TREX Watershed Modeling Framework User's Manual:
Model Theory and Description

Prepared by:

Mark L. Velleux
HDR
1 International Boulevard, 10th Floor, Suite 1000
Mahwah, NJ 07495

John F. England, Jr.
U.S. Army Corps of Engineers
Risk Management Center
12596 West Bayaud Ave, Suite 400
Lakewood, CO 80228

Pierre Y. Julien
Colorado State University
Department of Civil and Environmental Engineering
Fort Collins, CO 80523-1372

AUTHORSHIP AND ACKNOWLEDGEMENTS

This document is the user's manual for the TREX watershed model developed at Colorado State University (CSU). TREX is a generalized watershed rainfall-runoff, sediment transport, and contaminant transport modeling framework. This framework is derived from earlier versions of the CASC2D watershed model as developed at CSU (Julien et al. 1995; Johnson et al. 2000; Julien and Rojas, 2002) with chemical transport and fate processes from the USEPA WASP and IPX series of stream water quality models (Ambrose et al. 1993; Velleux et al. 1996; Velleux et al. 2001).

The major developers of this code and authors of this manual are Mark Velleux (HDR, formerly CSU), John England (U.S. Army Corps of Engineers [USACE] Risk Management Center, formerly Bureau of Reclamation and CSU), and Dr. Pierre Julien (CSU). The contributions of Dr. Rosalia Rojas (CSU) toward TREX code development and for CASC2D-SED source code are gratefully acknowledged. Aditya Maroo (CSU) assisted with programming. Dr. Billy E. Johnson (USACE-ERDC-EL) and Dr. Zhonglong Zhang (SpecPro, Inc.) provided comments and technical suggestions for extended framework development. Concepts for development of the simulation relaunch option were contributed by Dr. Veeradej Chynwat (ENSCO, Inc) and Troy Seiler (ENSCO, Inc). James Hallden and Dr. Masa Takamatsu (HydroQual, Inc.) also provided programming support for TREX development. Model development efforts also include contributions from Daniel Schwartz, Lihong (Lucy) Yang, Christopher Fanelli (HDR|HydroQual). Dr. Glen Liston (CSU) graciously provided source code for MicroMET, of which some elements were adapted for use with TREX. Dr. Jiming Jin (University of Arizona) graciously provided source code for SNTHERM, of which some elements were also adapted for use with TREX. Code modifications for overland transport capacity and shear stress partitioning were an outgrowth of collaboration with Lea Dreier (Leibniz University Hanover).

The TREX user's manual was first developed for the U.S. Army Corps of Engineers, Engineer Research and Development Center, Environmental Laboratory. Subsequent revisions were performed in support of the National Oceanic and Atmospheric Administration, National Severe Storms Laboratory. Snow hydrology features were added and supporting manual revisions completed as part of research and development efforts funded by HydroQual, Inc. Code enhancements and manual revisions associated with the October 2009 TREX release were an outgrowth of a project performed for the American Chemistry Council. Code enhancements and manual revisions for the July 2010 TREX release were developed for ENSCO, Inc in support of a Department of Defense contract. Code enhancements and manual revisions for the February 2011 TREX release were funded by HDR|HydroQual. Code enhancements and manual revisions for the August 2011 TREX release were an outgrowth of a project performed for Myjongi University (Republic of Korea). Code enhancements and manual revisions for the October 2017 TREX release were an outgrowth of the authors' technical interests.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1.0 INTRODUCTION

TREX, the Two-dimensional Runoff, Erosion, and Export model, is generalized watershed rainfall-runoff, sediment transport, and contaminant transport modeling framework. This framework is based on the CASC2D watershed model (Julien et al. 1995; Johnson et al. 2000; Julien and Rojas, 2002) with chemical transport and fate processes from the USEPA WASP and IPX series of stream water quality models (Ambrose et al. 1993; Velleux et al. 1996; Velleux et al. 2001). TREX has three main components: 1) hydrology; 2) sediment transport; and 3) chemical transport and fate. Model theory and process descriptions are presented in Section 2.0. The numerical implementation of the process in the TREX model computer code is presented in Section 3.0. Descriptions of model input files are presented in Section 4.0. Descriptions of model output files are presented in Section 5.0. Developmental features are discussed in Section 6.0. A programming guide, including availability of source code, is presented in Section 7.0.

The code has been subjected to extensive testing to ensure accuracy and error-free performance. However, it should be noted that (like most software) TREX is large and complex and coding errors (bugs) may exist. It is also important to note that some aspects of the TREX framework and model source code are under ongoing development. Where possible, developmental features and code are noted. Inclusion of these development portions of the framework is intended to demonstrate how the basic framework can be readily expanded to permit model use for an even wider range of conditions than can already be simulated. Nonetheless, users are advised to carefully review the TREX source code before use to ensure it performs correctly for any given application.

2.0 MODEL THEORY AND PROCESS DESCRIPTIONS

A review of hydrologic and sediment transport process descriptions is informative to illustrate the physics behind individual model process representations. The processes reviewed are specific to those needed to formulate a fully distributed watershed runoff, erosion, and chemical transport and fate modeling framework. Major components of the framework are hydrology, sediment transport, and chemical transport and fate. Each of these major components can be viewed as submodels within the overall framework. The reviews that follow are grouped by submodel.

2.1 HYDROLOGIC PROCESSES

The main processes in the hydrologic submodel are: (1) precipitation and interception; (2) snowmelt; (3) infiltration and transmission loss; (4) storage; and (5) overland and channel flow.

2.1.1 Precipitation and Interception

The hydrologic cycle begins with precipitation. Precipitation includes both rainfall and snowfall. When surface air temperatures are near 0°C, precipitation is nearly always snowfall (Eagleson, 1970, Gray and Prowse 1993). Snowfall can be represented as an equivalent depth (or volume) of water and may be expressed as equivalent precipitation. The gross volume of water reaching the near surface is:

$$\frac{\partial V_g}{\partial t} = i_g A_s \tag{2.1}$$

where: V_g = gross precipitation water volume [L^3]

i_g = gross precipitation rate [L T^{-1}]

A_s = surface area over which precipitation occurs [L^3]

t = time [T]

Interception is the reduction in the volume of gross precipitation due to water retention by vegetative cover. As precipitation falls to the surface, a portion of the gross precipitation at the surface may contact vegetative canopy and may be held on foliage by surface tension (Eagleson, 1970). Much of the precipitation falling during the early period of an event may be stored on vegetative surfaces (Linsley et al. 1982). Intercepted water can return to the atmosphere by evaporation. Alternatively, intercepted water may reach the land surface when the force of gravity acting of water drops exceeds the surface tension force holding water to plant surfaces. Conceptually, interception may be represented as a volume. Net precipitation volume equals gross precipitation volume minus the volume lost to interception (Linsley et al. 1982):

$$V_i = (S_i + Et_R)A_s \qquad (2.2)$$

$$V_n = V_g - V_i \qquad for: V_g > V_i$$

$$\qquad (2.3)$$

$$V_n = 0 \qquad for: V_g \leq V_i$$

where: V_i =   interception volume [L^3]

         S_i =   interception capacity of projected canopy per unit area [$L^3 L^{-2}$]

         E =   evaporation rate [$L T^{-1}$]

         t_R =   precipitation event duration [T]

         V_n =   net precipitation volume reaching the surface [L^3]

Note that when the cumulative gross precipitation volume that occurs during an event is less than the interception volume, the net precipitation volume (or depth) reaching the land surface is zero. For single storm events, recovery of interception volume by evaporation can be neglected. Net precipitation volume may also be expressed as a net (effective) precipitation rate:

$$i_n = \frac{1}{A_s}\frac{\partial V_n}{\partial t} \qquad (2.4)$$

where: i_n =   net (effective) precipitation rate at the surface [$L T^{-1}$]

2.1.2 Snowpack and Snowmelt

Snow on the land surface can be quantified in terms of the water content of the snowpack and is expressed as snow-water equivalent (SWE) volume or depth (Dingman, 2002). Precipitation that falls as snow is accumulated in the snowpack. Snowmelt occurs as a function of ambient air temperature and incoming shortwave solar radiation. Shortwave solar radiation can be scattered by cloud cover, solar angle, topographic slope and other factors (Dingman, 2002; Liston and Elder, 2006). Incoming solar radiation can also be affected by tree canopy, buildings, and other obstructions that reduce the open view of the sky (Svensson and Eliasson, 2002; Eliasson and Svensson, 2003; Schnorbus and Alila, 2004). As modified to account for sky view, a snowmelt rate equation that combines temperature and solar radiation effects is (Pellicciotti et al. 2005):

$$M_S = \begin{cases} F_T(T_a - T_M) + F_{SR}F_{SV}(1-\alpha_s)G_{SR} & for \quad T_a > T_M \\ F_{SR}F_{SV}(1-\alpha_s)G_{SR} & for \quad T_a \leq T_M \end{cases} \qquad (2.5)$$

where: M_S =   snowmelt rate [$L T^{-1}$]

$$F_T \quad = \quad \text{temperature factor for snowmelt } [\text{L T}^{-1} \theta^{-1}]\ ^{[1]}$$

$$T_a \quad = \quad \text{ambient air temperature } [\theta]$$

$$T_M \quad = \quad \text{temperature at which snowmelt occurs } [\theta]$$

$$F_{SR} \quad = \quad \text{solar radiation factor for snowmelt } [\text{M}^{-1} \text{L T}^2 \theta^{-1}]$$

