Documentation for SINOUS Meander Evolution Model

Version 1: July, 2021

This is a guide to the meander evolution model "SINOUS" that has evolved from Howard and Knutson (1984). Additions to this model include floodplain topographic modeling, a variety of chute cutoff mechanisms, a variety of bank erosion formulations, and sediment routing and long-profile modeling. The flow model is based upon Ikeda, Parker, and Sawai (1981) and Johannesson and Parker (1989). Those using the program should refer to the papers in the Bibliography for more specific descriptions of the model framework and example results.

Feedback (suggestions/errors/corrections) on the ***sinous*** program or the present document are welcome to *ahoward@psi.edu* .

Bibliography:

Ikeda, S., Parker, G., Sawai, K., (1981). Bed theory of river meanders, 1, linear development. *Journal of Fluid Mechanics*, 112, 363-377.

Johannesson, J. and Parker, G. (1989). Linear theory of river meanders. In S. Ikeda and G. Parker (eds), *River Meandering*, Water Resources Monograph 12, American Geophysical Union, 181-214.

Howard, A.D. and Knutson, T.R. (1984). Sufficient Conditions for River Meandering: A simulation approach: *Water Resources Research*, v. 20, p. 1659-1667.

Howard, A. D., (1992). Modeling channel migration and floodplain sedimentation in meandering streams: in *Lowland Floodplain Rivers: Geomorphological Perspectives*, edited by P. Carling and G. Petts, Chichester, John Wiley and Sons, Ch. 1, p. 1-41. *[procedures for flow modeling are incorporated in Appendix A]*

Howard, A. D., (1996) Modelling channel evolution and floodplain morphology, book chapter for M. A. Anderson, D. E. Walling, and P. D. Bates, editors, *Floodplain Processes*, Chichester, John Wiley & Sons, p. 15-62.

Matsubara, Y., and Howard, A. D., (2014) Modeling planform evolution of the mud-dominated Quinn River, Nevada, USA, *Earth Surface Processes and Landforms,* 39, 1365-1377, DOI: 10.1002/esp.3588.

Description of Model Components:

The model is programmed in Pascal - a cutting-edge language in 1984, but now gathering dust by the wayside. A free and efficient "Free Pascal" compiler with a visual IDE is available from *www.freepascal.org* for most common operating systems.

Flow model: The model can utilize either the Ikeda et al. (1981) or the Johannesson and Parker (1989) flow model. The model uses a downstream-marching autoregressive solution using weighted values of upstream centerline curvature, as described in Howard (1992), Appendix A. The basic flow and transport input parameters are:

- β Cross-stream slope effect on cross-stream bedload transport
- M Exponent relating velocity to bedload transport rate

F_0 Froude Number: $F_0 = U_0/(g H_0)^{0.5}$, where U_0 and H_0 are reach-averaged velocity and depth for a straight channel with gradient equal to the valley gradient, and g is the gravitational acceleration.

γ_0 Channel width/depth ratio, $\gamma = W/H_0$, where W is channel width

C_f Coefficient of friction, $C_f = (u_*/U)^2$, where u_* is the shear velocity

Note that channel width, channel gradient, sediment properties, and discharge are only expressed in terms of their effect on the above listed parameters. The F-component of the model can only be used for meandering channels where downstream effects dominate (sub-resonant conditions) over upstream effects (super-resonant conditions). If the specified values of the flow parameters imply super-resonant conditions the program will crash if the sediment continuity condition (F-velocity) solution is included. The model can still be run for super-resonant conditions if only the curvature-related (C-velocity) solution is used. In particular, super-resonant conditions primarily occur for large γ_0 and imply upstream bend migration. See, e.g., Lanzoni, S., and Seminara, G., (2006). On the nature of meander instability, *J. Geophys. Res.*, 111, F04006, doi:10.1029/2005JF000416.

Meander evolution: The channel is represented by nodes representing the centerline. In the default implementation, the channel width is assumed to be constant for all nodes, but width can optionally be specified as a function of centerline curvature. The nodes are represented as a linked list, where the next node downstream is represented as a link to the each node progressing downstream. A number of variables are represented for each node. The model progresses through equal timesteps where the direction of migration is normal to the channel centerline and the default rate is controlled by the near-bank velocity perturbation, u_{1b}, calculated by the flow model multiplied times the bank erodibility. Optionally, the bank erosion rate can be an additive weighted function of u_{1b} and the near-bank depth perturbation h_{1b} less a critical value, π, such that no migration occurs unless the calculated value is greater than zero. Finally, the migration rate can be made to also depend upon the bank height, as discussed in Matsubara and Howard (2014).

Node Spacing: As the channel migrates, the spacing between node points representing the channel centerline can increase where the channel length expands or decrease where the centerline contracts. Two node-spacing thresholds scaled by channel width are imposed, such that nodes are eliminated if the spacing drops below the lower threshold and added when the spacing increases beyond the upper threshold. When a point is deleted, the X-Y location of the point upstream from the deleted point becomes the average of the existing location and the location of the point downstream from the deleted point. When a new point is inserted the new point is located to lie on the circle fitted to the two centerline points upstream and the point downstream from the location of insertion.

