'MARSSIM' LANDSCAPE EVOLUTION MODEL *release 3.0*

Alan D. Howard

Department of Environmental Sciences
University of Virginia
291 McCormick Rd, P.O. Box 400123
Charlottesville, VA 22904-4123
205 Clark Hall
(434) 924-0563
(434) 982-2137 FAX
alanh@virginia.edu
Webpage: http://erode.evsc.virginia.edu

This is the documentation for a landform evolution model variously termed DELIM and MARSSIM the publications listed below. The model is built upon the landform evolution model first described by Howard (1994). The core model primarily focuses on landform evolution at relatively long temporal scales (relative to the timescale for noticeable landform change) through fluvial and mass wasting processes. The program is designed be computationally efficient such that individual runs can be done on a modern microcomputer in no more than a few tens of hours. The more recent additions to the model have focused on processes relevant to planetary landscapes, including lava flows, groundwater seepage and sapping, impact cratering, surface-normal accretion and ablation, and volatile redistribution by radiation-induced sublimation and recondensation. Individual process formulations vary from completely heuristic to modestly mechanistic. Important limitations for some potential applications are the assumption of a single representative bed material grain size in the fluvial system and no tracking of internal stratigraphy of sedimentary deposits (except for total thickness and surface morphology). Some stratigraphic information can be gleaned by frequent reporting of elevation changes through time on sedimentary deposits. The process formulation is described in the following publications:

1994 Howard, A.D., A detachment-limited model of drainage basin evolution, *Water Resources Research*, Vol. 30, No. 7, p. 2261-2285.

1997 Howard, A. D., Badland morphology and evolution: Interpretation using a simulation model. *Earth Surface Processes and Landforms*, v. 22, 211-227.

1999 Howard, A.D., Simulation of Gully Erosion and Bistable Landforms, book chapter, in *Incised River Channels*, edited by S. Darby and A. Simon, John Wiley & Sons, p. 277-300 & plates.

1999 Howard, A. D., Simulation of lava flow inundation on Martian cratered terrain, Lunar and Planetary Science Conference XXX, Abstract 1112. http://www.lpi.usra.edu/publications/meetingpubs.shtml

2004 Forsberg-Taylor, N.K., Howard, A.D. and Craddock, R.A., Crater degradation in the Martian Highlands: morphometric analysis of the Sinus Sabaeus region and simulation modeling suggest fluvial processes, *Journal of Geophysical Research, Planets.*109, E05002, doi:10.1029/2004JE002242.

2004 Fagherazzi, S., Howard, A. D., and Wiberg, P. L., Modeling fluvial erosion and deposition on continental shelves during sea level cycles. *Journal of Geophysical Research,* v. 109, doi:10.1029/2003JF000091.

2004 Howard, A. D., Simple non-fluvial models of planetary surface modification, with application to Mars, Lunar and Planetary Science Conference XXXV, Abstract 1054. http://www.lpi.usra.edu/publications/meetingpubs.shtml

2007 Howard, A. D., Simulating the development of martian highland landscapes through the interaction of impact cratering, fluvial erosion, and variable hydrologic forcing, *Geomorphology*, v. 91, p. 332-363.

2008 Howard, A. D., and Moore, J. M., Sublimation-driven erosion on Callisto: A landform simulation model test. *Geophysical Research Letters,* v. 35, L03203, doi:10.1029/2007GL032618.

2008 Luo, W., and Howard, A. D., Computer simulation of the role of groundwater seepage in forming Martian valley networks, *Journal of Geophysical Research, Planets,* vol. 113, E05002, doi:10.1029/2007JE002981.

2009 Barnhart, C. J., Howard, A. D., Moore, J. M., Long-term precipitation and late-stage valley network formation: Landform simulations of Parana Basin, Mars, *Journal of Geophysical Research Planets,* vol. 114, E01003, doi:10.1029/2006JE003122.

In addition, the following document, included with the distribution, summarizes most of the model components and also includes two unpublished sections on dimensionless scaling and two schemes for measuring the relative amounts of surface modification by various processes (fluvial transport, mass wasting, eolian deposition, and impact cratering).