F_{SV} = sky view factor (proportion of hemisphere visible from a point [dimensionless]

α_s = albedo (light reflectance) of snow [dimensionless]

G_{SR} = incoming shortwave solar radiation [M T^{-3}]

2.1.3   *Infiltration and Transmission Loss*

Infiltration is the downward transport of water from the surface to the subsurface. The rate at which infiltration occurs may be affected by several factors including hydraulic conductivity, capillary action and gravity (percolation) as the soil matrix reaches saturation. Many relationships have been used to describe infiltration including expressions presented by Green and Ampt (1911), Richards (1931), Philip (1957), and Smith and Parlange (1978). The Green and Ampt relationship is often used because of its ease of application. This relationship assumes a sharp wetting front exists between the infiltration zone and soil at the initial water content (piston flow) and that the length of the wetted zone increases as infiltration progresses. Neglecting the depth of ponding at the surface (i.e. assuming that the pressure head is much smaller than the suction head), the general equation showing the Green and Ampt relationship can be expressed as (Li et al. 1976; Julien, 2002):

$$f = K_h \left(1 + \frac{H_c (1 - S_e)\theta_e}{F} \right) \tag{2.6}$$

where: f = infiltration rate [L T^{-1}]

K_h = effective hydraulic conductivity [L T^{-1}]

H_c = capillary pressure (suction) head at the wetting front [L]

θ_e = effective soil porosity = $(\varphi - \theta_r)$ [dimensionless]

φ = total soil porosity [dimensionless]

θ_r = residual soil moisture content [dimensionless]

S_e = effective soil saturation [dimensionless]

F = cumulative (total) infiltrated water depth [L]

----

[1] The symbol θ is used to represent the fundamental units of temperature in the LTMθ system (Dingman, 2002). θ indicates degree and θ^{-1} indicates degree^{-1}.

Similar to infiltration in overland areas, water in stream channels may be lost to the subsurface by transmission loss. The rate at which transmission loss occurs in a channel may be affected by several factors, particularly hydraulic conductivity. For ephemeral streams, capillary suction head may be significant when stream sediments are unsaturated. Relationships to describe the volume of transmission loss are presented by Lane (1983). Abdullrazzak and Morel-Seytoux (1983) and Freyberg (1983) use the Green and Ampt (1911) relationship to assess transmission loss. Following the form of the Green and Ampt relationship and accounting for the depth of (ponded) water in the stream channel (hydrostatic head), the transmission loss rate may be expressed as:

$$t_l = K_h \left(1 + \frac{(H_w + H_c)(1 - S_e)\theta_e}{T} \right) \tag{2.7}$$

where:

t_l	=	transmission loss rate [L T^{-1}]	
K_h	=	effective hydraulic conductivity [L T^{-1}]	
H_w	=	hydrostatic pressure head (depth of water in channel) [L]	
H_c	=	capillary pressure (suction) head at the wetting front [L]	
θ_e	=	effective sediment porosity = $(\varphi - \theta_r)$ [dimensionless]	
φ	=	total sediment porosity [dimensionless]	
θ_r	=	residual sediment moisture content [dimensionless]	
S_e	=	effective sediment saturation [dimensionless]	
T	=	cumulative (total) depth of water transported by transmission loss [L]	

During infiltration, water may not completely displace all air from soil void spaces. As a consequence of entrapped air, effective hydraulic conductivities are generally smaller than saturated hydraulic conductivities. Hydraulic conductivities may also be affected by surface crusting in bare soils and macropores in vegetated areas (Rawls et al. 1983; Rawls et al. 1993).

For single storm events, recovery of infiltration capacity by evapotranspiration and percolation can be neglected. Similarly, recovery of transmission loss capacity by evaporation or other processes can also be neglected for single storm events.

2.1.4 Storage

Water may be stored in depressions on the land surface as small, discontinuous surface pools. Precipitation retained in such small surface depressions is depression storage (Linsley et al. 1982). Water in depression storage may be conceptualized as a volume or, when normalized by surface area, a depth. In effect, the depression storage depth represents a threshold limiting the occurrence of overland flow. When the water depth is

below the depression storage threshold, overland flow is zero. Note that water in depression storage is still subject to infiltration and evaporation.

Similar to depression storage in overland areas, water in channels may be stored in depressions in the stream bed (as the channel water depth falls below some critical level, flow is zero and the water surface discontinuous but individual pools of water remain). This mechanism is termed dead storage. Note that water in dead storage is still subject to transmission loss and evaporation.

For single storm events, recovery of depression storage volume by evaporation can be neglected. Similarly, recovery of dead storage volume by evaporation can also be neglected for single storm events.

2.1.5   Overland and Channel Flow

Overland flow can occur when the water depth on the overland plane exceeds the depression storage threshold. Overland flow is governed by conservation of mass (continuity) and conservation of momentum. The two-dimensional (vertically integrated) continuity equation for gradually-varied flow over a plane in rectangular (x, y) coordinates is (Julien et al. 1995; Julien, 2002):

$$\frac{\partial h}{dt} + \frac{\partial q_x}{dx} + \frac{\partial q_y}{dy} = i_n - f + W = i_e \tag{2.8}$$

where: h =   surface water depth [L]

q_x , q_y =   unit discharge in the x- or y-direction = Q_x/B_x , Q_y/B_y [$L^2 T^{-1}$]

Q_x , Q_y =   flow in the x- or y-direction [$L^3 T^{-1}$]

B_x , B_y =   flow width in the x- or y-direction [L]

W =   unit discharge from/to a point source/sink [$L^2 T^{-1}$]

i_e =   excess precipitation rate [$L T^{-1}$]

Momentum equations for the x- and y-directions may be derived by relating the net forces per unit mass to flow acceleration (Julien et al. 1995; Julien, 2002). In full form, with all terms retained, these equations can be expressed in dimensionless form as the friction slope and are known as the Saint Venant equations. The full Saint Venant equations may be simplified by neglecting small terms that describe the local and convective acceleration components of momentum, resulting in the diffusive wave approximation (of the friction slope) for the x- and y-directions:

$$S_{fx} = S_{0x} - \frac{\partial h}{dx} \tag{2.9}$$

$$S_{fy} = S_{0y} - \frac{\partial h}{dy} \tag{2.10}$$

where: S_{fx}, S_{fy} = friction slope (energy grade line) in the x- or y-direction [dimensionless]

 S_{0x}, S_{0y} = ground surface slope in the x- or y-direction [dimensionless]

To solve the overland flow equations for continuity and momentum, five hydraulic variables must be defined in terms of a depth-discharge relationship to describe flow resistance. Assuming that flow is turbulent and resistance can be described using the Manning formulation (in S.I. units), the depth-discharge relationships are (Julien et al. 1995; Julien, 2002):

$$q_x = \alpha_x h^\beta \tag{2.11}$$

$$q_y = \alpha_y h^\beta \tag{2.12}$$

$$\alpha_x = \frac{S_{fx}^{1/2}}{n} \tag{2.13}$$

$$\alpha_y = \frac{S_{fy}^{1/2}}{n} \tag{2.14}$$

where: α_x, α_y = resistance coefficient for flow in the x- or y-direction [$L^{1/3} T^{-1}$]

 β = resistance exponent = 5/3 [dimensionless]

 n = Manning roughness coefficient [$T L^{-1/3}$]

Similarly, channel flow can occur when the water depth in the channel exceeds the dead storage threshold. Channel flow is also governed by conservation of mass (continuity) and conservation of momentum. At the watershed it is convenient to represent channel flows in a watershed as one-dimensional (along the channel in the down-gradient direction). The one-dimensional (laterally and vertically integrated) continuity equation for gradually-varied flow along a channel is (Julien et al. 1995; Julien, 2002):

$$\frac{\partial A_c}{dt} + \frac{\partial Q}{dx} = q_l \tag{2.15}$$

where: A_c = cross sectional area of flow [L^2]

 Q = total discharge [$L^3 T^{-1}$]

 q_l = lateral unit flow (into or out of the channel) [$L^2 T^{-1}$]

W = unit discharge from/to a point source/sink [$L^2 T^{-1}$]

Based on the momentum equation for the down-gradient direction and again neglecting terms for local and convective acceleration, the diffusive wave approximation may be used for the friction slope (see Eq. 2.9). To solve the channel flow equations for continuity and momentum, the Manning relationship may be used to describe flow resistance (Julien et al. 1995; Julien, 2002):

$$Q = \frac{1}{n} A_c R_h^{2/3} S_f^{1/2} \tag{2.16}$$

where: R_h = hydraulic radius of flow = A_c/P_c [L]

P_c = wetted perimeter of channel flow [L]

2.2   SEDIMENT TRANSPORT PROCESSES

The movement of water across the overland plane or through a channel network can transport and redistribute soil and sediment throughout a watershed. The main processes in the sediment transport submodel are: (1) advection-diffusion; (2) erosion; (3) deposition; and (4) bed processes (bed elevation response to erosion and deposition).