Neck Cutoffs: After each number of iterations specified by the loopcheck parameter have taken place, the program searches downstream from each node to each node downstream looks to see if a node lies closer than a threshold distance, scaled to channel width, of the given point. If that occurs, locations between the upstream node and the downstream close node, including the downstream close node are eliminated.

Chute and Avulsion Cutoffs: The model will optionally determine if chute and avulsion cutoffs will occur. As with neck cutoffs, the evaluation progresses downstream for potential locations that will be the end, receiving, node for the cutoff. A variety of cutoff criteria can be employed as discussed more fully in Howard (1996). For these longer cutoffs new nodes are generated along the straight-line path between the upstream cutoff starting node and the downstream receiving node. Five planimetric and relief properties are considered as

possible contributors to chute cutoffs: (1) The ratio R_c of gradient across the potential shue path to the existing channel gradient; (2) the distance D_c across the potential cutoff; (3) the elevation E of the floodplain across which the chute develops; (4) the planimetric angle ψ between the existing channel direction and the path of the chute, and (5) the relative magnitude of the near-bank velocity of the potential chute. The probability P of chute development at a given time and across a given chute path is expressed as:

$$P = K_c R_c \exp\left(-K_d D_c - K_e E + K_a \cos\psi + K_v u_{1b}\right) \text{ , where} \quad \text{(a)}$$

$$R_c = (D_p - D_c) / D_p = (\mu_c - 1) / \mu_c \ . \quad \text{(b)}$$

The gradient ratio R_c depends on the relative distances (and hence also overall gradients(across the existing flowpath D_p and across the potential chute D_c, and it can also be expressed in terms of the cutoff sinuosity $\mu_c=D_p/D_c$. In evaluating the cutoff the downstream channel gradient is assumed to be uniform. The terms in the exponential reflect the presumed smaller probability of cutoff across longer chute lengths and floodplains with higher elevation (E can be specified either as the maximum or average elevation across the chute, measured relative to the average bed elevation). Additionally, the probability of cutoff is assumed to decrease as the angle between the existing flowpath and the cutoff path increases. Finally, the probability of cutoff is assumed to increase if the near-bank velocity perturbation u_{1b} is large and positive. The coefficients K_c, K_d, K_e, K_a, and K_v are assumed to be temporally and areally invariant. The sum of the exponent terms is restricted to values ≤0.

Floodplain Evolution: Optionally deposition on, and erosion of, the floodplain can be modeled. The floodplain is represented as a fixed spatial matrix of locations overlaid on the channel centerline. The size of the floodplain domain is set by xpoints and ypoints in the **const** header of the program. The default values are 700 and 200 channel width-equivalents, respectively. It would be useful to have execution-time array definition as with "allocation" in Fortran; there can be dynamically created arrays in Free Pascal, but using them is more complicated. So just recompile the program with larger or smaller xpoints and ypoints. The floodplain is divided into cells with dimension of [one ***x*** one] channel width, and distances are measured in channel-width equivalents. The centerline of the channel migrates across the spatial domain, and the location of the centerline relative to the fixed floodplain locations is calculated. The general procedures are discussed in Howard (1992, 1996). Erosion only occurs when the channel migrates through a floodplain node, with the new elevation of the node being set by the channel depth on the distal bank of the channel as it migrates across the node. The rate of deposition is modeled as a function of distance from the nearest node along the channel and the existing floodplain level. The deposition rate, ϕ, is specified by (Howard, 1996):

$$\phi = K_d \left[\nu + \mu e^{-D/\lambda}\right] e^{-\eta\left(E_a - E_b\right)^2} \ , \quad \text{(c)}$$

where K_d is the deposition scaling rate, ν is a background sedimentation rate, μ is a distance-dependent rate constant, D is the distance from the nearest channel node, λ determines the rate of falloff with distance, η scales the rate of diminishment of deposition with floodplain elevation, E_a is the floodplain elevation, E_b is the mean bed elevation of the nearest channel node. Note that advection of sediment across the floodplain is *not* directly modeled. Howard (1992) used a simpler deposition rate formula:

$$\phi = K_d \left(E_m - E_a\right)\left[\nu + \mu e^{-D/\lambda}\right] \ , \quad \text{(d)}$$

where E_m is the maximum permissible floodplain height relative to the channel. The use of Eq.(d) versus Eq. (c) can be set by the parameter linearrateuse with the values 1 and 0, respectively.

If the floodplain is spatially modeled, then the bank erodibility can be modeled as a function of the bank type and bank properties. For example, bank erodibility can be set to zero or to a fixed low value to model erosion of valley walls when the channel migrates against the valley wall. Similarly, sediment deposited in cutoff oxbow lakes can be made more resistant (i.e., clay plugs). Examples of simulations employing location-dependent bank erodibility are presented in Howard (1996).