2002 Howard, A.D., Simulation models for landform evolution on early Mars: Cratering, lava emplacement, eolian modification, weathering, mass wasting, and fluvial processes. ***[Marsmodel.pdf, included in distribution]***

The program is coded in standard Fortran 90 in free-form format. It has been successfully compiled with a number of compilers, including *Intel Fortran* (windows and linux), *PGI Fortran* (linux), and the public-domain *gfortran*. The program runs in a command-line environment (Command Prompt window in Windows, Terminal window in Linux). The following programs and files are included with the distribution (subroutines and functions in the various files are indicated in bold):

Marssim_program.doc - this document

Source files These are distributed in ***source_files.zip***. (files include subroutines that are generally grouped by process or function) **:**

program_global_variables.f90	Definition of global variables. These are grouped into several modules related to particular process suites. Most matrices are defined as allocatable arrays whose size is determined at runtime by the main input file (*marssim.prm*). Only matrices used for the selected processes are actually allocated.
main_program.f90	This is the main program (**program marrsim**), which reads in parameters, opens output files, initializes the program, conducts the iterations, and closes the program. In addition, this file contains the subroutine **read_input_parameters** which performs as suggested, as well as allocating matrices, reading in the initial elevations plus any other necessary input files.
initialize_variables.f90	The subroutine **initialize_variables** initializes a number of variables and small matrices primarily used for fluvial and mass wasting erosion and file output. In addition it sets up lookup tables for rates of mass wasting and rock weathering as functions of local gradient. If a time-varying ocean level is specified, it reads in the requisite file of levels and times. **normal_random_deviate, lognormal_random_deviate, lognormal_random_deviate1, exponential_distribution,** and **rrand** generate random numbers following the indicated probability distributions (**rrand** provides uniformly distributed numbers between zero and one).
erosion_iterations.f90	These subroutines are concerned with fluvial and mass wasting processes. **setup_fluvial_slope_erosion** initializes several initial variables and matrices. **do_fluvial_and_slope** is called each iteration when fluvial and/or slope erosion is modeled to invoke the processes. It also calls various routines at set intervals to report on process rates and landform states and to write data files. **finalize_fluvial_slope_erosion** closes down the fluvial and slope modeling parts of the program, summarizes the system state, and writes out and closes output files.
boundary_conditions.f90	This includes a number of subroutines that are called at

	intervals or during each iteration that control changes of boundary conditions, including process rates and programmed"events". **boundary_conditions** is called each iteration to control base level as a function of time (for simulations with an eroding lower boundary) and performs any rock deformation. **determine_erodibility** is called if rock resistance varies through 3-D space. It calls **read_erodibility** to read rock resistance from an input file, '***resist.in***' **determine_erosion_rate** is called if process parameters change abruptly at set time intervals. **make_event** is called at set times if some abrupt change in the system state is desired. For example, changing certain process parameters or leveling part of the landscape by simulated wave erosion (as included in the source file). This routine would typically be tailored to specific landform evolution scenarios. **find_ocean_elevation** determines the relative level of the ocean if it is time-varying. **change_flow_direction** is used if flow across alluvial surfaces or deltas has a memory requiring a probabilistic event to change (e.g., avulsions of birdfoot deltas)
fluvial_slope_erosion.f90	This contains the main subroutines for fluvial and mass wasting erosion. **fluvial_detachment** determines the local fluvial bedrock and regolith erosion rates as a function of shear stress and/or abrasion. It calls **local_values** to determine the spatial variability of controlling variables. **do_the erosion** is the master subroutine coupling fluvial erosion, sediment transport and deposition, mass wasting, weathering, and seepage weathering for each iteration, calling a number of the other subroutines. It controls the overall time increment if fluvial and slope erosion is modeled. It also does elevation changes by fluvial and slope processes during each iteration and controls changes in state (whether locations are alluvial or bedrock, in normal or accelerated state of erosion, if the erosion is in a surface crust). **write_debug** and **print_around** are called for debugging purposes.
mass_wasting.f90	**do_mass_wasting** is called by **do_the_erosion** to determine the rate of mass wasting on regolith-covered slopes and to route weathered debris on rock slopes to the nearest downslope non-bedrock location.