2.2.1   Advection-Diffusion

For the overland plane in two-dimensions (vertically integrated), the concentration of particles is governed by conservation of mass (sediment continuity) (Julien, 1998):

$$\frac{\partial C_s}{dt} + \frac{\partial \hat{q}_{tx}}{dx} + \frac{\partial \hat{q}_{ty}}{dy} = \hat{J}_e - \hat{J}_d + \hat{W}_s = \hat{J}_n \tag{2.17}$$

where: C_s = concentration of sediment particles in the flow [$M L^{-3}$]

$\hat{q}_{tx}$, $\hat{q}_{ty}$ = total sediment transport areal flux in the x- or y-direction [$M L^{-2} T^{-1}$]

$\hat{J}_e$ = sediment erosion volumetric flux [$M L^{-3} T^{-1}$]

$\hat{J}_d$ = sediment deposition volumetric flux [$M L^{-3} T^{-1}$]

$\hat{W}_s$ = sediment point source/sink volumetric flux [$M L^{-3} T^{-1}$]

$\hat{J}_n$ = net sediment transport volumetric flux [$M L^{-3} T^{-1}$]

The total sediment transport flux in any direction has three components, advective, dispersive (mixing), and diffusive, and may be expressed as (Julien, 1998):

$$\hat{q}_{tx} = v_x C_s - (R_x + D)\frac{\partial C_s}{dx} \tag{2.18}$$

$$\hat{q}_{ty} = v_y C_s - (R_y + D)\frac{\partial C_s}{dy} \tag{2.19}$$

where: v_x, v_y =   flow (advective) velocity in the x- or y-direction [L T^{-1}]

R_x, R_y =   dispersion (mixing) coefficient the x- or y-direction [L^2 T^{-1}]

D =   diffusion coefficient [L^2 T^{-1}]

$v_x C_s$ =   advective flux in the x-direction = J$_x$ [M L^{-2} T^{-1}]

$v_y C_s$ =   advective flux in the y-direction = J$_y$ [M L^{-2} T^{-1}]

$R_x \dfrac{\partial C_s}{dx}$ =   dispersive flux the x-direction [M L^{-2} T^{-1}]

$R_y \dfrac{\partial C_s}{dy}$ =   dispersive flux the y-direction [M L^{-2} T^{-1}]

$D \dfrac{\partial C_s}{dx}$ =   diffusive flux the x-direction [M L^{-2} T^{-1}]

$D \dfrac{\partial C_s}{dy}$ =   diffusive flux the y-direction [M L^{-2} T^{-1}]

The dispersive and diffusive flux terms in Eqs. (2.18) and (2.19) are negatively signed to indicate that mass is transported in the direction of decreasing concentration gradient. Note that both dispersion and diffusion are represented in forms that follow Fick's Law. However, dispersion represents a relatively rapid turbulent mixing process while diffusion represents a relatively slow a Brownian motion, random walk process (Holley, 1969). In turbulent flow, dispersive fluxes are typically several orders of magnitude larger than diffusive fluxes. Further, flow conditions for intense precipitation events are usually advectively dominated as dispersive fluxes are typically one to two orders smaller than advective fluxes. As a result, both the dispersive and diffusive terms may be neglected.

Similarly, for the channel plane in one-dimension (laterally and vertically integrated), the concentration of particles is governed by conservation of mass (sediment continuity) (Julien, 1998):

$$\frac{\partial C_s}{dt} + \frac{\partial \hat{q}_{tx}}{dx} = \hat{J}_e - \hat{J}_d + \hat{W}_s = \hat{J}_n \tag{2.20}$$

Individual terms for the channel advection-diffusion equation are identical to those defined for the overland plane. Similarly, the diffusive flux term can be neglected. The

dispersive flux is expected to be larger than the corresponding term for overland flow. However, the channel dispersive flux still may be negligible relative to the channel advective flux during intense runoff events.

2.2.2 Erosion

Erosion is the entrainment (gain) of material from a bottom boundary into a flow by the action of water. The erosion flux may be expressed as a mass rate of particle removal from the boundary over time and the concentration (bulk density) of particles at the boundary:

$$J_e = v_r C_{sb} \qquad (2.21)$$

where: J_e = erosion flux [M L^{-2} T^{-1}]

v_r = resuspension (erosion) velocity [L T^{-1}]

C_{sb} = concentration of sediment at the bottom boundary (in the bed) [M L^{-3}]

Entrained material may be transported as either bedload or suspended load. However, for overland sheet and rill flows, bedload transport by rolling and sliding may predominate as the occurrence of saltation and full suspension may be limited (Julien and Simons, 1985). Entrainment rates may be estimated from site-specific erosion rate studies or, in general, from the difference between sediment transport capacity and advective fluxes:

$$v_r = \begin{cases} \dfrac{J_c - v_a C_s}{\rho_b} & for \quad J_c > v_a C_s \\ \\ 0 & for \quad J_c \le v_a C_s \end{cases} \qquad (2.22)$$

where: v_r = resuspension (erosion) velocity [L T^{-1}]

J_c = sediment transport capacity areal flux [M L^{-2} T^{-1}]

v_a = advective (flow) velocity (in the x- or y-direction) [L T^{-1}]

C_s = concentration of sediment entrained in the flow [M L^{-3}]

ρ_b = bulk density of bed sediments [M L^{-3}]

In the overland plane, particles can be detached from the bulk soil matrix by raindrop (splash) impact and entrained into the flow by hydraulic action when the exerted shear stress exceeds the stress required to initiate particle motion (Julien and Frenette, 1985; Julien and Simons, 1985). The overland erosion process is influenced by many factors including precipitation (rainfall) intensity and duration, runoff length, surface slope, soil characteristics, vegetative cover, exerted shear stress, and particle size. Raindrop impact

may generally be neglected when flow depths are greater than three times the average raindrop diameter (Julien, 2002).

2.2.2.1 Soil Erosion Relationships

Extensive reviews of hillslope and watershed-scale soil erosion models are presented by Aksoy and Kavvas (2005) and Merritt et al. (2003). Soil erosion relationships range in complexity from simple empirical equations to physically-based models that are applicable over different spatial and temporal scales. Common soil erosion relationships include the Universal Soil Loss Equation (USLE) and its variants. The USLE (Wischemeier and Smith, 1978) is an empirical based on a large database of field plot measurements. It was developed to predict soil losses from agriculture and is designed to estimate long-term average annual soil loss associated with sheet and rill erosion using six factors that are associated with climate, soil, topography, vegetation and land use management:

$$A = R \, \hat{K} \, LS \, \hat{C} \, \hat{P} \qquad (2.23)$$

where: A = average annual soil loss due to sheet and rill erosion (tons/acre/year) $[M \, L^{-2} \, T^{-1}]$

R = rainfall erosivity factor [dimensionless]

$\hat{K}$ = soil erodibility factor (tons/acre) [dimensionless]

LS = slope length-gradient factor normalized to a field with a standard length of 23.2 meters (76.2 ft) and a slope of 9% [dimensionless]

$\hat{C}$ = cropping-management factor normalized to a tilled area that is continuously fallow [dimensionless]

$\hat{P}$ = conservation practice factor normalized to straight-row farming up and down the slope [dimensionless]

The Revised Universal Soil Loss Equation (RUSLE) and later versions of the RUSLE framework (Renard et al., 1991; Renard et al., 1994; Renard et al. 1997) have the same basic form as the original USLE but use extended methods to calculate how soil erosion factors are determined. In particular, a subfactor approach to determine crop management factors enables RUSLE to be applied to crops and management systems that were not examined in the original experiments used to develop the USLE. RUSLE is applicable to one-dimensional hillslopes that do not produce deposition as a result of changes in slope gradient. RUSLE2 (Foster et al., 2003) provides an approach that estimates erosion on a daily basis and accounts deposition resulting from slope gradient changes on hillslopes.

The Modified Universal Soil Loss Equation (MUSLE) (Williams, 1975) estimates soil erosion loss (yield) for a single storm event by replacing the rainfall erosivity factor with a runoff energy factor determined by flow:

$$Y_e = \alpha(Qq_p)^\beta \, \hat{K} \, LS \, \hat{C} \, \hat{P} \qquad (2.24)$$

where: Y_e =   sediment yield from an individual storm (tons/acre) [M L^{-2}]

Q_v =   storm runoff volume (m^3) [L^3]

q_p =   peak runoff rate (m^3/s) [L^3 T^{-1}]

α =   empirical soil erosion coefficient = 11.8

β =   empirical soil erosion exponent = 0.56 [dimensionless]

More detailed review of the USLE family of soil erosion relationships is presented by Kinnell (2010).