Downstream Sediment Transport: Optionally, bed sediment can be routed through the channel system evolving the channel profile. The downstream routing procedure adapts the backwater aggradation and degradation model using the 1D Sediment Transport Morphodynamics ebook of Gary Parker, RTe-bookAgDegBW.xls:

http://hydrolab.illinois.edu/people/parkerg/morphodynamics_e-book.htm

This part of the meander model has been recently developed and tested to examine the effects of cutoffs on channel migration patterns as well as sediment-flux dependent bank erosion. The model parameters and formulation is discussed in the detailed model parameter specifications, below, but results of simulations have not been published (*caveat emptor*).

Input Parameter Files Description. In the Type column the values are:

A = Text
I = Integer
F = Floating point (Real)

Main Parameter file (*meander_parameters.prm*)

Parameter	Type	Description
description	A	Line of descriptive text
iteration	I	Number of Iterations
updateint	I	Number of iterations between updating floodplain data and reporting iteration number, elapsed time, and bank erosion rate
printint	I	Number of iterations between output of variable data and tables
loopchkint	I	Number of iterations between checks for neck and chute cutoffs
plotint	I	Number of iterations between data output
miniteration	I	The number of elapsed iterations before data output starts
nom_time	F	The nominal time increment, in years
nom_rate	F	The nominal maximum bank erosion rate in meters/year
maxinterval	F	The maximum permitted time increment, in years
ratemultuse	I	Selection parameter for sinuosity correction on channel gradient (0=No, 1=Yes) assuming constant valley gradient
slope_factor	F	Sinuosity-gradient Exponent

min_adj_factor	F	Causes node deletion when node spacing, in channel width units, drops below this value
max_adj_factor	F	Causes new node insertion when node spacing, in channel width units, increases above this value
max_distance	F	Maximum distance, in width units, to check for cutoffs
ndim	I	Number of upstream nodes used in flow calculations
usenewparker	I	Whether to use the original Ikeda et al. (0) or Johannesson & Parker flow and bed topography model (1)
fvelo_use	I	Use only the curvature forced bed and flow perturbations (C_velocity) (0) or include the sediment continuity terms (F_velocity) (1)
mconst	F	Exponent relating bed sediment flux to mean velocity [M]
betaconst	F	Cross-stream slope effect on cross-stream bedload transport [β]
depthweight	F	Relative weighting of depth perturbation in migration rate calculation [0.0≤weighting≤1.0]
velocityweight	F	Relative weighting of velocity perturbation in migration rate calculation [0.0≤weighting≤1.0, depthweight+velocityweight=1.0]
loopcheckfactor	F	Neck cutoffs occur when downstream channel node is closer than this distance (in width units) to an upstream node
bias	F	downstream bank erosion bias (<=0.0 for no bias, 0<bias<1.0) [***recommend not using bias***]
start_time	F	The initial time, generally 0.0 unless continuing an existing simulation.
input_print	I	=1 if initial configuration is output, otherwise 0
valley_width	F	valley width, in channel width units, <0.0 for unlimited width
twowgt	F	The weighting given to the second node calculated migration rate. For stability, this is usually set to 0.0
detailprint	I	Selects whether output files contain detailed variable information for each node [0=no, 1=yes]
firstwrite	I	Output of node-related data starts at this node, =0 for all nodes
ideposit	I	=0 if floodplain deposition and erosion is not modeled, =1 for floodplain modeling
readdeposit	I	=0 floodplain elevations set to default values, =1 floodplain data read from file
elevrate	F	rate of floodplain deposition [meters/iteration] [μ in Eq. (3)]
maxelevation	F	maximum permitted floodplain elevation [meters]. This is only used if linearrateuse is set to 1, so that the