	rapid_creep is a lookup function for the rate of mass wasting as a function of local gradient. **rock_mass_wasting** is a lookup function for the rate of mass wasting of bedrock slopes as a function of local gradient
sediment_routing.f90	Contains several routines associated with transport and deposition of alluvium in channels, fans, and deltas. **find_downstream_location** is used to determine the next location downstream for routing of sediment. **write_debug_data** does as it says. **sediment_transport_flux** determines the rate of sediment transport as a function of gradient, flow properties, and sediment characteristics. **equilibrium_sediment_gradient** determines the steady-state alluvial gradient corresponding to specified values of bedload flux and local discharge. **sediment_flux_divergence** is called by **do_the_erosion** and determines the rate of change in alluvial surface elevation as a function of the spatial divergence of sediment transport. **route_sediment** is called by **do_the_erosion** and determines changes in alluvial surface elevation by routing sediment at an equilibrium gradient through channels, across fans, or on deltas. The procedures and assumptions are presented in Howard (1994). This subroutine is called multiple times during an iteration for each location where a bedrock channel debouches on to an alluvial surface.. **smoothsed** is optionally called each iteration to cosmetically smooth out the sediment surface calculated by **route_sediment.** **check_if_change_flow_direction** is optionally called *within* individual iterations to change the flow directions on alluvial surfaces resulting from the **route_sediment** subroutine. **print_sediment_diagnostics** does as indicated.
weathering.f90	This subroutine weathers rocks. **calculate_divergence** is called by **do_weathering** to determine local slope divergence. **do_weathering** weathers the bedrock surface both for locations with exposed bedrock (*is_rock_surface* true) and for regolith-covered locations. If it is a regolith-covered surface it increases the thickness of the regolith (*regolith* takes a positive value indicating regolith thickness in this case). If it is a bedrock surface *regolith* takes a negative value indicating the rate of bedrock

	weathering. For bedrock surfaces weathering rates can be determined by slope steepness and local divergence (e.g., exfoliation) as well as solar radiation (on planetary surfaces) and groundwater seepage rates. If it is a regolith-covered surface the rate of weathering of the bedrock surface can either be a negative exponential function of regolith thickness or a humped function of regolith thickness (chemical weathering). **find_depression** just determines if the location is a local elevation minimum.
gradient_and_flow_directions.f90	***gradient_and_flow_direction*** determines local topographic gradients (*d8_gradient*) and downstream flow directions (*flow_direction*). The latter is negative for local topographic minima and is unity for fixed boundary locations.
default_flow_routing.f90	This routes runoff across the landscape under either hyperarid conditions (where runoff disappears instantly in depressions) or fully wet conditions (no evaporation or infiltration – *complete_runoff* is true). **discharge_from_cell** determines the flux of water from each simulation cell, which can be a function of local slope divergence, seepage from groundwater, or state of accelerated erosion. Usually it is just proportional to cell area. **is_it_submerged** determines whether individual cells are under water. **drainage_basin_area_flow** does the flow routing, determining drainage areas and flow amounts within the drainage network, as well as routing sediment through the bedrock channel portions of the network (where transport rates are assumed to be very rapid compared to the simulation time step). Lake elevations and outlet locations are determined (if *complete_runoff* is true) and channel width is determined as a function of local discharge. **basin_report** is used for debugging.
lake_flow_routing.f90	This is a more sophisticated (albeit computationally slower) flow routing routine that allows for evaporation in lakes and lake overflow that is conditional on the balance of runoff and evaporation (see Howard, 2007). **drainage_basin_lake_flow** functions like **drainage_basin_area_flow** except for the conditional lake overflow. **drawline** and **drawsline** are used to connect inflows to lakes to their exit in order to create images of the flow network. **pelagic_deposit** deposits suspended sediment in lakes. It