2.2.2.2   *Overland Sediment Transport Capacity Relationships*

Julien (1998, 2002) recommends a modified form of the Kilinc and Richardson (1973) relationship that includes soil erodibility, cover, and management practice terms from the Universal Soil Loss Equation (USLE) (Meyer and Weischmeier, 1969) to estimate the total overland sediment transport capacity (for both the x- and y-directions):

$$q_s = 1.542 \times 10^8 \, q^{2.035} \, S_f^{\,1.66} \hat{K}\hat{C}\hat{P} \qquad (2.25)$$

$$J_c = \frac{q_s}{B_e} \qquad (2.26)$$

where: q_s =   total sediment transport capacity (kg/m s) [M L^{-1} T^{-1}]

q =   unit flow rate of water = $v_a \, h$ [L^2 T^{-1}]

S_f =   friction slope [dimensionless]

$\hat{K}$ =   USLE soil erodibility factor [dimensionless]

$\hat{C}$ =   USLE soil cover factor [dimensionless]

$\hat{P}$ =   USLE soil management practice factor [dimensionless]

B_e =   width of eroding surface in flow direction [L]

Building on the initial work of Julien and Simons (1985), Prosser and Rustomji (2000) summarized relationships to describe the sediment transport capacity of overland flow. A generalized overland flow sediment transport capacity equation is:

$$q_s = k_T \, q^{\beta_s} \, S_f^{\,\gamma_s} \qquad (2.27)$$

where: q_s =   total sediment transport capacity [M L^{-1} T^{-1}]

k_T = empirically or theoretically derived coefficient for (overland) sediment transport capacity [M L^{-1} T^{-1}]

q = unit flow (discharge) of water [L^2 T^{-1}]

β_s = empirically or theoretically derived exponent for discharge [dimensionless]

S_f = friction slope (local energy gradient) [dimensionless]

γ_s = empirical or theoretically derived exponent for local energy gradient [dimensionless]

The sediment transport capacity coefficient (k_T) represents the combined influence that rainfall intensity, overland flow, and landscape and particle characteristics such as soil erodibility, infiltration, surface roughness, and vegetative cover have on erosion. Prosser and Rustomji (2000) extended the review of discharge (β_s) and local energy gradient (γ_s) exponent values presented by Julien and Simons (1985) and concluded that $1.0 \leq \beta_s \leq 1.8$ and $0.9 \leq \gamma_s \leq 1.8$ are generally applicable for use in soil erosion modeling.

2.2.2.3 *Channel Transport Capacity Relationships*

In channels, sediment particles can be entrained into the flow when the exerted shear stress exceeds the stress required to initiate particle motion. For non-cohesive particles, the channel erosion process is influenced by factors such as particle size, particle density and bed forms. For cohesive particles, the erosion process is significantly influenced by inter-particle forces (such as surface charges that hold grains together and form cohesive bonds) and consolidation. Total (bed material) load transport capacity relationships account for the both bedload and suspended load components of sediment transport. Yang (1996) and Julien (1998) provide summaries of numerous total load transport relationships. The Engelund and Hansen (1967) relationship is considered a reasonable estimator of the total load:

$$C_w = 0.05\left(\frac{G}{G-1}\right)\frac{v_a S_f}{[(G-1)g d_p]^{0.5}}\left[\frac{R_h S_f}{(G-1)d_p}\right]^{0.5} \tag{2.28}$$

$$J_c = \frac{v_a C_t}{A_c} \tag{2.29}$$

where: C_w = concentration of entrained sediment particles by weight at the transport capacity [dimensionless]

G = particle specific gravity [dimensionless]

v_a = advective (flow) velocity (in the down-gradient direction) [L T^{-1}]

S_f = friction slope [dimensionless]

R_h	=	hydraulic radius of flow [L]
g	=	gravitation acceleration [L T^{-2}]
d_p	=	particle diameter [L]
A_c	=	cross sectional area of flow [L^2]
C_t	=	concentration of entrained sediment particles at the transport capacity = $10^6 GC_w / [G + (1 - G)C_w]$ (g/m^3) [M L^{-3}]

2.2.2.4 Erosion Thresholds for Transport Capacity Relationships

One feature common to both the Kilinc and Richardson (1973) and Engelund and Hansen (1967) relationships is that the implicit threshold for incipient motion is zero. This means that the transport capacity of every particle type simulated will always be greater than zero, regardless of particle size or the exerted shear stress, as long as the unit flow or flow velocity and friction slope are non-zero. This can lead to inconsistent results when erosion rates are computed from sediment transport capacities. The inferred erosion rate will almost always be greater than zero because the difference between the transport capacity and advective flux will nearly always be greater than zero. Consequently, a non-zero erosion rate can be computed even when the exerted shear stress is far less than the incipient motion threshold for the material. To address this limitation, incipient motion thresholds were added to the modified Kilinc and Richardson (1973), generalized overland transport capacity, and Engelund and Hansen (1967) relationships:

$$q_s = 1.542 \times 10^8 (q - q_c)^{2.035} S_f^{1.66} \hat{K}\hat{C}\hat{P} \qquad (2.30)$$

$$q_s = k_T (q - q_c)^{\beta_s} S_f^{\gamma_s} \qquad (2.31)$$

$$C_w = 0.05 \left(\frac{G}{G-1}\right) \frac{(v_a - v_c)S_f}{[(G-1)gd_p]^{0.5}} \left[\frac{R_h S_f}{(G-1)d_p}\right]^{0.5} \qquad (2.32)$$

where:	q_c	=	critical unit flow for erosion = $v_c h$ [L^2 T^{-1}]
	v_c	=	critical velocity for erosion [L T^{-1}]
	h	=	surface water depth [L]

2.2.2.5 Total Shear Stress and Grain Shear Stress Partitioning for Erosion

The total shear stress exerted at the soil or sediment surface is computed as:

$$\tau_T = \rho g R_h S_f \qquad (2.33)$$

where:	τ_T	=	total shear stress (N/m^2) [M L^{-1} T^{-2}]

$$\rho \quad = \quad \text{gravitation acceleration [L T}^{-2}\text{]}$$

$$g \quad = \quad \text{gravitation acceleration [L T}^{-2}\text{]}$$

$$R_h \quad = \quad \text{hydraulic radius of flow [L]}$$

$$S_f \quad = \quad \text{friction slope [dimensionless]}$$

The effective shear stress acting on particles at the soil or sediment surface is generally less than the total shear stress exerted on the surface as a consequence of friction and energy losses to vegetation, surface irregularities such as bedforms and drag and has been extensively studied (Einstein and Banks, 1950; van Rijn, 1984a,b; Rauws, 1988; van Rijn, 1993; Prosser et al., 1995; Julien and Klaasen, 1998; Atkinson et al., 2000; van Rijn, 2007a,b,c; Julien, 2010; Al-Hamden et al., 2011; Al-Hamden et al., 2012; Al-Hamden et al., 2013). For overland flow, the relationship of Al-Hamden et al. (2012) is used to determine the ratio of grain friction to total friction as a function of bare soil:

$$\frac{f_g}{f_T} = 0.033 \, e^{(3.41 f_b)} \tag{2.34}$$

where: f_g / f_T = ratio of grain friction to total friction [dimensionless]

f_b = fraction of bare soil area to the total ground area [dimensionless]

For channel flow, Julien (2010) describes several simple relationships that can be used to approximate the ratio of grain friction and total total friction based on flow depth, the mean particle diameter comprising the aggregate sediment bed, and bedform geometry:

$$\frac{f_g}{f_T} \approx \frac{0.32 \left(\dfrac{d_{50}}{h} \right)^{1/3}}{0.32 \left(\dfrac{d_{50}}{h} \right)^{1/3} + \dfrac{\overline{\Delta}}{\overline{\Lambda}}} \tag{2.35}$$

$$\overline{\Delta} \approx 2.5 h^{0.7} d_{50}^{0.3} \tag{2.36}$$

$$\overline{\Lambda} \approx 6.5 h \tag{2.37}$$

where: d_{50} = mean particle diameter of bed sediments (m) [L]

h = flow depth (m) [L]

$\overline{\Delta}$ = average height of bedforms (m) [L]

$\overline{\Lambda}$ = average wavelength of bedforms (m) [L]

The grain shear stress is the computed from the total shear stress as:

$$\tau_g = \frac{f_g}{f_T}\tau_T \qquad (2.38)$$

where: τ_g =   grain shear stress (N/m^2) [M L^{-1} T^{-2}]

2.2.3   Deposition

Deposition is the sedimentation (loss) of material entrained in a flow to a bottom boundary by gravity. The deposition process is influenced by many factors including particle density, diameter, and shape, and fluid turbulence. The deposition flux may be expressed as a mass rate of particle removal from the water column over time and the concentration of sediment particles that are entrained in the flow:

$$J_d = v_{se}C_s \qquad (2.39)$$

where: J_d =   deposition flux [M L^{-2} T^{-1}]

           v_{se} =   effective settling (deposition) velocity [L T^{-1}]

           C_s =   concentration of sediment particles in the flow [M L^{-3}]

