CHAPTER 1

Equations for geomorphology

Overview of Badlands formulation

Peruvian landscape from the European Space Agency
The development of the **Badlands** model (Basin and landscape dynamics) is made in the context of the growing number of landscape evolution models (or surface processes models - SPMs) that have been proposed during the last decades. Many geomorphological applications have demonstrated the usefulness of these models, whose predictions help researchers test simple to complex hypotheses on the nature of landscape evolution. SPMs also provide connection between small-scale, measurable processes and their long-term geomorphic implications (Tucker et al., 2001b). Below, we give a brief overview of SPMs, their history and their conceptual basis.

**Surface Processes Models**

The roots of landscape evolution theory can be found in the pioneering work of Gilbert (1877), who proposed a set of hypotheses to relate various landforms to the mechanisms of weathering, erosion and sediment transport. The first quantitative models appeared later in the 1960s (e.g., Culling, 1960; Scheidegger, 1961; Ahnert, 1970; Kirkby, 1971).
These models formalize the concepts of Gilbert (1877) to the development of hillslope profiles. A few years later, these models were extended to two dimensions, although still focused on hillslope morphology (e.g., Ahnert, 1976; Kirkby, 1986). During the last two decades, as computers continued to get faster, a number of sophisticated numerical SPMs have been developed, mainly focusing on watershed or mountain belt evolution (e.g., Willgoose et al., 1991; Howard, 1994; Braun and Sambridge, 1997; Densmore et al., 1998; Tucker et al., 2001b; Crave and Davy, 2001; Coulthard et al., 2002). Both hillslope and fluvial processes are involved in these models, which differ from each other by the parameterization of these processes and their numerical resolution. For a complete overview of surface processes modeling and the recent SPMs, the reader is referred to the numerous reviews on that topic (Beaumont et al., 1999; Coulthard, 2001; Martin and Church, 2004; Willgoose, 2005; Codilean et al., 2006; Bishop, 2007; Tucker and Hancock, 2010).

### REFERENCES


Section 2

Badlands: general equations

**Key Points**

1. Continuity of mass.
2. Overland flow model.
3. Diffusion law for hillslope processes.
4. Flexural isostasy
5. Sediment compaction

**Continuity of mass**

In the simplest case where there is no distinction between a regolith layer and the bedrock underneath, the mass continuity equations for a column of soil or rock is expressed as:

\[ \frac{\partial z}{\partial t} = U - \nabla \cdot \mathbf{q}_s \]

where the elevation \( z \) (m) is measured vertically, \( \mathbf{q}_s \) is the total downhill soil flux, \( \nabla \cdot \) is the spatial divergence operator and \( U \) (m/yr) is a source term that can either represent the rate of incision of channel streams at the hillslope boundaries or uniform uplift. This is one of several variations; for discussion of others, see Tucker and Hancock (2010).

**Sediment transport modules**

To describe the rates of sediment erosion/transport/deposition, we use the following laws

**Overland flow: Detachment-limited model**

The soil transport rate per unit width by flowing water, \( q_r \), is modeled as a power function of topographic gradient \( \nabla z \) and surface water discharge per unit width \( q_w \) (m\(^2\)/yr):

\[ q_r = -\kappa_r(q_w)^m(\nabla z)^n \]

Note that \( (\nabla z)^n \) is short-hand for \( ||\nabla z||^{n-1}\nabla z \), as soil flux is oriented in the direction of the topographic gradient.

This detachment-limited incision rate, which is calculated as a power-law function of fluvial discharge only applies where channel slope is positive, (e.g., Howard, 1994; Whipple and Tucker, 1999):
\[
\frac{\partial z}{\partial t} = - \kappa_r(q_w)^m(\nabla z)^n
\]

This expression corresponds to a simplified form of the usual expression of sediment transport by water flow (Howard, 1980), in which the transport rate is assumed to be equal to the local carrying capacity, which is itself a function of boundary shear stress or stream power per unit width (Tucker and Hancock, 2010). We consider additionally no threshold for particle entrainment. The exponents \(m\) and \(n\) have values between 1 and 2 (Willgoose et al., 1991; Howard et al., 1994; Tucker and Bras, 1998; Prosser and Rustomji, 2000; Bogaart et al., 2003a).

**HILLSLOPE: SIMPLE CREEP OR LINEAR DIFFUSION**

The parameterization of hillslope transport is based on a linear dependence to the topographic gradient. As pointed out by Fernandes & Dietrich (1997), this linear law has in fact been used to represent a variety of transport processes such as creep (e.g., Culling, 1963; Armstrong, 1987), biogenic activity (e.g., Dietrich et al., 1987; Heimsath et al., 1999, 2002) or rain splash (e.g., De Ploey and Savat, 1968; Dunne et al., 2010). Downslope simple creep is commonly regarded as operating in a shallow superficial layer (Braun et al., 2001) and is defined as:

\[
q_d = - \kappa_d \nabla z
\]

Note that because of the multi-process parameterization of soil transport, the coefficient \(\kappa_d\) is also scale-dependent (Dietrich et al., 1995), like the \(\kappa\)-scale parameters of the other transport laws included in our model.

**FLEXURAL ISOSTASY**

The equations governing elastic deformation are given by:

\[
D \nabla^2 \nabla^2 w(x, y) = - (\rho_m - \rho_{fill}) g w(x, y) + q(x, y)
\]

where \(D\) is the flexural rigidity of an elastic plate, \(w(x, y)\) is the vertical deflection of the plate; \(\rho_m\) is the density of mantle; \(\rho_{fill}\) is the density of the material occupying the space created by the flexural subsidence (air/water/sediment). \(q(x, y)\) is the net force per unit area exerted by the applied load,

\[
q(x, y) = \rho_{load} g h_{load}(x, y)
\]

\(\rho_{load}\) and \(h_{load}\) are the density and physical height of the load, respectively; \(g\) is the gravitational acceleration.