	also diffuses the deposited sediment to create a smooth basin floor. **check_flow_path** is a debugging routine
groundwater_flow.f90	This routes groundwater as DuPuit (horizontal unconfined) flow as a function of assigned infiltration rate, aquifer depth, and permeability. Steady flow is assumed. The rate of seepage back to the surface is determined. This seepage can optionally contribute to (or dominate) surface flows and increase rock weathering rates (see Luo and Howard, 2008) **exponential_hydr_cond_grndwtr** calculates the groundwater flow under the assumption that permeability decreases exponentially with depth beneath the surface. **constant_hydr_cond_grndwtr** calculates the groundwater flow assuming a constant thickness, constant permeability aquifer.
impact_cratering.f90	Geometrically simulates impact cratering using random spatio-temporal impacts following a given production function and crater geometry. See Forsberg_Taylor et al. (2004) and Howard (2007). **do_impact_cratering** is the master routine called for each simulated impact, calling the other routines. **get_crater_size** determines the crater size as a function of the production function. **find_modification_range** determines how far out from the impact site that ejecta deposition must be modeled **find_impact_site** determines where in X-Y space the impact occurs **create_crater** does the heavy shoveling, transporting, and ejecta spreading. **find_reference_elevation** determines the average ground location into which the crater is excavated.
lava_flows.f90	This simulates episodic lava flows from multiple specified vents. There is some documentation in the file. **do_lava_flows** is the main subroutine. **find_active_lava_sites** determines where on lava flows new lava extension can occur **find_lava_start_place** determines where a new flow starting from a vent goes **find_next_lava_site** determines where the next cell to be occupied by the flow is.
eolian_erosion_deposition.f90	This heuristically models eolian landform mantling. See Forsberg-Taylor et al. (2004). **do_eolian_change** is the main subroutine. **exposure** determines the degree to which a given location is “exposed” or “sheltered” from the wind.

	total_exposure is an alternate method for determining "exposure"
surface_erosion_deposition.f90	This routine includes heuristic modeling of surface-normal erosion or deposition on planetary surfaces. **do_accretion_ablation** models surface-normal or vertical uniform addition or removal of sediment from a surface. **do_exposure_dependent**_creep models mass wasting where the creep diffusivity depends upon the surface exposure (as in the eolian modeling) as well as gradient **find_top_exposure_index** is another method for determining "exposure" **setup_distance_weighting** sets up a matrix of weights that decrease as a negative exponential from the given location **rad_erode** heuristically models solar-induced sublimation from planetary surfaces by reflected-reemitted IR radiation. See Howard and Moore (2008). **find_rad_change** determines the sublimation rate at a given location **deposit_ice** heuristically models ice accretion on surfaces not exposed to reflected solar radiation.
summary_statistics.f90	Calculates a variety of morphometric parameters on simulated landscapes. Designed primarily for fluvially-eroded landscapes with near-steady-state topography. **moments** sums the first four moments of passed values reset_moments zeros out the moment vector **calculate_moments** calculates the first four statistical moments (mean, variance, skewness, kurtosis) of values passed in an X-Y array **print_moments** prints out the calculated moments **find_topographic_extrema** identifies the total number of summits, sinks, and saddles on a topographic surface **calculate_topo_divergence** prints the moments of planform and profile curvature, gradient divergence, and ln(area/gradient). **print_morphometry** summarizes various statistical characteristics of a simulated landscape **print_simulation_information** episodically reports on the statistical characteristics of a simulation, including rates of landform modification **print_variable_summaries** calculates correlations and anova relationships between simulation state and rate variables as well as percentile distribution values for a number of state variables **correlate** finds the correlation between two matrices **print_bedrock_statistics** finds correlations between rate

	processes and topographic properties. **print_rate_statistics** summarize the rates of landform modification
channel_properties.f90	**channel_properties** summarizes representative stream profiles within the simulation domain
stream_network_properties.f90	**summarize_channels** calculates a number of classic measures of channel network geometry. For specifics see the source file. **percentiles** calculates the 16^{th}, 25^{th}, 50^{th}, 75^{th} and 84^{th} percentile values of a vector
determine_stream_network.f90	**find_stream_network** calculates and prints information about stream network geometry, working together with **summarize_channels**
write_debug_information.f90	**print_debugging_data** does as it says, and utilizes a number of associated routines: **print_integer_matrix_data** **print_basin_information** **print_logical_matrix_data** **print_real_matrix_data** **summarize_matrix_data** **summarize_regolith_data** **summarize_logical_matrix**
read_and_write_data_files.f90 ***alternate_read_and _write_data_files.f90***	As the name suggests, most data input and output is done through this subroutine. Most read or write values of a single matrix. The default version of this file (***read_and_…f90***) uses F90 standard non-advancing output to write binary image files, whereas (***alternate_read_and_...f90***) uses the non-standard $ output control. The difference is that the default version appends a CRLF pair to the end of each image file, whereas the $ output does not. Includes the following, mostly self-explanatory subroutines: **read_alluvial_locations** **read_bistable_locations** **read_bedrock_locations** **read_sediment_base** **read_regolith_thickness** **read_deformation** **write_debug_info** **write_gradient_info** (just a shell) **write_alluvial_locations** **write_bedrock_locations** **write_lake_info** **write_erosion_depth_index** **write_accelerated_erosion_state**