Coarse particles (>62 µm) are typically inorganic and non-cohesive and generally have large settling velocities under quiescent conditions. Numerous empirical relationships to describe the non-cohesive particle settling velocities are available. Summaries of relationships and settling velocities are presented by Yang (1996) and Julien (1998). For non-cohesive (fine sand) particles with diameters from 62 µm to 500 µm, the settling velocity can be computed as (Cheng, 1997):

$$v_s = \frac{\nu}{d_p}\left[\left(25 + 1.2d_*^{\,2}\right)^{0.5} - 5\right]^{1.5} \qquad (2.40)$$

$$d_* = d_p\left[\frac{(G-1)g}{\nu^2}\right]^{1/3} \qquad (2.41)$$

where: v_s =   quiescent settling velocity [L T^{-1}]

           ν =   kinematic viscosity of water [L^2 T^{-1}]

           d_* =   dimensionless particle diameter [dimensionless]

Medium particles (10 µm < d_p <62 µm) can vary in character. Inorganic particles may behave in a non-cohesive manner. In contrast, organic particles (potentially including particles with organic coatings) may behave in a cohesive manner. Fine particles (<10 µm) often behave in a cohesive manner. If behavior is largely non-cohesive, settling velocities may be estimated as described by Julien (1998). If the behavior is cohesive, flocculation may occur. Floc size and settling velocity depend on the conditions under

which the floc was formed (Burban et al. 1990; Krishnappan, 2000; Haralampides et al. 2003). When flocculation occurs, settling velocities of cohesive particles can be approximated by relationship of the form (Burban et al. 1990):

$$v_{sf} = a d_f{}^m \qquad (2.42)$$

where: v_{sf} = floc settling velocity (cm/s) [L T^{-1}]

a = experimentally determined constant = 8.4 x 10^{-3}

d_f = median floc diameter (μm) [L]

m = experimentally determined constant = 0.024

However, depending on fluid shear, particle surface charge, and other conditions, fine particles may not flocculate. Under conditions that limit floc formation, fine particles can have very small, near zero settling velocities.

As a result of turbulence and other factors, not all particles settling through a column of flowing water will necessarily reach the sediment-water interface or be incorporated into the sediment bed (Krone, 1962). Beuselinck et al. (1999) suggest that this process also occurs for the overland plane. As a result, effective settling velocities in flowing water can be much less than quiescent settling velocities. The effective settling velocity of a particle can be described as a reduction in the quiescent settling velocity by the probability of deposition (Krone 1962; Mehta et al. 1989):

$$v_{se} = P_{dep} v_s \qquad (2.43)$$

where: v_{se} = effective settling velocity [L T^{-1}]

v_s = quiescent settling velocity [L T^{-1}]

P_{dep} = probability of deposition [dimensionless]

The probability of deposition varies with shear stress near the sediment bed and particle size. As particle size decreases or shear stress increases, the probability of deposition decreases. For non-cohesive particles, the probability of deposition has been described as a function of bottom shear stress (Gessler, 1965; Gessler 1967; Gessler, 1971):

$$P_{dep} = P = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{Y} e^{-0.5x^2} dx \qquad (2.44)$$

$$Y = \frac{1}{\sigma}\left(\frac{\tau_{cd,n}}{\tau} - 1\right) \qquad (2.45)$$

where: P = probability integral for the Gaussian distribution

σ =   experimentally determined constant = 0.57

τ_0 =   bottom shear stress [M L^{-1} T^{-2}]

$\tau_{cd,n}$ =   critical shear stress for deposition of non-cohesive particles, defined as the shear stress at which 50% of particles deposit [M L^{-1} T^{-2}]

For coarse particles, the critical shear stress for deposition can be computed from a force balance following the method of van Rijn (1984a,b) as summarized by QEA (1999), with the particle diameter equal to the mean diameter for a range of particle sizes in a class (i.e. $d_p = d_{50}$).

For cohesive particles, the probability of deposition has also been described as a function of bottom shear stress (Partheniades, 1992):

$$P_{dep} = 1 - P = 1 - \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{Y} e^{-0.5x^2} dx \tag{2.46}$$

$$Y = \frac{1}{\sigma} ln\left[0.25\left(\frac{\tau_{cd,c}}{\tau} - 1 \right) e^{1.27\tau_{cd}} \right] \tag{2.47}$$

where: σ =   experimentally determined constant = 0.49

         τ_0 =   bottom shear stress [M L^{-1} T^{-2}]

         $\tau_{cd,c}$ =   critical shear stress for deposition of cohesive particles, defined as the shear stress at which 100% of particles deposit [M L^{-1} T^{-2}]

The probability integrals in Equations 2.46 and 2.47 can be approximated as (Abramowitz and Stegun, 1972):

$$P = 1 - F(Y)\left(0.4362X - 0.1202X^2 + 0.9373X^3\right) \qquad \text{for } Y > 0 \tag{2.48}$$

$$P = 1 - P(|Y|) \qquad\qquad\qquad\qquad\qquad\qquad \text{for } Y < 0$$

$$F(Y) = \frac{1}{\sqrt{2\pi}} e^{-0.5Y^2} \tag{2.49}$$

$$X = \left(1 + 0.3327Y\right)^{-1} \tag{2.50}$$

2.2.4 Soil and Sediment Bed Processes

In response to the difference between bedform transport, erosion, and deposition fluxes, the net addition (burial) or net loss (scour) of particles from the bed causes the bed surface elevation to increase or decrease. The rise or fall of the bed surface is governed by the sediment continuity (conservation of mass) equation, various forms of which are attributed to Exner (1925) (see Simons and Sentürk, 1992). Julien (1998) presents a derivation of the bed elevation continuity equation for an elemental control volume that includes vertical and lateral (x- and y-direction) transport terms. Neglecting bed consolidation and compaction processes, and assuming that only vertical mass transport processes (erosion and deposition) occur, the sediment continuity equation for the change in elevation of the soil or sediment bed surface may be expressed as:

$$\rho_b \frac{\partial z}{dt} = v_{se} C_s - v_r C_{sb} \tag{2.51}$$

where:
z = elevation of the soil surface or sediment bed [L]

ρ_b = bulk density of soil or bed sediments [M L^{-3}]

v_{se} = effective setting (deposition) velocity [L T^{-1}]

C_s = concentration of sediment particles in the water column [M L^{-3}]

v_r = resuspension (erosion) velocity [L T^{-1}]

C_{sb} = concentration of sediment particles in the soil or sediment bed [M L^{-3}]

2.3 CHEMICAL TRANSPORT AND FATE PROCESSES

The movement of water and sediment across the overland plane or through a channel network can also transport and redistribute chemicals throughout a watershed. On the land surface and in channel environments, chemical typically exist in three phases: 1) dissolved in water, 2) bound with dissolved organic compounds (DOC) or other binding ligands or complexation agents; and 3) particle-associated. The pathways that affect chemical movements and interactions in the environment depend on the phase in which the chemicals are present. The main processes in the chemical transport and fate submodel are: 1) chemical partitioning and phase distribution; 2) advection-diffusion; 3) erosion; 4) deposition; 5) infiltration and 6) and mass transfer and transformation processes (chemical reactions).

2.3.1 Chemical Partitioning and Phase Distribution

Many chemicals are hydrophobic and readily partition between dissolved, bound, and particle-associated (particulate) phases. Partitioning to bound and particulate phases is a function of chemical affinity for surfaces and ion exchange (ionic chemicals) or organic carbon (neutral chemicals) (Karichoff et al. 1979; Schwarzenbach et al. 1993; Chapra, 1997). The equilibrium distribution of chemicals between phases is described by the partition (distribution) coefficient, concentration and binding effectiveness of binding

agents, and the concentration of particles or organic carbon. Mechanistically, partitioning is a function of the equilibrium rate at which chemicals sorb (move out of the dissolved phase) and desorb (move back into the dissolved phase). If the rates at which chemicals partition are much faster than the rates of other mass transfer processes, local equilibrium is assumed to exist and the dissolved, bound and particulate phase chemical concentrations can be expressed in terms of the total chemical concentration (sum of phases) (Thomann and Mueller, 1987; Chapra, 1997).