Various approaches to the numerical solution of the flexural deformation of an elastic plate have been described in the literature. Hodgetts et al. (1998) solved the 3D equation in the wave number domain via the 2D Fourier transforms. Another popular technique is to split the problem into two coupled Poisson equations for the deformation and the curvature (\(w\) and \(u\)). In Badlands, the adopted approach consists first in interpolating the load from the unstructured grid to an uniform one. Then 2 methods could be used by the user:

1. a 13-point second-order grid approximation,
2. a four-order multi-grid solutions to the biharmonic equation (Atlas et al., 1998).

The model is capable of simulating both local and regional isostatic compensation (Li et al., 2004).

**POROSITY AND COMPACTION**

For basin-scale applications, compaction is an important phenomena which is the consequence of changes in sediment porosity due to overburden and pore fluid pressure. As for now Badlands only takes into account a unique time of sediment mixture and compaction during the
simulation is calculated using a look-up table containing porosity values as a function of effective stress. At given time interval and for each TIN node, the technique consists in

- iteratively computing the lithostatic pressure for all sedimentary time layers,
- deriving each sedimentary layer averaged porosity value from the look-up table.
- summing each layers compaction changes.

REFERENCES


Badlands: spatial and temporal framework

**KEY POINTS**

1. Spatial framework, finite volume approach.
2. Parallelization strategy.
3. Temporal framework.
4. 3D displacement: mesh geometry.

### SPATIAL FRAMEWORK

Equations detailed in previous section describe the evolution of soil thickness and surface elevation. However, solving these partial differential equations (PDEs) is non-trivial and numerical methods are needed to perform their integration in space and time (for a given set of initial and boundary conditions). This step requires the discretization of the terrain surface, $z(x, y)$, and soil thickness, $h(x, y)$, into a finite number of 'hillslope elements', i.e., a lattice of connected points $(x_i, y_i)$ and their associated cells (i.e., the part of the hillslope they represent) at which the solution of the PDEs is calculated. Here, we describe how the topography is discretized in Badlands and how the continuity of mass is applied within each element.

### DELAUNAY TRIANGULATION AND VORONOI DIAGRAM

Here, we choose to use an irregular spatial discretization scheme similar to the CASCADE (Braun and Sambridge, 1997) and CHILD (Tucker et al., 2001a,b) models to numerically solve the geomorphic equations. The computational mesh is created in three stages: (1) The 2D planimetric space is first discretized as a set of points in any arbitrary configuration, (2) these points are then connected using the Delaunay triangulation (Voronoi, 1908; Delaunay, 1934) to form the nodes of a Triangulated Irregular Network (TIN), and (3) the Voronoi diagram, which corresponds to the dual graph of the Delaunay's TIN, is constructed. Initial values of surface elevation and soil thickness are then assigned to each node. These values will be updated every time step of a Badlands simulation.

The Delaunay triangulation and the Voronoi tessellation are well established in the field of computational geometry (e.g., Fortune, 1992; Sambridge et al., 1995; Du, 1996). The Delaunay triangulation of a set of irregularly spaced points may be defined as the unique triangulation for
APPLICATION OF THE FINITE-VOLUME APPROACH

As pointed out by Tucker et al. (2001b), the dual Delaunay-Voronoi framework lends itself to a numerical solution of the continuity equation using the Finite-Volume (FV) approach (Eymard et al., 2000). Applying this approach in our two-dimensional problem consists of integrating the continuity equation over small areas (‘finite-volume’ cells) surrounding the nodes of the mesh, which here correspond to the Voronoi cells associated to each node. Hereafter, we illustrate the application of the FV method to the node $i$ and its associated Voronoi cell of surface area $\Omega_i$. Using the divergence theorem, the integral over $\Omega_i$ of the divergence of the soil flux, $\nabla \cdot q_s$, can be expressed as the integral of the soil fluxes at the boundary of the Voronoi cell:

$$\int_{\Omega_i} \nabla \cdot q_s d\Omega_i = \int_{w_i} n \cdot q_s d\lambda_i \approx \sum_{j=1}^{n_{b_i}} w_{ij} q_{s,ij}$$

where $w_i$ is the total length of the boundary of the Voronoi cell, $\mathbf{n}$ is the unit vector normal to the boundary and pointing outward, $n_{b_i}$ is the number of natural neighbours connected to the node $i$, $w_{ij}$ is the width of the edge shared by adjacent Voronoi cells associated to the node $i$ and its neighbour $j$, and $q_{s,ij}$ is the total bulk volumetric soil flux across this edge per unit width.

The flux $q_{s,ij}$ is calculated by summing the fluxes defined in previous section, we note that this flux depends on the local topographic gradient across -and normal to- the shared Voronoi edge between the nodes $i$ and $j$. This gradient is approximated as the topographic gradient between the nodes themselves:

$$\| (\nabla z)_{ij} \| \approx S_{ij} = \frac{z_j - z_i}{\lambda_{ij}}$$
where \( z_i \) and \( z_j \) are the surface elevation associated to the nodes \( i \) and \( j \), respectively, and \( \lambda_{ij} \) is the planimetric distance between these nodes. Considering the downslope movement of soil, \( q_{s,ij} \) is positive if the net soil flux is from \( i \) to \( j \), and negative if the net flux is from \( j \) to \( i \). Note also that FV methods are conservative, i.e., that the flux entering (or leaving) a given cell equals the flux leaving (or entering) the adjacent cell (\( q_{s,ij} = -q_{s,j,i} \)). Finally, integrating both sides of the continuity equation over \( \Omega_i \) gives:

\[
\frac{dh_i}{dt} = \kappa P_{s,i} - \frac{1}{\Omega_i} \sum_{j=1}^{nb} w_{ij} q_{s,ij}
\]

Intuitively, this equation simply states that a non-zero net soil flux out of the Voronoi cell surrounding the node \( i \) would result in a general decrease of soil thickness within this cell if the net flux is not balanced by the rate of soil production (inversely, a non-zero net flux entering the cell will always result in accumulation of soil within the cell). Note that in FV applications, the cells -and not the nodes of the mesh- are the primitive elements of the spatial discretization. Because the soil fluxes acting inside the cells are not considered, by using the divergence theorem it is only possible to describe the average rate of soil removal or accumulation, \( \frac{dh}{dt} \), within each cell. When mentioning values of soil thickness, \( h_i \), and surface elevation, \( z_i \), assigned to the node \( i \), we thus refer to the averages over the corresponding Voronoi cell.