		Howard(1992) floodplain deposition rate law (Eq. (d)) is used instead of Howard(1996) (Eq. (c))
elevchannel	F	elevation of channel bed [meters]
depthfraction	F	mean water level as a fraction of elevation difference from channel bed to maximum floodplain elevation [0<value<1]
floodrate	F	relative background floodplain deposition rate [v in Eq. (c)]
bankrate	F	relative overbank deposition rate [η in Eq. (c)]
bankdistancefactor	F	distance decay factor for overbank deposition [μ in Eq. (c)]
probcutoffuse	I	Use a probability cutoff for chute cutoffs [0=No, 1=Yes]
velocityfactoruse	I	Use the near-bank velocity perturbation in chute cutoffs [0=No, 1=Yes]
anglefactoruse	I	Use the angular deviation in chute cutoffs[0=No, 1=Yes]
maximumelevationuse	I	Use the maximum floodplain elevation in chute cutoffs [0=No, 1=Yes]
averageelevationuse	I	Use the average floodplain elevation in chute cutoffs[0=No, 1=Yes]
anglefactor	F	Weighting for angular deviation ψ in chute cutoff [K_a in Eq. (a)]
velocityfactor	F	Weighting for velocity perturbation u_{1b} in chute cutoff [K_v in Eq. (a)]
elevationfactor	F	Weighting for elevation factor E in chute cutoff [K_e in Eq. (a)]
distancefactor	F	Weighting for cutoff distance in chute cutoff [K_d in Eq. (a)]
minperpdist	F	Minimum perpendicular distance between the existing channel centerline and the cutoff path. This assures that trivial cutoffs do not occur
aggraderate	F	In floodplain modeling, the channel bed will aggrade relative to the floodplain at this rate if >0, or degrade if <0 (meters/iteration)
depositdecay	F	Elevation decay factor for deposition rate in [γ in Eq. (a)]
bankelevuse	I	This is an index for setting initial bank elevation. If 0 then the bed elevation is used, if 1 then the elevation is set to the local stream bankdepth times meandepth.
linearrateuse	I	If this is 1 then the bank deposition formula from Howard(1992) (Eq. (d)) is used, and if 0 then the Howard(1996) value is used (Eq. (c))
skippoints	I	Skip points during cutoff check (0=No, 1=Yes)
probmult	F	Multiplicative factor in chute cutoff probability [K_c in Eq. (a)]
erodenominal	F	Nominal bank erodibility

erodeplug	F	Nominal plug erodibility relative to nominal bank erodibility
erodevalley	F	Nominal erodibility of valley walls
plugdistance	I	Distance from active channel before node in an abandoned channel receives deposition of sediment with plug erodibility
readerodibility	I	Determines whether floodplain erodibility is read from a file (0=No, 1=Yes)
stickyuse	I	0 for less sticky and 1 for more sticky banks and plugs
critical_ustar	F	Critical $u*$ value for bank migration
widthvariation	I	Vary channel width as a function of local curvature (0=No, 1=Yes)
curvature_width_variation	F	Slope of width versus curfature relationship (may be +, -, or zero)
use_nonlinear_bank_erosion	I	Use a non-linear relationship between u_{1b} and bank erosion rate. (0=No, 1=Yes)
bank_erosion_exponent	F	Exponent for non-linear bank erosion rate (if used)

Parameter file for Johanneson&Parker-related parameters (*jp_parameters.dat*). Note that this is a read/write file because they may vary during certain types of simulations.

Parameter	Type	Description
channel_width	F	Channel width (meters)
cfriction	F	Coefficient of friction, C_f
wdratio	F	Width/Depth ratio, γ_0
froude	F	Froude Number, F_0

Parameter file for parameters related to downstream sediment routing (sediment_parameters.prm). These relate to parameters in the Parker ebook RTe-bookAgDegBW.xls

Parameter	Type	Description
channel_width	F	Channel width (meters) [the value entered here should be the same as that in the ***jp_parameters.dat*** file]
initial_gradient	F	Initial gradient. For new run the profile will be linear with this gradient
discharge	F	Formative discharge, in m^3/s (e.g., mean annual flood
shear_exceed_factor	F	shearexceed factor taustar/tauc (≤1.0 is a threshold channel, >1 for active transport
transport_increment	F	Transport increment, in years (simulation uses subiterations for sediment transport - this determines number of subiterations per master migration iteration
use_grain_roughness	I	bed friction only used grain size (0=No, 1=Yes)
roughness	F	roughness height

sediment_transport_exponent	F	Exponent in sediment transport relationship
transport_factor	F	Multiplier in sediment transport relationship
flow_fraction	F	Fraction of year with formative discharge (0.0<fraction≤1.0)
transport_crititcal_dim_shear	F	Transport dimensionless critical shear
upwind	F	Upwinding factor in sediment transport calculation (range from 0.5 to 1.0)
bedmaterial_relative_depth_fraction	F	Relative depth of bed sediment in bank/bar deposits
maximum_faults	I	maximum number of successive negative predicted depths (faults) to stop program (because of predicted supercritical flow)
reset_number	I	Number of depth predictions iterations without a negative depth to reset the number of faults
smooth_method	I	Smoothing of large cutoffs (0=None, 1=cutting (non-conservative), 2-regression (conservative))
smooth_length	I	Initial range up and downstream for smoothing, N, so that smoothing goes from =N to +N
provisional_use	I	Use provisional elevations in iterative smoothing of profile (0=no, 1=yes)
max_fault_iterations	I	Maximum number of iterations for profile smoothing
use_normal_approximation	I	Use normal flow approximation (0=use backwater profile, 1=use normal flow approximation)
bedload_width_use	I	Use channel width beinf a fuction of bedload transport rate (=-no, 1=indirect, 2=flux-width)
bedload_width_constant	F	Bedload width constant; ~50.0 for indirect, ~1.0 for flux width. (constant>0.0)

Initial.prm **contains miscellaneous parameters that are not often modified.**

Parameter	Type	Description
seed	F	Floating point random number seed
similpar	F	Similarity parameter for resitance (0.0≤similpar≤1.0)
error	F	Stamdard error for resistance (>0.0 for random bank resistance)
logstdev	F	Discharge standard error (>0.0 if discharge is randomly varied
res_type	I	=0 for no bank resistance averaging, else 1

Sediment_flag.prm **has a single integer parameter, usesedimentroute. If this is 1 then downstream sediment routing is used, else 0 for a constant gradient valley profile.**

Data input files.