Chemicals may partition to all particle types (sorbents) present in a solution. The equilibrium partition (distribution) coefficient to any particle is defined as (Thomann and Mueller, 1987):

$$\Pi_{p_n} = K_{p_n} = f_{oc_n} K_{oc} \tag{2.52}$$

where: Π_{p_n} = equilibrium partition (distribution) coefficient for particle "n" $[L^3 M^{-1}]$

K_{p_n} = equilibrium partition (distribution) coefficient for particle "n" $[L^3 M^{-1}]$

f_{oc_n} = fraction organic carbon of particle "n" [dimensionless]

K_{oc} = organic carbon normalized partition coefficient $[L^3 M^{-1}]$

For particulate phases in the water column, equilibrium partition coefficients vary with the concentration of suspended solids as a result of particle interactions. Particle-dependent partition coefficients are described as (Di Toro, 1985):

$$\Pi_{px_n} = \frac{\Pi_{p_n}}{1 + \sum_{n=1}^{N} m_n \Pi_{p_n}/v_x} = \frac{f_{oc} K_{oc}}{1 + \sum_{n=1}^{N} f_{oc_n} m_n K_{oc}/v_x} \tag{2.53}$$

where: Π_{px_n} = particle-dependent partition coefficient $[L^3 M^{-1}]$

n = particle index = 1, 2, 3, etc.

m_n = concentration of particle "n" $[M L^{-3}]$

v_x = particle interaction parameter [dimensionless]

For the bound phase, the equilibrium binding coefficient is defined as:

$$\Pi_b = D_e f_{oc_D} K_{oc} \tag{2.54}$$

where: Π_b = equilibrium binding coefficient $[L^3 M^{-1}]$

f_{oc_D} = fraction organic carbon of DOC [dimensionless]

D_e =   DOC-binding effectiveness coefficient [dimensionless]

Conceptually, dissolved organic compounds are composed entirely of organic carbon ($f_{oc_D} = 1$). Under those conditions, the equilibrium binding coefficient would equal the organic carbon partition coefficient. However, at least for neutral organic chemical binding in some surface waters (the Great Lakes), observed binding coefficients were up to 100 times smaller than K_{oc} (Eadie et al. 1990; Eadie et al. 1992). Also, in sediment observed binding coefficients were up to 10 times smaller than K_{oc} (Landrum et al. 1985; Landrum et al. 1987; Capel and Eisenreich, 1990). One explanation for decreased binding efficiency is photobleaching of DOC by ultraviolet (UV-B) radiation (Kashian et al. 2004).

The equilibrium partition coefficient can be used to describe the fraction of the total chemical that is associated with each phase as follows (Thomann and Mueller, 1987; Chapra, 1997):

$$f_d = \frac{1}{1 + D_{oc}\,\Gamma_b + \sum_{n=1}^{N} m_n\,\Gamma_{px_n}} \tag{2.55}$$

$$f_b = \frac{D_{oc}\,\Gamma_b}{1 + D_{oc}\,\Gamma_b + \sum_{n=1}^{N} m_n\,\Gamma_{px_n}} \tag{2.56}$$

$$f_{p_n} = \frac{m_n\,\Gamma_{px_n}}{1 + D_{oc}\,\Gamma_b + \sum_{n=1}^{N} m_n\,\Gamma_{px_n}} \tag{2.57}$$

$$f_d + f_b + \sum_{n=1}^{N} f_{p_n} = 1 \tag{2.58}$$

where: f_d =   fraction of the total chemical in the dissolved phase [dimensionless]

         f_b =   fraction of the total chemical in the DOC-bound phase [dimensionless]

         n =   particle index = 1, 2, 3, etc.

         f_{p_n} =   fraction of the total chemical in the particulate phase associated with particle "*n*" [dimensionless]

Equations 2.55-2.57 are presented for the water column. For the sediment bed, Γ_{p_n} is used in place of Γ_{px_n}.

Lu and Allen (2001) present extensive assessments of copper partitioning onto suspended particulate matter in river water. They performed a series of adsorption experiments and found that many factors may influence the partition coefficient including pH, total suspended solids concentration, total copper concentration, dissolved organic matter, particulate organic matter, hardness, and ionic strength. Their results suggest that adsorption to organic matter binding sites in aqueous and solid phases plays the biggest role in controlling the extent of copper partitioning. However, Lu and Allen (2001) found that the most significant environmental factors affecting the value of the partition coefficient were the total suspended solids (TSS) concentration and pH. Graphs showing variation of the copper partition coefficient as a function of key environmental conditions are presented in Figure 2-1. Holm et al. (2003) found that cadmium partitioning, like copper, was highly correlated with soil cation exchange capacity, which is largely determined by organic carbon and clay content. Also, cadmium partition coefficients were found to decrease by an order of magnitude as soil pH decreased from 6.7 to 5.3. Similar behavior is also expected for zinc because, like copper and cadmium, it is divalent. Sauvé et al. (2000, 2003) noted that distribution coefficients for cadmium,
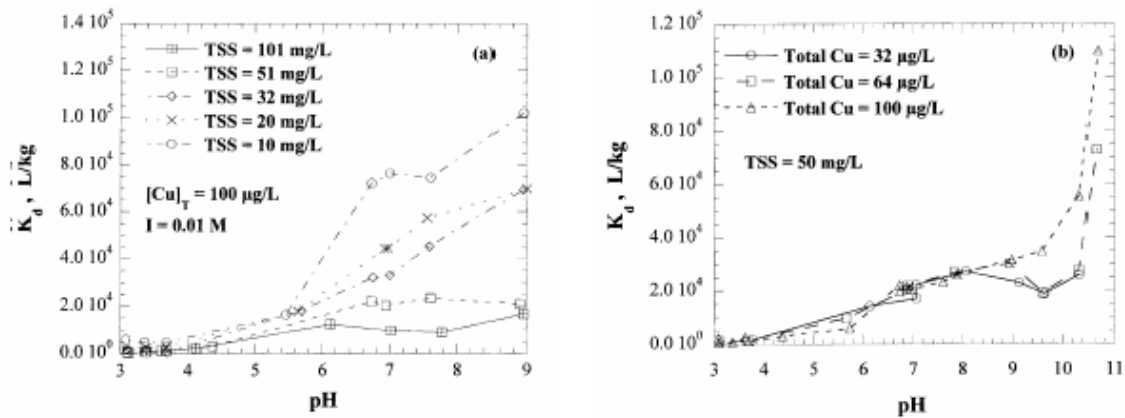
a) K_d as a function of pH and TSS concentration   b) K_d as a function of pH and copper concentration
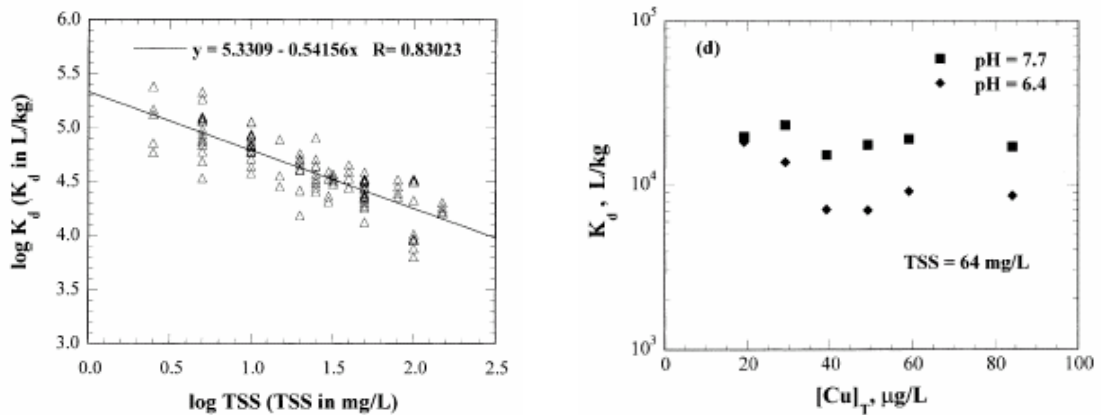
c) K_d as a function of TSS concentration   d) K_d as a function of copper concentration and pH

Figure 2-1. Copper partitioning vs. environmental conditions (Lu and Allen, 2001).

copper, and zinc and other divalent metals are sensitive to pH. Sauvé et al. (2003) reported distribution coefficients (log K_d) values for acidic (pH 4.4) soils were low: Cd log K_d = 3.05; Cu log K_d = 2.98; and Zn log K_d = 2.75. While pH may be the most important variable for partitioning, Sauvé et al. (2000, 2003) also noted the importance of organic matter as, after being normalized for pH, sorptive capacities for organic soils were reported to be up to 30 times larger than those observed for mineral soils.

2.3.2 Chemical Advection

Advection transports all chemical phases. For two-dimensional flow in the overland plane, a chemical continuity (conservation of mass) equation analogous the sediment continuity equation can be written as:

$$J_{xc} = v_x \left(f_d + f_b + \sum_{n=1}^{N} f_{p_n} \right) C_c = v_x C_c \qquad (2.59)$$

$$J_{yc} = v_y \left(f_d + f_b + \sum_{n=1}^{N} f_{p_n} \right) C_c = v_y C_c \qquad (2.60)$$

where: J_{xc} , J_{yc} =   chemical advective flux x- or y-direction [M L^{-2} T^{-1}]

v_x , v_y =   advective velocity in the x- or y-direction [L T^{-1}]

n =   particle index = 1, 2, 3, etc.

v_{r_n} =   resuspension (erosion) velocity of particle "*n*" [L T^{-1}]

f_d =   fraction of the total chemical in dissolved phase in the water column [dimensionless]

f_b =   fraction of the total chemical in the bound phase in the water column [dimensionless]

f_{p_n} =   fraction of the total chemical in particulate phase associated with particle "*n*" in the sediment column [dimensionless]

C_c =   total chemical concentration in the water column [M L^{-3}]

Similarly, for one-dimensional flow in channels a chemical continuity (conservation of mass) equation analogous the sediment continuity equation is identical to Equation 2.59.