### Nodes Ordering

Key to the model is to find the order in which one must go through the nodes to compute discharge by adding progressively the contribution of each node to the river discharge (or its contributing area to the total drainage area). To perform this operation, we assume that water goes down the path of the steepest slope (O’Callaghan and Mark, 1984; Gallant and Wilson, 2000). Our algorithm is thus based on the single-flow-direction approximation (each node has a single receiver) (O’Callaghan and Mark, 1984).

The approach we used is based on the work from Braun & Willett (2013) who presented a algorithm to solve very efficiently the stream power equation, regardless of problem geometry or boundary conditions. Its main advantages are that the computational time is linearly dependent on the number of nodes used to discretize the landform.

### Parallelization Approach

To increase the model’s computational efficiency, a two-level mapping parallelism technique is employed.

The first level of parallelism handles the mesh partitioning. Features and functions directly related to the mesh and its geometry are resolved and stored on each partition.

The second level of parallelism is specifically designed to efficiently compute stream network dynamics.

### Mesh Partitioning

At the beginning of a simulation, surface nodes positions representing initial are known. Once the irregular surface has been generated, the set of nodes is partitioned into subdomains. Each subdomain can then be mapped onto a processor of a parallel architecture. For our purposes, we consider a mesh partitioner desirable if it produces subdomains of nearly equal size (where size is measured by number of nodes) and with as few nodes shared between processors (ghost nodes) as is reasonably possible. The model partitioner uses the Hilbert Space-Filling Curve method algorithm, a description of the method is found in Zoltan library (Catalyurek et al., 2007; Devine et al., 2009).
STREAM NETWORK PARTITIONING

The channel network is used as an acyclic, directed graph, such that reaches and junctions are ordered from upstream to downstream direction. Individual sub-basins draining to channel reaches form the basis for partitioning a large domain into smaller units and synchronizing effort in an upstream to downstream order.

The basic tradeoff in the parallelization of the model is determining how much computational effort occurs in a single sub-basin vs. the amount of message passing required between sub-basins. The computational effort in a sub-basin is determined by the number of nodes retained in the TIN and the sub-basin size. Data exchanges between sub-basins consist of lateral surface fluxes through the channel network in upstream to downstream order. Exchanges are performed through message passing using “ghost” cells, which are located at junctions between sub-catchments.

The variables exchanged through ghost cells occur within a model time step according to the sequential order of the processes and the upstream to downstream arrangement of domain nodes. Thus, the computations are synchronized by the structure of the interconnected system of basin nodes. The ghost cells receive, store and send the variables across the TIN nodes when these are located on separate processors.

Sub-basin (or graph) partitioning is generated through the graph partitioning code METIS (Karypis and Kumar, 1999). This approach is based on adjacency in the ordered flow network instead of geo-graphical proximity. The partitioning balances the number of TIN nodes across processors and minimizes the dissections that occur in the channel network through surface ghost cells.

TEMPORAL FRAMEWORK

In Badlands, the time evolution of soil thickness and surface elevation is treated explicitly, i.e., the values at the beginning of the next time step are fully determined by the values at the beginning of the ‘current’ time step. The time integration of the continuity equations gives:

\[
\begin{align*}
    h_i(t + \Delta t) &= h_i(t) + \frac{dh_i}{dt} \Delta t \\
    z_i(t + \Delta t) &= z_i(t) + \left( \frac{dh_i}{dt} - \kappa P_{s,i} + U \right) \Delta t
\end{align*}
\]

where \( \Delta t \) is the duration of the time steps, and where the values of \( P_{s,i} \), \( U \) and all time-dependent variables used to calculate \( \frac{dh_i}{dt} \) (i.e., the variables involved in the expressions of the soil fluxes) are those at the beginning of the current time step.
Using an explicit time integration scheme, the length of the time steps must be less than a certain time to ensure numerical stability (i.e., to avoid any exponential magnification of approximation errors or numerical artifacts as the simulation proceeds).

**CFL-like conditioning**

At the beginning of each iteration, the maximal time step is calculated for the adopted first-order, forward-in-time scheme, and is chosen by calculating the minimum CFL factor for the simulated surface processes.

For the simple creep, the CFL factor takes the form:

$$\Delta t_{max} < min_{ij} \left( \frac{\lambda_{ij}}{2\kappa_d} \right)$$

The determination of the CFL limit for the nonlinear diffusion equation is quite tricky (see e.g. Pelletier, 2008; Press et al., 1992). We calculate an approximate limit as:

$$\Delta t_{max} \ll min_{ij} \left( \frac{\lambda_{ij} - |z_j - z_i|}{s^2} \right)$$

where the inequality is forced by multiplying the right-hand side by a factor such as 0.1 or less. The last expression must be evaluated at each iteration, since it depends on the surface slopes.

For the detachment-limited transport law the CFL upper limit for a given time step is given by:

$$\Delta t_{max} < min_{ij} \left( \frac{\lambda_{ij}}{\kappa_d q_{w,i}^{m} \left( \frac{|z_i - z_j|}{\lambda_{ij}} \right)^{n-1}} \right)$$

These CFL conditions serve as initial estimates of required time step but we apply an additional constraint detailed below to ensure numerical stability.