Indata.dat **contains information on X,Y coordinates of initial stream nodes. There are 2 columns of floating point data. IMPORTANT NOTE: The node coordinates are input from the downstream end to the upstream beginning node.** If natural stream data is input the channel should be ordered from the downstream end to the upstream end. For digitized channels the downstream centerline point spacing should approximate one channel width.

Xvalue - the X coordinate of the node in meters
Yvalue - the Y coordinate of the node in meters

If the purpose of the simulation is to try to match the pattern of migration of a real stream, then the accessory program ***test_fit.f90*** will measure the included area between the simulated meandering and the target real stream The digitized initial stream centerline would be the older real planform and the match would be to the younger real planform. The iteration number for the lowest included area (if other than the inital included area) would constitute the best fit. This can be used, for example, to calibrate the bank erodibility so that iteration number can be related to actual elapsed time. The distribution includes a Powerpoint ***madidi_river.pptx*** showing this type of simulation.

If the input parameter readdeposit is set to unity, the starting values of floodplain age, age and distance to the nearest channel are read in from the file ***aelev.dat***. This would typically occur if the simulation is restarting an earlier simulation. Similarly, if the parameter readerodibility is greater than zero, the bank erodibility values are read from ***inebank.dat***.

Data output files. Several of these are only written for specific types of simulations or if specified to be written by a flag in the parameter files.
Files related to the progress of the meandering:

list.dat contains a general report on the simulation, including input parameters, calculated variables, and the progress of the simulation.

outdata.dat contains a listing of the stream size and X,Y node positions written each plotint number of iterations.

plotdata.dat contains a listing of stream node X,Y locations as well as 18 associated node variables written each plotint number of iterations. See the procedure writelongfile in the source program for a listing of the variables.

average_values.dat contains average values of 13 node variables written each plotint number of iterations. See the procedure writelongfile in the source program for a listing of the variables.

debug.dat As configured in the program in the write_debug procedure the stream node bed elevation, local depth, local gradient, sediment flux and local cfriction are written each printtime number of iterations. This procedure can be reconfigured to output any desired simulation state data.

vector.dat primarily prints diagnostic information for simulations using downstream sediment routing. See the procedure write_vector for the 10 variables printed out. This is called every printtime number of iterations.

jp_parameters.prm At the end of the simulation this input file is overwritten with the final values of the parameters that can vary through time. Thus the file can be used in conjunction with restarting the simulation.

timestat.dat This details the evolution of the simulation by writing every updateint iterations the iteration number, simulation time, sinuoisity, and average migration rate.
detail.dat If the detailprint parameter is 1, then, at the end of the simulation, this prints out variable data for each stream location between firstwrite and lastwrite the X,Y location, planform curvature, C_rate, F_rate, CS_curvature,localgradient, localdepth, sedimenfluc, bedelevation cdepth, fdepth, sense, and newsense.
bedprofile.dat If sediment routing is modeled, then this writes out at the beginning of the simulation the X,Y locations of the nodes and the bed elevation.

Files related to cutoffs:

neckcapt.dat is written to after each neck capture. It contains the iteration number, the X,Y location of the capture node and the receiving node, the distance across the capture, and the cutoff geometry. See the procedure cutoff for details.
chutecapt.dat is written to after every chute capture. it contains the iteration number, the X,Y location of the capture site, the chute geometry, and the values of the parameters in Eqs. (a, b) relating to the capture probability. See the procedure checkloop for details.

Files related to floodplain modeling (if the parameter usedeposit is true):

aelev.dat is written at the end of the simulation and contains, for each floodplain cell, the I,J coordinates, the age since last occupied by the channel, the elevation, and the distance to the nearest channel.
outbank.dat is written at the end of the simulation and contains for each floodplain cell the I,J coordinates and the bank erodibility, This file and the ***aelev.dat*** file can be used as inputs to restart the simulation.
elev.pgm is written at the end of the simulation and is a greyscale image scaled to the elevation range of the floodplain. It can be read by Photoshop or Gimp.
age.pgm is written at the end of the simulation and is a greyscale image scaled to the age range of the floodplain cells. It can be read by Photoshop or Gimp.
agesum.dat writes statistics about the ages, elevations, and distances to the nearest channel.
All the above files are written in the procedure writeageelev, and the specifics formats and variables output are detailed there.

Example Simulations: Parameters and results from three simulations are included with the program source. These are indicated by the subdirectory containing the files. If the parameter and indata.dat files from these directories are copied to a new directory and ***sinous.exe*** is executed, the resulting output can be compared with that in the example directory.