	write_sediment_base **write_regolith_thickness** **write_deformation** **write_rock_resistance** **write_first_data_matrix** **write_second_data_matrix** **write_data_sample** **write_report** **write_image** **write_shaded_relief_image** **write_groundwater_flow** **write_groundwater_elevation** **find_groundwater_flux** **write_routed_discharge** **write_submerged_locations** **grad_disch_write** **write_color_shaded_relief_image** **write_lava_info** **write_lava_ages**

Input files:

inelev.dat -initial elevations for the simulation It, and most other data files read or written by the program, is read by the following pseudocode:

```
read (indata,*) mx,my !the x and y dimensions of the simulation domain
do i=1,mx
do j=1,my
 read(indata,*) elevation(i,j) !elevation is the surface elevation at location i,j
enddo
enddo
```

marssim.prm -all the master parameters for the simulations

erode.stop -This file is read every 10 iterations, reading a single integer. If it is 1 the program stops and writes out files, otherwise if 0 the program continues. To stop the program in mid-simulation without discarding results, copy ***erode.stop.yes*** to ***erode.stop***. But be sure to copy ***erode.stop.no*** to ***erode.stop*** before the next simulation.

clearerode.bat -This is an MSDOS batch file that deletes most files left over from previous runs. Since most of the output is appended to any existing files, run this before you start a new run.

runname.bat -This is a MSDOS batch file that renames output files from the program to a single prefix with a variety of suffixes and moves most of them to a ".zip" file. Requires pkzip.exe to be in executable path. The batch file requires a 6 digit alphanumeric code in the command line that uniquely identifies the run, e.g. ***runname abcdef,*** followed for the next run by, say, ***runname abcdeg***. It does not incorporate the image (*.raw) files. This batch file requires the shareware program ***pkzip.exe*** be in the search path. Presumably any command-line zip utility could be substituted.

For simple simulations, ***inelev.dat*** and ***marssim.prm*** are all the files that are needed, but for some simulations, depending on switches set in ***marssim.prm***, the following may be needed:

resist.in -a 3-D matrix of rock resistance
continue.dat -a file containing the state of the simulation at the close of the simulation (the implementation is not up-to-date and should not be used in the present release)
inreg.dat -initial conditions for regolith thickness
inrates.dat -reads time-varying simulation parameters
inriver.prm -allows specification of rivers entering the domain from outside
events.prm - specifies the time of specified programmed "events'

Output files (most are written in subroutines in ***read_and_write_data_files.90*** – see the program routines for formatting and further explanation – most of these consist of several concatenated records of data output at intervals during the simulation in the same general format as ***inelev.dat*** – relatively important output files are indicated in bold italics). Some files will not be created unless the appropriate processes are included in the simulation, and some will be created but not written to.

basin.lst -a text file that summarizes the simulation parameters plus a good bit of data on the progress of the simulations and a variety of rate-process information

outelev.dat -a text file of the surface elevations written at various times during the simulation. This file usually includes several sequential datasets. However, other simulation variables can also be written to this file if desired. Controlled by the input vector *writetype* read in in ***marssim.prm***
continue.dat -a binary file of most simulation variables to be used to restart the simulation (not up-to-date in this release – do not use)
alluvial.dat -an ascii file of 0's to indicate locations that are bedrock channels, and 1's for alluvial channels
discharge???.raw - raw image files of the logarithm of discharges within the drainage network. Normalized so that low discharges are black, highest is white. The dimensions of these images equals that of the elevation matrix.
bedrock.dat -an ascii file of 0's for regolith-covered locations, and 1's for bare bedrock
submerge???.raw - raw image files indicating submerged (black) and unsubmerged (white) locations. The dimensions of these images equals that of the elevation matrix.
submerged.dat-an ascii file of 1's for underwater locations, otherwise 0's for subaerial
erosion_depth_index.dat -if variable rock resistance is used, this is the z-index of the surface in the 3-D rock resistance file