**Adjustment of soil transport fluxes**

A straightforward application of the time integration of the continuity equations would lead to incorrect or undesirable results. The main reason is that even with small, though reasonable, time steps, numerical instability may be locally initiated by too-rapid removal or accumulation of soil. To ensure that this undesirable effect never occurs during a simulation, we assume that the amount of soil leaving a cell cannot produce a lowering of surface elevation greater than the difference - multiplied by a factor $\alpha (< 1)$ - of elevation between the cell and its downslope adjacent cell (receiver $z_r$). And that the amount of deposition inside a given cell cannot be higher than the difference - multiplied by $\alpha$ - of elevation between the cell and the lowest upstream adjacent node $z_m$. Because diffusive processes tend to smooth out the roughness of the terrain surface, they are not favorable to the initiation of numerical instability. In fact, the addition of a maximum threshold of elevation lowering is only imposed by the action of transport by overland flow. The restrictions described above can be expressed as the following set of inequalities:

$$\frac{\Delta t}{\Omega} \left( \sum_j w_{ij}q_{s,ij}^{in} - \sum_j w_{ij}q_{s,ij}^{out} \right) \leq \alpha(z_r - z_i), \quad \sum_j w_{ij}q_{s,ij}^{out} > \sum_j w_{ij}q_{s,ij}^{in}$$

$$\leq \alpha(z_m - z_i), \quad \sum_j w_{ij}q_{s,ij}^{in} > \sum_j w_{ij}q_{s,ij}^{out}$$
where $q_{i,j}^{\text{out}}$ is the flux of soil leaving the Voronoi cell of the node $i$, $q_{i,j}^{\text{in}}$ the fluxes of soil coming from the adjacent cells.

To meet these restrictions for any amount of available soil and/or soil fluxes of any magnitude, a solution consists in choosing the minimal time step $\Delta t$ which satisfies this set of inequalities.

### Deforming mesh under 3D displacements

Due to tectonic advection, the density of surface nodes evolves over time, which leads to areas showing rarefaction or accumulation of nodes. In order for the interpolation schemes (based on local methods) to remain accurate and to avoid unnecessary computations, a local addition and deletion of nodes and the consequent remeshing of the triangulated surface are therefore required. The deletion, addition, and triangulation of nodes are performed at given time generally coincident with the coupling time with a geodynamics model (Underworld).

While the Badlands surface is required to be an irregular mesh to avoid potential directional bias of water flow (Braun & Sambridge, 1997), too irregular a mesh is not appropriate. The Triangle library is used to produce constrained Delaunay and to limit the creation of too irregular grids. For each newly advected TIN surface, Badlands algorithm ensures that the new nodal distribution

1. prevents the occurrence of low node density areas, and
2. enforces the required resolution for surface processes.

In addition, a node deletion/merging algorithm has been implemented to circumvent the problem of progressive increase of node number density (for example at mountain fronts) which could lead to significant slowdown of the surface process model.

---

**REFERENCES**


CHAPTER 2

Badlands install & quick start

Badlands source code is available from GitHub
badlands v1.0

Photography by Adriana Franco, National Geographic
Badlands: installation & dependencies

**KEY POINTS**

1. Instruction for installation.
2. Dependencies.
3. Compilation.

**DEPENDENCIES**

This section provides steps and links on obtaining and setting up Badlands model. The following instructions assume a beginner’s level of familiarity with the Unix/Linux operating environment, including:

- How to open a terminal window,
- How to navigate between directories using a terminal application,
- How to open, edit, save and close a file using an editor through a terminal application.

Badlands relies on a suite of software applications, which need to be installed prior to compiling the code. It is recommended that you install each program in the order listed.

**COMPILERS**

Badlands is currently supported for GNU compilers, however it works as well with Intel compilers. Both C and fortran compilers are required.

For Linux OS, they should already been installed with your distribution. GNU compilers for Mac OS can be downloaded from the HPC website.

To install them you could follow the explanations provided on the website or proceed as follow in a Terminal:

```
$ gunzip gcc-X.X-bin.tar.gz
$ sudo tar -xvf gcc-X.X-bin.tar -C /
```

It installs everything in /usr/local. Then edit your ~/.profile or ~/.bashrc and set up the path to the installed compilers.

```
export PATH=/usr/local/bin:$PATH
export LD_LIBRARY_PATH=/usr/local/lib:$LD_LIBRARY_PATH
```

Now update the configuration file in your Terminal by typing:
$ source ~/.profile or ~/.bashrc

MESSAGE PASSING INTERFACE

Badlands supports MPICH and OpenMPI. Below you will find guide to install MPICH. MPICH can be downloaded from the here. You will need to download the source code such as the mpich-3.1 (stable release).

Once downloaded, open a Terminal, go to the untar folder and do:

$ tar -xvf mpich3-xxx.tar
$ cd mpich3-xxx
$ CC=gcc FC=gfortran CXX=g++ ./configure --prefix=/usr/local/mpich
    --enable-fast=all --enable-shared --enable-sharedlibs=osx-gcc
$ make
$ sudo make install

Then edit your ~/.profile or ~/.bashrc and set up the path to the installed compilers.

export MPI_DIR=/usr/local/mpich
export PATH=$MPI_DIR/bin:$PATH
export LD_LIBRARY_PATH= $MPI_DIR/lib:$LD_LIBRARY_PATH

Now update the configuration file in your Terminal by typing:

$ source ~/.profile or ~/.bashrc

HDF5

HDF5 can be downloaded from the Hdf5 website. You will need to download the source code available for all platform such as the HDF5-1.8.13 release. Once downloaded, open a Terminal and navigate to he untar folder and do:

$ CC=mpicc FC=mpif90 ./configure --prefix=/usr/local/hdf5 --enable-
    parallel --enable-fortran

After the configuration script has finished, checked the summary of the log which is printed on your screen and ensure that both HDF5 for Fortran and C are present as well as the HDF5 compression facility (zlib). If zlib is missing you will have to install it first following documentation in the zlib website.