Planform_evolution_example. This is a simple simulation starting from a nearly linear equally-spaced nodes along the x-axis with slight initial random perturbations from zero along the y-axis. The initial planform is read from ***indata.dat***. The node spacing is equal to the channel width. Note that nodes are read from downstream to upstream. The accessory program ***make_meander_input.f90*** was used to create the initial planform, with parameters read from ***make_meander.prm***. The input and output files are described above.

Floodplain_deposition_example. This simulation is an example of modeling the topographic evolution of the floodplain as the stream meanders. The initial planform is the

output from the planform_evolution_example run. The final node locations in ***outdata.dat*** are processed by the accessory program ***invertdata.f90*** using parameter data in ***invertdata.prm*** which produces the output file ***inverted.dat*** which was copied to ***indata.dat*** in the floodplain_depsition_example directory. The input and output files are described above. In particular, the image files ***age.pgm*** and ***elev.pgm*** files give a quick visual appraisal of the resulting age and elevations of the simulated floodplain.

Sediment_routing_example. This simulation is an example of modeling both planform and profile evolution of the channel by downstream sediment routing. The routing procedure can be used to examine the effect bed elevation perturbations caused by cutoffs upon subsequent meander evolution or by inputting initial sediment load parameters that are out of equilibrium with the initial channel gradient and changes in meander patterns caused by subsequent gradient adjustments. The initial centerline is the 1984 centerline of the Madidi River in the Amazon region extracted from Landsat imaging by procedures developed by Alex Bryk (U. Cal. Berkeley). This example also illustrates testing predicted meander evolution against actual river migration over a 5 year period. The program ***test_fit.f90*** compares the simulated planform through time as compared to the actual 1989 planform of the Madidi River. The parameters for the comparison are read from ***test_fit.prm*** and the quality of the fit through time is assessed as the included area between the simulated planform and the target final natural channel planform as reported in ***diff_result.dat***, with the minimum included area as the best fit. The model time for the best fit can be used to calibrate the best estimate of the bank erodibility parameter (see, e.g. Matsubara and Howard (2014)). See the Powerpoint ***maddidi_river.pptx*** for further exposition of this modeling approach.

Accessory Programs: A number of programs are included in the *accessory_programs* folder. Some are written in Pascal and some in Fortran90. Windows executable versions of the programs are also included. These are listed below:

make_meader_input.f90 creates an initial planform of nearly linear equally-spaced nodes along the x-axis with slight initial random perturbations from zero along the y-axis. The channel length, channel width, and degree of random perturbations are read from ***make_meander.prm***. Such a nearly straight initial planform is useful for investigating the intrinsic meander wavelength and evolution of meander planform through time as a function of the model parameters. The initial spacing of nodes is one channel width.

invertdata.f90 takes output data from a simulation in the ***outdata.dat*** file which is written as node X,Y pairs in the downstream direction and inverts the order of the nodes to make a file ***inverted.dat*** which is suitable to be copied to an ***indata.dat*** file for a subsequent simulation from the final planform of the previous simulation. The input and output files are specified in ***invertdata.prm***.

test_fit.f90 compares the sequential output planforms created by a simulation in ***outelev.dat*** and compares them to a target natural planform later than the initial input planform for the simulation to find the best-fit time and degree of miss-match between the

planforms as measured by the area included between successive crossing of the planforms. See the description above in the *sediment_routing_example*.

Space.pas takes an input file of X,Y nodes (often a digitized or otherwise extracted centerline of a natural river) in the file ***space.dat*** and outputs nodes with node spacing (sample_length) specified in ***space.prm*** and outputs a file **stat.dat** whose X,Y nodes are equally spaced. The ***sinuous*** program enforces node X-Y spacings within a narrow window around the channel width value by either eliminating nodes too closely spaced or inserting nodes between nodes too widely spaced. To avoid excessive (and possibly inacurate) initial adjustments of the nodes read in from ***indata.dat*** the spacing.pas program produces equally-spaced nodes. The sample_length parameter should be set to the channel width specified in the parameter files for the ***sinuous*** program. If the input data contains X,Y pairs ordered downstream the data should be input into ***invertdata.f90*** to reverse node spacing and make an ***indata.dat*** file required for the ***sinuous*** program.

combine_digit.f90 creates a single output file of equally spaced X,Y nodes from three input files of centerlines digitized over the same reach of a natural rivers in a downstream direction. The averaging of data from the three centerlines produces an output file which presumably contains less 'noise' resulting from the digitizing process. Ideally the digitized data should have node spacing less than or equal to a channel width. The resulting output file of averaged data should be input to ***space.pas*** to regularize the spacing at the specified channel width to form an input file for the simulation.

There are a series of programs that analyze centerline data output from the ***sinuous*** program to extract statistical information such as emergent wavelength, sinuosity, loop assymetry, and other loop statistics for comparison with similar centerline data for natural streams. These analytical programs will be made available as a separate program release.