2.3.3 Erosion and Deposition of Particulate Phase Chemicals

Chemicals associated with particles in the water column will enter the sediment bed if those particles settle. Similarly, chemicals associated with particles in the sediment bed will return to the water column if those particles are entrained (resuspend). The factors that control particle transport between the water column and sediment bed were described in Section 2.2.2. Since particle phase chemicals move with the particles transported, the

erosion and deposition fluxes of chemicals are described as (Thomann and Mueller, 1987):

$$J_{ec} = \sum_{n=1}^{N} v_{r_n} f_{p2_n} C_{c2} \qquad (2.61)$$

$$J_{dc} = \sum_{n=1}^{N} v_{se_n} f_{p1_n} C_{c1} \qquad (2.62)$$

where:
J_{ec}	=	chemical erosion flux [M L^{-2} T^{-1}]
J_{dc}	=	chemical deposition flux [M L^{-2} T^{-1}]
n	=	particle index = 1, 2, 3, etc.
v_{r_n}	=	resuspension (erosion) velocity of particle "n" [L T^{-1}]
v_{se_n}	=	effective settling velocity of particle "n" [L T^{-1}]
f_{p1_n}	=	fraction of the total chemical in particulate phase associated with particle "n" in the water column [dimensionless]
f_{p2_n}	=	fraction of the total chemical in particulate phase associated with particle "n" in the sediment column [dimensionless]
C_{c1}	=	total chemical concentration in the water column [M L^{-3}]
C_{c2}	=	total chemical concentration in the soil/sediment column [M L^{-3}]

2.3.4 *Chemical Infiltration and Subsurface Transport*

Chemicals associated with the dissolved and bound phase in the water column will enter the soil or sediment bed if the water transporting those chemicals infiltrates. When chemical partition coefficients are low and a significant fraction of the total chemical mass is in a mobile form, chemical infiltration may be significant. To account for this process, the chemical infiltration flux can be computed from the water infiltration flux as:

$$J_{ic} = v_i \left(f_{d1} + f_{b1} \right) C_{c1} = v_i f_{m1} C_{c1} \qquad (2.63)$$

where:
J_{ic}	=	chemical infiltration flux [M L^{-2} T^{-1}]
v_i	=	infiltration rate or transmission loss of water [L T^{-1}], previously defined as f in Eq. (2.5) or t_l in Eq. (2.6)
f_{d1}	=	fraction of the total chemical in dissolved phase in the water column [dimensionless]
f_{b1}	=	fraction of the total chemical in bound phase in the water column [dimensionless]
f_{m1}	=	fraction of the total chemical in the mobile phase in the water column [dimensionless] = $f_{d1} + f_{b1}$

$$C_{c1} \quad = \quad \text{total chemical concentration in the water column [M L}^{-3}\text{]}$$

Once in the subsurface, infiltrated chemicals would be subject to repartitioning with the chemical mass associated with porewater and particles in the soil column and transport via groundwater. The flow of groundwater through the soil also has the potential to leach chemicals from the soil column. Due to adsorption and the comparatively high bulk density of particles in the soil, subsurface chemical transport is subject to retardation (Fetter, 2001). Chemicals subject to retardation travel trough the subsurface at rates less than the average linear velocity of water. The retardation factor for a chemical in the subsurface is computed as (Fetter, 2001):

$$R = 1 + \frac{\rho_b}{\theta_e} K_p \tag{2.64}$$

where: R =   Retardation factor [dimensionless]

ρ_b =   soil bulk density [M L^{-3}]

θ_e =   effective soil porosity (void volume/total volume) [dimensionless]

K_p =   chemical equilibrium partition (distribution) coefficient [L^3 M^{-1}]

2.3.5 *Other Chemical Mass Transfer and Transformation Processes*

Beyond partitioning and mass transport processes, the fate of chemicals is potentially influenced by a number of other processes such as biodegradation, hydrolysis, oxidation, photolysis, and volatilization, and dissolution. However, for general simulation of elemental metals such as cadmium, copper, and zinc, volatilization, biodegradation, and photolysis do not occur. Hydrolysis and oxidation can affect the ionic speciation and phase distribution of metallic chemicals but do not affect the total chemical concentration. The effect that possible hydrolysis or oxidation reactions have on phase distributions of metals can be represented in terms of the chemical distribution (partitioning) coefficient.

3.0 TREX WATERSHED MODELING FRAMEWORK IMPLEMENTATION

3.1   GENERALIZED CONCEPTUAL MODEL FRAMEWORK

A generalized conceptual framework for the TREX watershed chemical transport and fate model is presented in Figure 3-1. At present, this framework research focuses on the event transport of metals in surface waters. Consequently, several possible processes in the general conceptual framework can be neglected because storm events are short-lived, lasting no more than a few hours. In particular, mass transfer and reactions processes such as volatilization, biodegradation, hydrolysis, and photodegradation can be neglected because of the short time scale for simulations or because these processes do not occur for metals. Other processes, such as dispersion and diffusion can also be neglected because at the time scale of event simulations transport processes are reasonable expected to be dominated by advection. At the event time scale, subsurface transport is also neglected. As a result, the transport and fate processes most important for the event simulation of metals are:

- Advective water column transport;

- chemical partitioning between water (truly dissolved), dissolved organic compounds (DOC) (or other binding agents) (bound), and solid (particulate) phases;

- Transport (erosion, deposition, net burial/unburial) of solids and particulate chemicals;

- Infiltration of dissolved and bound (mobile) phase chemicals;

- External sources and sinks of water, solids and chemicals.

Dynamic mass balance equations were developed based on the process descriptions presented in Section 2.0. In their most general form, these mass balance equations form a system of coupled partial differential equations that are functions of time and space. These equations describe the relationship between material inputs (precipitation or loads) and mass (water depth or constituent concentrations). To solve these equations, three simplifying assumptions were made and the equations expressed in finite difference form (Thomann and Mueller, 1987; Chapra, 1997):

1. Water column volumes are constant with respect to time during any interval ($\partial V/\partial t = 0$);

2. Surficial sediments do not move horizontally within the sediment bed; and

3. Chemical partitioning to solids and binding is rapid relative to other processes (local equilibrium).
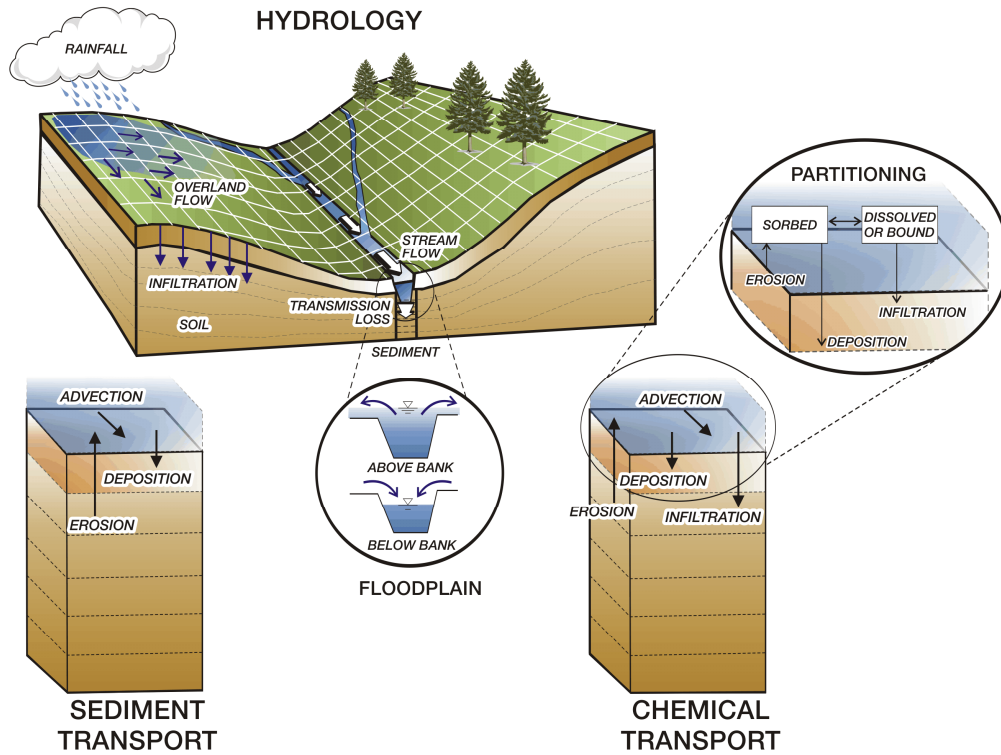
a) Overview of hydrology, sediment transport, and chemical transport and fate submodels
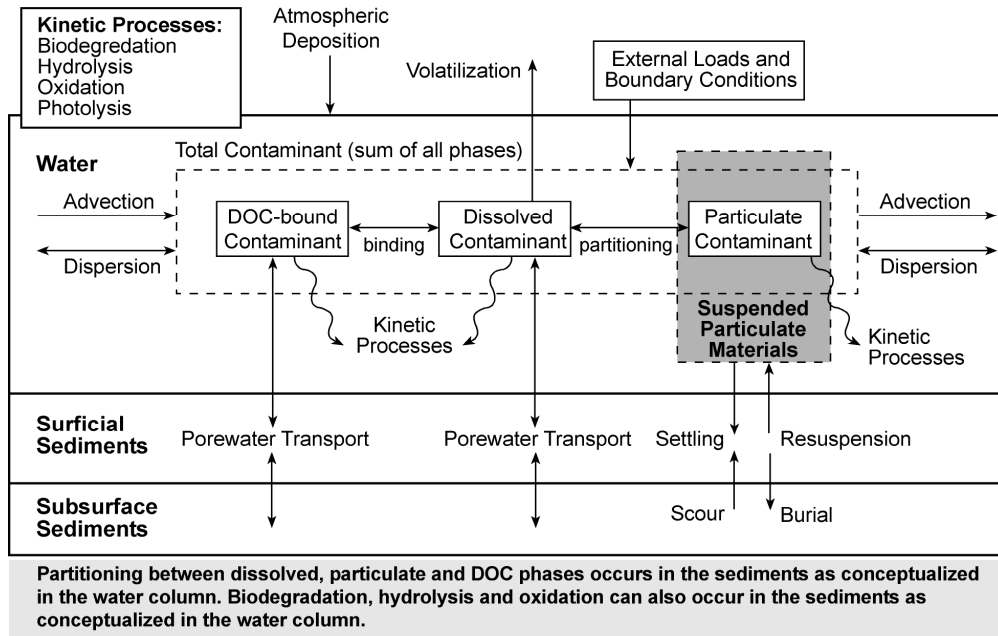
b) Details of chemical transport and fate processes