$ make
$ sudo make install

XML PARSER

FoX XmL can be downloaded from there. You will need to download the source code available as a tarball such as FoX=4.1.2.tar.gz. Once downloaded, open a Terminal and navigate to the untar folder and do:

$ cd FoX-xxx
$ FCFLAGS=’-O2 -fPIC’ FC=mpif90 ./configure --prefix=/usr/local/FoX
$ make
$ sudo make install

Once compiled, you will need to open the /usr/local/FoX/bin/FoX-config file using your preferred editing file:

$ sudo emacs /usr/local/FoX/bin/FoX-config

And change the comp_prefix (third line of the file) to the correct prefix in this case: /usr/local/FoX and delete the /objs at the end of the path.

METIS

The Metis library is available from there. Once downloaded, open a Terminal and navigate to the untar folder and do:

$ cdmetis-xxx
$ make config cc=mpicc prefix=/usr/local/metis
$ sudo make install
ZOLTAN

Zoltan needs to be downloaded from Sandia website. Once downloaded, open a Terminal and create a building folder:

$ mkdir zoltanbuild
$ cd zoltanbuild
$ ../Zoltan_v3.81/configure --enable-f90interface --enable-mpi --with-mpi-compilers --with-gnumake --prefix=/usr/local/zoltan
$ make everything
$ sudo make install

FIRE-UP BADLANDS

Badlands can be cloned from GitHub.
$ git clone git://github.com/badlands-model/Badlands.git

Please always refer to the above site for latest software updates. Prior to starting the compilation process, you will need to check the path to the libraries you’ve installed on your system as they might differ from one to the other. In Badlands, the configuration are set in the config folder. The folder contains several files associated to different environments e.g. Linux/Mac or Intel/Gnu. Depending on the ones you’ve installed, copy the corresponding file and edit it:

$ cp Makefile.cfg.osx Makefile.cfg.myconfig
$ emacs Makefile.cfg.myconfig

In the file, check that the paths are set correctly. If not sure look at your ~/.profile or ~/.bashrc. Make the necessary changes and save your file. Now open the Makefile.inc file and change the first line to match with your new configuration file name:

CFGFILE = $(TOP)/config/Makefile.cfg.myconfig

Then compile Badlands sources from the top directory:

$ make clobber
$ make dist
Badlands is using XML input file. External ASCII files are referenced inside the XML document and provide information on the basal node position, the changes in forcing parameters, e.g. uplift/subsidence, rainfall and sea level.

To start with a working XML input file it is recommended to use one of the examples provided in the next chapter.

**GENERAL PARAMETERs**

**GEOMETRY**

**<struct_geometry>**

The first structure which appears in the input file is related to the definition of the geometry: initial surface (regular grid) & triangular irregular network (TIN) parametrization.

**<regular_grid>**

refers to the name of the regular surface nodes file ASCII and its path. The path is from the main input file location. The file provides for each line the following information:

- X coordinates in meters (this axis has a West to East orientation),
- Y coordinates in meters (this axis has a South to West orientation), &
- Z coordinates in meters.

Nodes must be defined in increasing order from the South/West corner, first along the X axis.

**<delaunay_area>**

sets a maximum area constraint when generating the Delaunay triangulation. No triangle will be generated whose area is larger than that number. This area should be the at least equal to a regular grid cell area.

**<delaunay_angle>**
sets a minimum angle constraint when generating the Delaunay triangulation. If the minimum angle is 20.7° or smaller, the triangulation algorithm is theoretically guaranteed to terminate. In practice, the algorithm often succeeds for minimum angles up to 33°. It usually doesn't terminate for angles above 34°.

<refine_area>
defines the number of high resolution areas in the simulation.

Note: the number of refine_area in v1.0 is limited to 1.

<ra>
initial entry class to set the properties of a specific high resolution area. A high resolution area is defined as a squared box which extends have to be defined within the simulated area and be at least two regular grid cells inside the domain. If the refine_area is set to 0 this XML component is not required.

• <xmin> SW X corner of the considered high resolution area,
• <ymin> SW Y corner of the considered high resolution area,
• <xmax> NE X corner of the considered high resolution area,
• <ymax> NE Y corner of the considered high resolution area,
• <r_area> set high resolution area.

Setting up the boundary conditions on the simulation domain:

• <boundN> Northern boundary condition,
• <boundS> Southern boundary condition,
• <boundE> Eastern boundary condition,
• <boundW> Western boundary condition.

Possible values for these fields are 0,1,2. Where 0 defines a fixed boundary, 1 a wall and 2 a slope boundary. By default the value is set to 2 for all borders.

• <outlet> defines an outlet on one of the corner of simulation area (allowed values are: 1 for SW corner, 2 for SE corner, 3 for NW corner and 4 for NE corner). When defining an outlet the previous boundary conditions will be set to 1 (wall boundary).

TIME

<struct_time>
The second structure defines simulation and processes time steps.

• <time_start> starting time of the simulation,
• <time_end> ending time of the simulation,
• <display_interval> time interval to create output.

In addition and if we are restarting a previous simulation:

• <restart_folder> name of the previous simulation folder,
• <restart_fileID> time interval of the restarted simulation,
• <restart_petNb> number of processors used in previous run,
• <time_step> set minimum time step for the run.

FORCING PARAMETERS

SEA-LEVEL FLUCTUATIONS

<struct_ocean>
Optional structure which define ocean fluctuations.