Description of Model Procedures

READ1PARAMETERS; First of three procedures to read the parameter values in the *meander_parameters.prm* file.

READ2PARAMETERS; Second procedure to read parameter values.

READ3PARAMETERS; Third procedure to read parameter values.

READ_ROUTE_PARAMETERS; Reads the sediment routing parameters from the *sediment_parameters.prm* file

FINDDISTANCE; calculates the straight-line distance between two nodes from their X and Y coordinates.

%%The next several procedures evolve the bed profile by doing bed sediment routing%%

WRITE_DEBUG; Writes values of bed-elevation, depth, channel gradient, sediment flux, and friction of all active nodes to the file *debug.dat*.

WRITE_VECTOR; Writes debugging values of a vector of points when smoothing the stream profile to the file *vector.dat*. See procedure for a list of vector variables written.

SMOOTH_PROFILE; This replaces the elevation of a node by fitting a constant-gradient profile starting from the elevation of a node N nodes upstream to the elevation of a node N nodes downstream and replacing the elevation of the central node by the value given by the constant-gradient profile. Used in sediment routing.

REGRESS_PROFILE; This replaces the elevation of a node by fitting a straight line regression of the elevation of the N nodes upstream and N nodes downstream of the location and replacing the elevation of the central node by the value given by the regression. Used in sediment routing.

MAKE_VECTORS; This constructs vectors of distance downstream, bed elevation, and channel gradient for the river. Used in sediment routing

CHECKPROFILE; This examines predicted values of channel depth to see if negative or large flow depths are predicted and if so, calls SMOOTH_PROFILE or REGRESS_PROFILE.

PARKER_BACKWATER_TRANSPORT; This routes bed-sediment downstream by first constructing the backwater water elevation profile, calculating flow depth and velocity, and determining elevation changes by calculating sediment flux divergence to deposit or erode the bed. This uses the Parker backwater profile sediment aggradation and degradation routine.

PUT_VECTORS; This determines vector values of depth, velocity, bed elevation, and sediment flux based upon changes predicted by PARKER_BACKWATER_TRANSPORT.

EVOLVE_STREAM_PROFILE; This procedure evolves the channel bed profile and sediment flux by calling PARKER_BACKWATER_TRANSPORT several iterations per master channel meander iterations. The number of subiterations is given by the parameter TRANSPORT_ITERATIONS.

DO_REGRESSION; ***this procedure is not currently used***

SET_INITIAL_ELEVATIONS; This creates the initial stream profile for simulations using sediment routing using the INITIAL_GRADIENT parameter and sets initial values

of velocity, flow depth, channel width, and bed friction using initial values that are read in.

%%The next several routines evolve the floodplain properties and elevation for a 2-D matrix of locations through which the channel migrates. The floodplain properties may in turn affect the channel migration.%%

WRITEBANKDATA; This writes an output file of integer values depending upon the whether the floodplain cell has normal or plug erodibility.

FINDAGE; This increments the age of each floodplain cell from the start of the simulation or the last time the channel has migrated through that cell, whichever is smaller. It also determines the elevation of the channel bank if the cell has been migrated through.

FINDCHANNELDISTANCE; Given a cell I,J location this calculates the real distance from the channel to the cell

DEPOSIT; This deposits sediment on the floodplain as a function of the distance from each cell to the nearest channel and resets the elevation to the channel bed elevation when the channel migrates through the cell.

FINDRANGE; This locates the channel node closest to each floodplain cell and the integer distance from the channel to the cell.

SETUPFLOODPLAIN; This initializes the variables defining the state of the floodplain, including the floodplain age, erodibility of the floodplain deposits.

WRITEWIDTH; This writes out the vertical width of the floodplain has been reworked by channel migration. Note that this assumes the channel initially has a vertical extent of zero (i.e., follows the X axis).

AGE_ELEVATION; This is the master routine for modeling the floodplain, calling FINDAGE, FINDRANGE, DEPOSIT, and WRITEWIDTH sequentially.

WRITEAGEELEV; This routine provides the primary output of the state of the floodplain, printing, for each I,J location the location, age, elevation, and integer distance to the nearest channel. It also writes binary image files of floodplain age and elevation, as well as several other summary data files.

%% The next several routines read initial variables and provide statistical characterization of the state of the simulation %%

GETVALUES; This reads several simulation parameters from the *initial.prm* file.

STATVAL; Adds an observation to the statistical database.

STATISTICS; Adds observations about migration rate and radius/curvature of planform to database.

STATINITIALIZE; Initializes the statistics variables.

STATPRINT; Writes out the statistics about radius/width and migration rate to the list file.

%% The next several routines calculate geometry, determine bank erodibility, and add or delete stream nodes %%

FINDPERPDIS; Finds the perpendicular distance between a point and a line defined by two other points.

CURVANGLE; Given three successive nodes on the centerline, calculates the acute angle of intersection of the lines joining the first two points and the line joining the second two points.