outbase.dat - the elevation of the bottom of alluvial deposits, or, where they are absent, the land surface elevation.
regolith.dat -the regolith thickness
deform.dat -if the rocks are actively deformed, writes the total amount of deformation during the simulation
resist.out -writes the erosional resistance for rocks at the surface (for variable rock resistance is a slice through resist.in
report.prn -a record of relief and erosion rate
record.dat -a record of several variables expressing the progression towards a steady-state landscape
summary.dat - some of the data printed out in basin.lst, but in bare-bones format
statistics.prn – writes a sampling of the simulation state for emergent points at intervals during the simulation
channel.dat -information on stream channels
crater.dat -information about individual simulated impact craters
relele???.raw -files consisting of raw b&w images of surface elevation- scaled so that the lowest elevation is black and the highest is white. The dimensions of these images equals that of the elevation matrix.
topo.dat -a file that gives information on the elevation range corresponding to the images in *relele???.raw*
bshade???.raw -shaded relief images of the surface topography, output in sequential order during the simulation and periodic intervals. Can be put together in Adobe Imageready to make a movie. If periodic boundary conditions are used these files past strips from the opposite side in order to better portray the topography. If neither boundary is periodic the image size is (2*(MX-1), 2*(MY-1), If the image is periodic in, say, the X dimension, then the horizontal image size increases to 2*(MX-1)+2*(MX/2+2) and similarly if the image is periodic in the Y dimension.
grad_disch.dat – file summarizing gradient and discharge within the matrix at the close of the simulation – to have this printed out change *writedetail* to true in **finalize_fluvial_slope_erosion**
state.dat – file summarizing flow directions, elevation, gradients, etc at the close of the simulation - to have this printed out change *writedetail* to true in **finalize_fluvial_slope_erosion**
bistable.dat – output file of locations that are (1) and are not (0) in the accelerated erosion state.
debug.prn - A file of debugging data written by **write_debug**
source.dat - A file of stream source information written by **stream_network_properties** if it is utilized
qq.dat -if groundwater flow is simulated, is the matrix of groundwater discharges
ewater.dat -if groundwater flow is simulated, is the water table elevation
active.dat -if groundwater sapping is simulated, is locations that are presently undergoing sapping erosion.
lava.dat -if lava flows are simulated, whether lava has been deposited at that location
lactive.dat -if lava flows are simulated, is the locations where lava flows are active
lage.dat -if lava flows are simulated, is the age since the last lava was deposited at that location.
eolian.dat -if eolian deposition is simulated, is the amount of eolian deposition or erosion

Accessory Programs (These are included in ***accessory_programs.zip*** **:**

extract.F -reads *outelev.dat* and extracts a specific output record to write out in the file *lastelev.dat*

tosurfer.F -like *extract.F,* but outputs an ascii ".grd" file for input into the commercial program **Surfer**.

Fromsurfer.F – just the opposite of the above.

Tologsurfer.F – makes a surfer file with a logarithmic transformation of the input file.

matrix_2D.F90 -makes a pseudo-random 2-D matrix to be input as initial elevations. See the program for documentation. Output file is named *matrix_2D.out*. A shaded-relief image *m3dshade.raw* is also created.

matrix_3D.F90 –makes a pseudo-random 3-D "cube", primarily for use as rock resistance input. See the program for documentation. Output file is named *matrix_3D.out*, and is a direct-access, unformatted file. For use as rock resistance input, rename this file to *resist.in* . See the program and *boundary_conditions.F90* for how to create and read the file.

Makecrater.F90 – this is largely equivalent to the routine in the main marssim program, but has some extra options. It will create a specified number of impact craters randomly located and drawn from an inverse power law distribution of diameter. Mostly used to create initial conditions files for simulation runs. It can, for example, be used to superimpose craters on topography produced by the matrix_2D program.

Makecrater_xy.F90 – Like above, but the size and location of impact craters are specified. It can, for example, be used to create a single crater to be subsequently modified by other processes.

Colorize_movie.F90 – can be used to create combining a shaded relief images with color elevation cuing to be made into a move. See, for example the animated gif on the home page at http://erode.evsc.virginia.edu .