<ocean_file> name of the file containing the sea level evolution through time. This ASCII file provides for each line the following information:
• time in years,
• sea-level elevation for the considered time in meters.

In addition the defined fluctuation times should be set in increasing order starting from the oldest time.

**UPLIFT/SUBSIDENCE EVOLUTION**

**<struct_geodyn>**

Optional structure which define tectonic displacements.

• `<fields_3D>` should be set to 1 if horizontal displacements are used (default value is set to 0),
• `<merge_dist>` the minimal distance (in meters) between two TIN nodes under which point merging algorithm is performed (required if `<fields_3D>` is equal to 1),
• `<interval_nb>` total number of displacement intervals that will be simulated. The displacement intervals need to be declared by increasing time and should not overlap with each others,
• `<disp>` definition of an individual displacement field,
• `<disp_start>` time in years at which the considered vertical displacement will start,
• `<disp_end>` Time in years at which the considered vertical displacement will end,
• `<disp_file>` name of the displacement file and its associated path. The path is from the main input file location.

The displacement ASCII file provides at each line the cumulative vertical displacement for each considered node in meters.

**Attention:** the file defines the cumulative displacement during the duration defined above and not a displacement rate.

**Note:** the nodes are read in increasing order based on the regular grid discretization starting from the South-West corner and going first along the X axis (West to East).

---

**CLIMATE EVOLUTION**

**<struct_rainfall>**

To simulate overland flows it is essential to defines rainfall parameters. It is possible to specify a series of maps with rainfall values defined for each nodes.

• `<rain_nb>` defines the number of rainfall series imported for the considered simulation.
• `<rain>` collection of rainfall parameters defining each series.

**Note:** the number of classes needs to be equal to the number given in the `<rain_nb>` element.

• `<rain_start>` start time of the considered rainfall grid interval in years,
• `<rain_end>` end time of the considered rainfall grid interval in years,
• `<rain_file>` name of the rainfall file and its associated path. The path is from the main input file location.

**OR**

• `<rain_value>` in case the rainfall is uniform over the simulation it is possible instead of using an external ASCII file `<rain_file>` to define the value within the XML file directly (in meters/year).

This ASCII file provides for at each line the averaged precipitation in meters/year.

**Note:** the nodes need to be defined in increasing order based on the regular grid discretization starting from the South-West corner and going first along the X axis (West to East)

**Note:** in case a specific node does not have any precipitation it is required to set it in the file with a 0.0 value.
OVERLAND FLOW PARAMETERS

DETACHMENT LIMITED

<struct_detachment>

Model equation:

\[ q_r = -\kappa_r (q_w)^m (\nabla z)^n \]

For detachment limited transport (stream power law), the following parameters are required.

<spl_m> corresponds to the coefficient \( m \) in the equation.

<spl_n> corresponds to the coefficient \( n \) in the equation.

<Cerodibility> corresponds to the coefficient \( \kappa_r \) in the equation.

In case where you are interested in purely erosive system the optional element below could be set to increase the computational time:

<pureErosion> to take into account erosion only, possible values are 0 or 1.

Hillslope parameters

SIMPLE CREEP

<struct_linear_diff>

Model equation:

\[ q_d = -\kappa_d \nabla z \]

For simple creep, the following parameters are required:

<coef_linear_aerial> \( \kappa_d \) parameter in aerial environment.

<coef_linear_marine> \( \kappa_d \) parameter in marine environment.

SEDIMENT LOADING

FLEXURAL ISOSTASY & COMPACTION

<struct_flex>

<flex_dt> flexural isostasy & compaction computation time step (a multiple of the display interval time),

<flex_dx> spatial discretization of the flexural grid in meters (values are around several kilometers),

<sed_dens> sediment density (\( kg/m^3 \)),

<mantle_dens> mantle density (\( kg/m^3 \)),

<flex_thick> elastic thickness (in meters),

<flex_order> choice is 2 (13-point second-order grid approximation) or 4 (four-order multi-grid solutions to the biharmonic equation),

<nb_Pfields> number of lithostatic pressure in the porosity table.choice is 2 (13-point second-order grid approximation) or 4 (four-order multi-grid solutions to the biharmonic equation),

<pressure_val> lithostatic pressures values (in GPa),

<porosity_val> associated porosity values for considered lithostatic pressures.

VISUALIZATION

OUTPUT

<output_directory>

Name of the output folder where both simulation results and run files will be stored. The directory is divided in two sub-directories:
• one called **outputs** which record simulation output,
• one called **runfiles** which record simulation informations

**Note:** the outputs sub-directory contains time-series files that can be visualized in **Paraview** or **Visit**.

**Note:** the runfiles sub-directory contains the checkpointing files required to restart a simulation.

**Note:** to preserve previous runs, you will have to change the output directory name otherwise it will be automatically rewritten and previous outputs will be lost.
All tests provided in the following section are available from GitHub
Badlands-doc v1.0
Section 1

Planar surface with single outlet

Key Points

1. Experimental settings.
2. River profile.

Planar surface with single outlet

Initial conditions

Above animation shows an example of surface simulation starting from a horizontal topographic surface of size $10 \times 10$ km$^2$, sampled randomly on about 10,000 points, that evolves through a time span of 500 kyr. The uplift rate is set to 1 mm/yr everywhere except on the outlet point where a subsidence of 0.2 mm/yr is imposed. The rainfall is set to 1 m/yr over the entire domain. All surface boundary points are closed, except for a single open outlet at (0,0) coordinates. This surface is allowed to evolve by simulating linear diffusion (simple creep) and detachment limited overland flow processes. Diffusion is implemented with a uniform diffusivity constant $\kappa_d = 0.1$ $m^2$/yr. Erosion due to channeling is simulated with the following parameters: $\kappa_r = 10^{-5} m^{1-2m}/yr$; $m = 0.5$;
\[ n = 1. \] Note that, in this simulation, all the model parameters are constant and uniform. The only spatially random element is given here by the position of the sampled points. A developed drainage network is already visible after 50 kyr evolution, as a result of competition between diffusive hillslope transport and channel incision processes. After 250 kyr, the surface has evolved and forms a fully developed, dendritic drainage pattern with maximum stream order of 7 over the last portion of the catchment just before reaching the outlet. Over the last 200 kyr the drainage pattern is stabilized.