INCLUDEANGLE; Like CURVANGLE, but calculates the interior angle.

FINDANGLE; Given two points, calculates the angular orientation of the line extending from the first point to the second.

FINDPLUGS; Determines whether a location on the floodplain is underlain by sediment of normal erodibility or is the location of a clay plug deposited in a cutoff loop.

FINDSTREAMLOC; Given an X,Y location finds the I,J location of a floodplain cell.

FINDERODIBILITY; Determines the bank erodibility of an X,Y location of a stream node. Returns the default erodibility if floodplain modeling is not invoked.

FINDCENTER; Finds the central X,Y location defined by three successive stream nodes.

ADELETE; Deletes the next channel node downstream from a given node, and resets the X,Y location of the given to be midway between its original position and that of the new next downstream node.

FINDNEWPOINT; Determines the X,Y location of a new point inserted between two existing stream nodes based upon the central location and radius of curvature.

INSERTS; This is the main code for inserting a new channel node between two existing nodes when the inter-node distance has increased beyond a critical value. It also assigns values to the variables associated with the new node by averaging the values for the existing upstream and downstream nodes. Calls FINDCENTER, FINDDISTANCE, FINDNEWPOINT.

ADJUST; This cycles through the nodes in the channel database and determines if a node needs to be inserted or deleted, and also adjusts the bank resistance if the channel has migrated more than one channel width.

CUTOFF; This procedure implements neck cutoffs by eliminating channel nodes of the abandoned loop. The procedure is initiated with the X,Y location of the current node and of the downstream location (CHECKPOINT) that will become the new next downstream location.

DOCUTOFF; This procedure implements chute cutoffs by eliminating the cutoff nodes and inserting new nodes along the cutoff path and assigns variable values associated with the new nodes.

CHECKPATH; This procedure checks the probability of a chute cutoff will occur between two nodes along the centerline

CHECKLOOP; This cycles downstream from node to node and determines if a neck or chute cutoff event will occur and calls the appropriate routine.

%% The next two procedures initialize the simulation variables %%

INITIALIZE; This reads in the initial values of the X,Y locations of stream nodes, the elapsed time at the beginning of the simulation, and the initial bank resistance. If the simulation is intended to compare modeled migration to observed natural migration over a time interval the actual final natural stream centerline is read in.

SETUP_ARRAY; This sets up the upstream autoregression array of distances, C_velocities, and F_velocities.

%% This procedure calculates the migration rate of the channel nodes based upon the autoregressive solution %%

ERODE; This calculates the normalized erosion rate based on the curvature and calculated flow and depth perturbations.

%% The next several procedures solve the autoregressive upstream calculations to calculate local velocity and bed perturbations, as outlined in Howard (1992) Appendix A %%

LUDECOMP; This solves for the discriminant of the upstream distance coefficients

MATRIXSOLVE; This solves the matrices determining upstream weights for calculating C_velocities and F_velocities setting up the weighting vectors D1TERM (1st derivative) and D2TERM (2nd derivative)

DERIVWEIGHTS; This sets up coefficients for evaluating the upstream weightings for calculating the C_velocities, and F_velocities, allowing for uneven distances between upstream nodes.

CALCCS; This calculates the secondary circulation term

FVELOCITY; This calculates the velocity and depth perturbations due to bedforms and sediment transport

RATESENSE; This determines whether the migration is to the left or right of the downstream direction.

PRINTDETAIL; This prints out a report of the major variables for a given stream location, including the X,Y coordinates, curvature, local gradient, velocity and depth perturbations.

LASTSTUFF; This migrates the stream and prints out details if it is time to do so

CVELOCITY; This calculates the velocity and depth perturbations due to planform curvature

RATEAVERAGE; This calculates the nominal migration rate based upon the calulated velocity and depth perturbations and other simulation parameters

FINDMAXNEWRATE; Finds the maximum possible values of velocity and depth perturbations and erosion rate.

LOCAL_PARAMETERS; This calculates local values of several flow parameters as well as unique values if channel width is specified to vary downstream.

CYCLE1; This is the main procedure that at each location calls routines to calculate the velocity and depth perturbations, to determine the rate and directional sense of node migration, and to actually migrate the channel. It then moves to the next point downstream until the end is reached.

ITERATE; This sets up variables for the given iteration, calls CYCLE1 to calculate the migration for each node, calculates migration for the last node, and prints out summary details for the iteration.

WRITELONGFILE; This cycles downstream and outputs the values of local variables for each node, then outputs several average values of variables.

WRITESHORTFILE; This just writes out the X,Y values of all stream nodes

BEGIN **[the main program]** This opens the major input and output files, reads simulation parameters and input data, calculates several additional fixed and initial parameters, sets up the floodplain if used, sets up the stream profile if sediment transport is used and then iterates over the specified number of time increments (calling ITERATE), outputs data at specific time increments, and finalizes the simulation and closes files.