf{n}$: exponent for the landscape evolution scheme

Numerical parameters

- δ_t: time step;
- P: percentage of land to remove;
- $\mathcal{B}_i$: initial blur applied to the landscape for the Landscape Synthesis mode;
- $\mathcal{G}_i$: initial noise applied to the landscape for the Landscape Synthesis mode;
- θ_i: initial water level;
- θ_0: ocean level;
- N_b: number of iterations for the water initialization;
- N_i: maximal number of iterations for the main loop;
- τ_θ: threshold coefficient for water visualization.

Notation

- h: generic landscape image;
- θ: generic water image;
- λ: generic sediment concentration image;
- m_c: current mass of the landscape;
- m_t: target mass of the landscape to reach depending on P;
- NC: width of the image;
- NR: height of the image;
- τ: transfer images (temporary);
- δ: steep;
- δ_e: amount to erode for the landscape evolution scheme;
- δ_s amount of sediment for the landscape evolution scheme;
- $\mathcal{N}_g$: gradient-based norm;
- $\mathcal{N}_i$: image-based norm.

Acknowledgment

Work supported by AHA Tosca project and MISS project, Centre National d'Etudes Spatiales, European Research Council by Advanced Grant Twelve labours, Centre National de la Recherche Scientifique, Ecole Normale Supérieure Paris-Saclay, Office of Naval Research, Duke University, Direction Générale de l'Armement by ASTRID Stéréo project, Ministère de l'Enseignement Supérieur et de la Recherche.

References

[1] J. Braun and M. Sambridge, *Modelling landscape evolution on geological time scales: a new method based on irregular spatial discretization*, Basin Research, 9 (1997), pp. 27–52. https://doi.org/10.1046/j.1365-2117.1997.00030.x.

[2] J. Braun and S. D Willett, *A very efficient $O(n)$, implicit and parallel method to solve the stream power equation governing fluvial incision and landscape evolution*, Geomorphology, (2012), pp. 170–179. https://doi.org/10.1016/j.geomorph.2012.10.008.

[3] S. Carretier and F. Lucazeau, *How does alluvial sedimentation at range fronts modify the erosional dynamics of mountain catchments?*, Basin Research, 17 (2005), pp. 361–381. https://doi.org/10.1111/j.1365-2117.2005.00270.x.

[4] A. Chen, J. Darbon, G. Buttazzo, F. Santambrogio, and J-M. Morel, *On the equations of landscape formation*, Interfaces and Free Boundaries, 16 (2014), pp. 105–136. https://doi.org/10.4171/IFB/315.

[5] A. Chen, J. Darbon, and J-M. Morel, *Landscape evolution models: A review of their fundamental equations*, Geomorphology, 219 (2014), pp. 68–86. https://doi.org/10.1016/j.geomorph.2014.04.037.

[6] A. Crave and P. Davy, *A stochastic "precipiton" model for simulating erosion/sedimentation dynamics*, Computers & Geoscience, 27 (2001), pp. 815–827. https://doi.org/10.1016/S0098-3004(00)00167-9.

[7] W. E. H. Culling, *Analytical theory of erosion*, The Journal of Geology, 68 (1960), pp. 336–344. https://doi.org/10.1086/626663.

[8] W. M. Davis, *The convex profile of bad-land divides*, Science, (1892), p. 245. https://doi.org/10.1126/science.ns-20.508.245.

[9] C. de Franchis, E. Meinhardt-Llopis, J. Michel, J-M. Morel, and G. Facciolo, *Automatic digital surface model generation from Pléiades stereo images*, Revue Française de Photogrammétrie et de Télédétection, 208 (2014).

[10] W. E. Dietrich and J. T. Perron, *The search for a topographic signature of life*, Nature, 439 (2006), pp. 411–418. https://doi.org/10.1038/nature04452.

[11] N. F. Fernandes and W. E. Dietrich, *Hillslope evolution by diffusive processes: The timescale for equilibrium adjustments*, Water Resources Research, 33 (1997), pp. 1307–1318. https://doi.org/10.1029/97WR00534.

[12] P. Gauckler, *Etudes Théoriques et Pratiques sur l'Ecoulement et le Mouvement des Eaux*, Comptes Rendus de l'Académie des Sciences, Paris, 64 (1867), pp. 818–822.

[13] G. K. Gilbert, *The convexity of hilltops*, The Journal of Geology, 17 (1909), pp. 344–350. https://doi.org/10.1086/621620.

[14] G. K. Gilbert and C. E. Dutton, *Report on the Geology of the Henry Mountains*, Govt. print. off., 1877.

[15] G. R. Hancock, T. J. Coulthard, C. Martinez, and J. D. Kalma, *An evaluation of landscape evolution models to simulate decadal and centennial scale soil erosion in grassland catchments*, Journal of Hydrology, 398 (2011), pp. 171–183. https://doi.org/10.1016/j.jhydrol.2010.12.002.

[16] A. D. Howard, *A detachment-limited model of drainage basin evolution*, Water Resources Research, 30 (1994), pp. 2261–2285. https://doi.org/10.1029/94WR00757.

[17] N. Limare, J.L. Lisani, J-M. Morel, A.B. Petro, and C. Sbert, *Simplest Color Balance*, Image Processing On Line, 1 (2011), pp. 297–315. https://doi.org/10.5201/ipol.2011.llmps-scb.

[18] W. Luo, *LANDSAP: a coupled surface and subsurface cellular automata model for landform simulation*, Computers & Geoscience, 27 (2001), pp. 363–367.

[19] G. E. Moglen, E. A. B. Eltahir, and R. L. Bras, *On the sensitivity of drainage density to climate change*, Water Resources Research, 34 (1998), pp. 855–862. https://doi.org/10.1029/97WR02709.

[20] J. D. Niemann, N. M. Gasparini, G. E. Tucker, and R. L. Bras, *A quantitative evaluation of Playfair's law and its use in testing long-term stream erosion models*, Earth Surface Processes and Landforms, 26 (2001), pp. 1317–1332. https://doi.org/10.1002/esp.272.

[21] K. Paik, *Simulation of landscape evolution using a global flow path search method*, Environmental Modelling & Software, 33 (2012), pp. 35–47. https://doi.org/10.1016/j.envsoft.2012.01.005.

[22] A. Refice, E. Giachetta, and D. Capolongo, *SIGNUM: A matlab, TIN-based landscape evolution model*, Computers & Geoscience, 45 (2012), pp. 293–303. https://doi.org/10.1016/j.cageo.2011.11.013.

[23] G. Simpson and F. Schlunegger, *Topographic evolution and morphology of surfaces evolving in response to coupled fluvial and hillslope sediment transport*, Journal of Geophysical Research, 108 (2003), p. 2300. https://doi.org/10.1029/2002JB002162.

[24] G. E. Tucker and R. L. Slingerland, *Erosional dynamics, flexural isostasy, and long-lived escarpments: A numerical modeling study*, Journal of Geophysical Research, 99 (1994), pp. 12229–12243. https://doi.org/10.1029/94JB00320.

[25] G. Willgoose, R. L. Bras, and I. Rodriguez-Iturbe, *A coupled channel network growth and hillslope evolution model: 2. Nondimensionalization and applications*, Water Resources Research, 27 (1991), pp. 1685–1696. https://doi.org/10.1029/91WR00936.