**RIVER PROFILE EVOLUTION**

To underline more quantitatively the consistency of the simulation, it is possible to represent a series of profiles of one particular river (the longest visible blue line roughly on the bottom diagonal of the squared domain) extracted from the TIN surface. We observe the progression of erosion in the upslope direction as time passes (top figure), indicated...
symbolically by the grey arrow. By a simulated time of 200 kyr, the entire river course has settled and the river profile shows an upward concavity (bottom figure).

**SENSITIVITY ANALYSIS**

Model sensitivity to both diffusion and detachment limited coefficient is explored for 3 different values of $\kappa_d$ and $\kappa_r$. The results presented in the next figure shows how the landscape has evolved for a considered period of 500 kyr. We observe that increase in hillslope processes magnitude (increase in $\kappa_d$) leads to a “smoothing” of surface topography but does not produce changes in the final morphology of the river network. The effect of $\kappa_r$ is tested by varying it between $8 \times 10^{-6}$ – $5 \times 10^{-5}$. Because it is a constant of proportionality, varying $\kappa_r$ influences the rate of landscape evolution; if $\kappa_r$ is increased by more than 10 for the considered set of model parameter, the landscape erodes nearly instantaneously.

**REFERENCES**


**Section 2**

**Delta evolution & sea-level fluctuations**

---

**KEY POINTS**

1. Experimental settings.
2. Hillslope & overland flow models.
3. Depositional/erosional patterns.
4. Stratigraphic evolution.

---

**Sediment transfer, delta formation and sea-level variation**

---

**Experimental settings**

The initial surface defines a mount which is a half ellipsoid of 2000 m height and about 8 km large ([youtube video](https://www.youtube.com/watch?v=dQw4w9WgXcQ)). The sea-level is initially set at the base of this mount at a elevation of 0 m. An uniform precipitation rate of 1 m/yr is applied on the all area and we test the evolution of the surface due to both hillslopes and overland flows. Two hillslope coefficient are defined for both the aerial and marine area. The simulation runs for 500,000 years and we vary the sea-level through time (values are defined in table next page). From its initial setting, we first perform a sea-level drop of 100 m after 200,000 years than we come back to the initial sea level at 300,000 years. We then perform a second cycle, this time by imposing a sea-level rise of 100 m for around 100,000 years,

---

Mountainous landscape evolution with sedimentation front evolving over time subject to sea-level fluctuations.
then finally, we reset the sea-level to 0 m until the end of the simulation.

In addition, the grid is divided in two resolutions by defining a bounding box within the input file. A high-resolution area (10,000 m²) is chosen around the mount, whereas the marine environment has a lower resolution of 40,000 m².

**Depositional/Erosional Patterns**

During the first phase of the simulation the mount is eroded and sediment are transported in the marine environment where we see a sedimentation front prograding towards the marine environment (two top views in the next figure).

As the sea-level falls, it creates a condition of forced regression, in which the coast is forced to build seaward. During the falling stage, rivers begin to incise on what was formerly a marine shelf, forming incised valleys. These incised valleys tend to widen and grow landwards. From 200,000 – 280,000 years, the low-stand system marks the greatest extent of subaerial exposure and erosion.
As the rate of relative sea-level rise increases, it leads to retrogradational parasequence stacking also referred as transgressive systems. During this stage the previously incised valleys are filled as shown in the previous figure at 300,000 years.

Between 320,000 – 380,000 yrs, the sedimentary system turns from retrogradational stacking in the transgressive system to progradational stacking in the highstand system. During this stage, the sediment supply to the shelf favors the development of progradational sequences.

During the last 100,000 years, the system returns to its original sea-level which is marked by incision of formerly marine shelf deposits and development of a new sedimentation front.

REFERENCES


Section 3

Climatic forcing on drainage evolution

Key Points

1. Experimental settings.
2. Natural analogues.
3. Temporal evolution of catchments.

Role of climate change in drainage network reorganization

The continuous feedbacks among tectonics, surface processes, and climate change are reflected in the distribution of catchments on active mountain fronts. Previous studies have shown a certain regularity of valley spacing on several mountain ranges worldwide, but what is at the origin of such geomorphological feature of landscapes is currently not well known.

This example illustrate long-term landscape evolution of an active mountain range. It shows how the constant valley spacing, achieved at steady state on both sides of the range, is progressively restored after simulating a migration of the main drainage divide caused by a precipitation gradient applied across the mountain belt.

From an initial steady state, drainage network reorganization under an imposed precipitation gradient.
EXPERIMENTAL SETTINGS

We simulate landscapes of various sizes, with a fixed TIN mean sampling distance of 100 m; the initial surface is completely flat at zero elevation. Sediment can flow across all four landscape boundaries. We apply a spatially uniform and constant uplift rate of 1 mm/yr over the whole domain.

After the initial topography has reached a steady state under a uniform, constant precipitation rate of 1 m/yr, a precipitation gradient is introduced, in a direction perpendicular to the belt axis, to simulate "orographic" precipitation; the maximum precipitation rate is set to 3 m/yr along the upper side, and it linearly decreases towards the opposite side reaching a minimum of 0.1 m/yr.

TEMPORAL EVOLUTION OF RIVER CATCHMENTS

The landscape reached the first steady-state topography under uniform precipitation after 15 Myr. The pattern of catchments, illustrated in the top right figure, consists of an alternation of larger, and smaller catchments etched on the triangular facets. The catchments on the North and South side of the mean drainage divide exhibit the same elongated characteristics.

After the precipitation gradient is applied, the main divide of the belt is pushed from the wet side (northern part) towards the dry side (southern part), where larger catchments shrink and, simultaneously, the smaller catchments on triangular facets grow. Furthermore, the drainage network and maximum stream order is consequently reduced in the larger watersheds.

A new steady state topography is reached under changed climatic conditions. The numerical results show that change in the precipitation pattern exerts a strong control on drainage network development and reorganization on both sides of the simulated mountain belt. In particular, the leeward side shows a more interesting evolution, where the interaction amongst all geomorphic processes leads to a significant catchment reorganization.
The final result of this complex mechanism of watershed reorganization is a new steady state topography.

Furthermore, the model of watersheds development proposed here, can explain some real landscapes in which we found analogies with the different evolutionary stages of the simulated belts.

REFERENCES


High-resolution depression filling model

Section 4

Pit crater geomorphological evolution

Central pit craters are a crater type that contain an approximately circular depressions in their floor or central peak. These craters have been found on Earth, Mars, Ganymede, and Callisto. Here we use their typical shape to perform a high resolution simulation using Badlands depression filling method.

Experimental settings

In this example, we impose over 100,000 years a constant precipitation rate of 1 m/yr over an initial 2 \times 2 \text{ km} crater if 250 m depth. The resolution of the TIN is lower than 10 m.

Pit evolution

Key Points

1. Experimental settings.
2. Pit evolution.
3. Crater filling.
Simulated fluvial stream power law implies the infill of the crater bottom with alluvial fan deposits, whereas the crater rim gradually backwastes while maintaining a generally steep slope.

Both the backwasting and the downcutting of the crater rim occur, so that in later stages of erosion the crater may become essentially rimless.

**CRATER INFILLING**

Degradation of craters by fluvial erosion and deposition in this simulation results in rapid initial infilling but a slowing rate through time likely due to the decreasing height of interior crater walls and the increasing area of deposition as the crater enlarges due to crater wall erosion. The result is in agreement with several other studies on this specific type of morphology (Forsberg-Taylor et al., 2004; Howard, 2007).

Progressive filling of the pit crater through time (top 1,000 years, middle 50,000 years and bottom 100,000 years).
REFERENCES


Section 5

3D tectonic coupling and drainage evolution

Key Points

1. Pull-apart basins.
2. Displacement fields parameters.

Pull-apart basins

Interlinks between deformation and sedimentation have long been recognized as an important factor in the evolution of continental rifts and basins development. However, determining the relative impact of tectonic and climatic forcing on the dynamics of these systems remains a major challenge.

Experimental Settings

In this example, we impose over 2,000,000 years a 3D displacement field produced with Underworld over an initial flat surface. We consider both hillslope (creep) and overland flow processes (detachment limited) induced by an uniform precipitation rate of 1 m/yr.

The 3D displacement field obtained from Underworld gives us an average rate which is updated every 5,000 years. The model is initially a 256 km square box at a resolution of 1 km. The video for this model is avail-
able on here. We test with Badlands the effects of high horizontal displacement rates on landscape geomorphological evolution. It is important to note that just the first displacement files for running the first thousands years are provided here.

**DISPLACEMENT FIELDS**

Pull-apart basins are depressions that are bounded by sideways-stepping, strike-slip faults parallel to their length. They form where the sense of stepping or bends along the faults have the same sense as fault motion.
The internal structure of these basins is highly variable both in space and time owing to complex stress fields and heterogeneous crustal rheology around the termination of the delimiting faults. This complexity has led to several unresolved problems regarding the kinematics and dynamics of pull-apart basins. Using the coupling between Badlands and Underworld it is now possible to test the time-dependent deformation patterns within pull-apart basins, and the relation of these basins with the adjacent deformed structural domains.

**Basins Evolution**

Cumulative erosional/depositional evolution for the different erodibility coefficients.
Section 6

Landscape evolution at continental scale

Key Points
1. Deep Earth & surface evolution.
2. Basin filling.
3. Deposit preservation.

Deep earth and surface evolution

Flow within the deep Earth shapes its surface over tens of million years and thousands of kilometers, with consequences for the evolution of continental-scale topography, drainage patterns and sediment deposition.

In this example, we investigate the interaction of solid-Earth and surface processes by coupling global tectonic reconstructions and computations of mantle flow (from CITCOMS) with models of erosion, sediment transport and deposition.
We consider the evolution of eastern Australia since the Late Cretaceous, which is characterized by two phases of modelled uplift (~200 m) between 120 Ma and 80 Ma, due to the cessation of long-lived subduction at around 100 Ma, and the fast motion of Australia towards the Pacific superswell between 60 Ma and 20 Ma (~200 m). These results are consistent with the uplift history deduced from the analysis of river profiles (Czarnota et al., 2014). We couple these dynamic topography predictions with a continental-scale surface process model that we use to compute 100 Myr of landscape evolution from a history of varying climate, sea level and mantle flow.

**Basin filling & deposit preservation**

The model predicts the time dependence of erosion and drainage patterns that we compare to the sedimentary record in key basins, paleodrainage from paleogeography and denudation rates. In particular, over 5 km of post-rift sediments were deposited in the Ceduna Basin (the southern margin of Australia) in two main phases starting at ~80 Ma and ~55 Ma. In order to match the Ceduna Basin sedimentation history, we vary the coefficient of erosion by an order of magnitude and the amplitude (between 200 m and 1,000 m) of an uplift event that affected eastern Australia at 95 Ma and that is not fully captured by the mantle flow model. Recorded basin stratigraphy shows the difference in thickness of preserved sediments for the different cases.

**References**