Figure 3-1. Generalized conceptual model framework.

The state variables in the model framework for the overland plane (denoted with the subscript "ov") and channel network (denoted with the subscript "ch") are: water depth (h), solids concentration (C_s), and chemical concentration (C_c). The corresponding equations for the water column and bed are:

Water Depth in the Overland Plane and Channel Network

$$\frac{\partial h_{ov}}{\partial t} = i_e - \frac{1}{B_x}\frac{\partial Q_{ovx}}{\partial x} - \frac{1}{B_y}\frac{\partial Q_{ovy}}{\partial y} - \frac{q_l}{L_c} + \frac{W_w}{A_c} \tag{3.1}$$

$$\frac{\partial(h_{ch}B_c)}{\partial t} = q_l - \frac{\partial Q_{ch}}{\partial x} + \frac{W_w}{L_c} \tag{3.2}$$

Solids in the Water Column of the Overland Plane and Channel Network

$$\frac{\partial C_{s,ov}}{\partial t} = v_r C_{sb,ov}\frac{A_s}{V_w} - v_{se}C_{s,ov}\frac{A_s}{V_w} - v_x C_{s,ov}\frac{A_c}{V_w} - v_y C_{s,ov}\frac{A_c}{V_w} - v_f C_{s,ov}\frac{A_c}{V_w} + \frac{W_s}{V_w} \tag{3.3}$$

$$\frac{\partial C_{s,ch}}{\partial t} = v_r C_{sb,ch}\frac{A_s}{V_w} - v_{se}C_{s,ch}\frac{A_s}{V_w} - v_x C_{s,ch}\frac{A_c}{V_w} + v_f C_{s,ov}\frac{A_c}{V_w} + \frac{W_s}{V_w} \tag{3.4}$$

Solids in the Soil and Sediment Bed

$$\frac{\partial C_{sb,ov}}{\partial t} = v_{se}C_{s,ov}\frac{A_s}{V_s} - v_r C_{sb,ov}\frac{A_s}{V_s} \tag{3.5}$$

$$\frac{\partial C_{sb,ch}}{\partial t} = v_{se}C_{s,ch}\frac{A_s}{V_s} - v_r C_{sb,ch}\frac{A_s}{V_s} \tag{3.6}$$

Total Chemical in the Water Column of the Overland Plane and Channel Network

$$\frac{\partial C_{c,ov}}{\partial t} = v_r C_{cb,ov}f_{pb}\frac{A_s}{V_w} - v_{se}C_{c,ov}f_p\frac{A_s}{V_w} - v_x C_{c,ov}\frac{A_c}{V_w} - v_y C_{c,ov}\frac{A_c}{V_w} - v_f C_{c,ov}\frac{A_c}{V_w}$$
$$- v_{i,ov}C_{c,ov}(f_d + f_b)\frac{A_s}{V_w} + \frac{W_c}{V_w} \tag{3.7}$$

$$\frac{\partial C_{c,ch}}{\partial t} = v_r C_{cb,ch} f_{pb} \frac{A_s}{V_w} - v_{se} C_{c,ch} f_p \frac{A_s}{V_w} - v_x C_{c,ch} \frac{A_c}{V_w} + v_f C_{c,ov} \frac{A_c}{V_w}$$
$$- v_{i,ch} C_{c,ch} \left(f_d + f_b \right) \frac{A_s}{V_w} + \frac{W_c}{V_w} \tag{3.8}$$

Total Chemical in the Soil and Sediment Bed

$$\frac{\partial C_{cb,ov}}{\partial t} = v_{se} C_{c,ov} f_p \frac{A_s}{V_s} - v_r C_{cb,ov} f_{pb} \frac{A_s}{V_s} \tag{3.9}$$

$$\frac{\partial C_{cb,ch}}{\partial t} = v_{se} C_{c,ch} f_p \frac{A_s}{V_s} - v_r C_{cb,ch} f_{pb} \frac{A_s}{V_s} \tag{3.10}$$

where: h =   flow depth of water column [L]

C_s , C_{sb} =   solids concentration in water column and bed [M L^{-3}]

C_c , C_{cb} =   total chemical concentration in water column and bed [M L^{-3}]

Q_x , Q_y =   flow in the x- or y-direction [L^3 T^{-1}]

q_l =   lateral unit flow from overland plane to channel (floodplain) [L^2 T^{-1}]

L_c =   length of channel in flow direction [L]

A_c , A_s =   cross sectional area in flow direction, bed surface area [L^2]

V_w , V_s =   volume of water and sediments [L^3]

v_x , v_y , v_f =   flow velocity in the x- or y-direction and between overland plane and channel (floodplain) [L T^{-1}]

v_r , v_{se} , v_i =   resuspension (erosion), effective settling (deposition), and infiltration (or transmission loss) velocities [L T^{-1}]

f_d , f_b , f_p =   dissolved, bound, and particulate chemical fractions [dimensionless]

$W_{w,s,c}$ =   material point source/sink: water [L^3 T^{-1}], solids, or chemical [M T^{-1}]

Each term in the mass balance equations represents a process in the conceptual framework. The variables in each term represent model parameters. Thomann and Mueller (1987) and Chapra (1997) provide more detailed presentations of mass balance equations for chemical transport.

3.2 GENERAL DESCRIPTION OF THE NUMERICAL FRAMEWORK

To simulate hydrologic, sediment, and chemical transport, values must be assigned to each model parameter and the mass balance equations defined by the conceptual model framework must be solved. Numerical integration techniques are needed to solve the model equations. TREX uses a finite difference control volume implementation of the generalized mass balance equation. To generate solutions, the framework computes dynamic mass balances for each state variable and accounts for all material that enters, accumulates within, or leaves a control volume through precipitation excess, external loads, advection, erosion, and deposition. TREX also features a "semi-Lagrangian" soil/sediment bed layer submodel to account for the vertical distribution of the physical and chemical properties of the overland soil and channel sediment columns (see Section 3.3). These equations are solved using Euler's method for numerical integration (Chapra and Canale, 1985):

$$s|_{t+\Delta t} = s|_t + \frac{\partial s}{\partial t}\bigg|_t \Delta t \qquad (3.11)$$

where: $s|_{t+\Delta t}$ = value of model state variable at time $t+\Delta t$ [L] or [M L^{-3}]

$s|_t$ = value of model state variable at time t [L] or [M L^{-3}]

$\frac{\partial s}{\partial t}\bigg|_t$ = value of model state variable derivative at time t [L T^{-1}] or [M L^{-3} T^{-1}]

Δt = time step for numerical integration [T]

3.3 TREX FRAMEWORK FEATURES

The initial basis for development of TREX was CASC2D. As part of the model development process, CASC2D's underlying hydrologic and sediment transport submodels were significantly enhanced before chemical transport and fate components were added to create the TREX framework. An overview of TREX features is presented in Table 3-1. As part of the overall development effort, the entire body of TREX source code is organized to significantly improve code structure and modularity and to provide complete, line-by-line documentation. Beyond allowing development of chemical transport and fate features to proceed, the TREX code is structured so that future categories of model features can be added to the framework without having to reconstruct the basic code. As presented in Figure 3-2, the code is designed so that the calculations for each process at any time level are independent and information is carried forward from hydrology to sediment transport to chemical transport in order to generate a solution. At any time level, flow is assumed to be unaffected by sediment and chemical transport and sediment transport is affected by chemical transport, so calculations for these three components have a natural hierarchy.
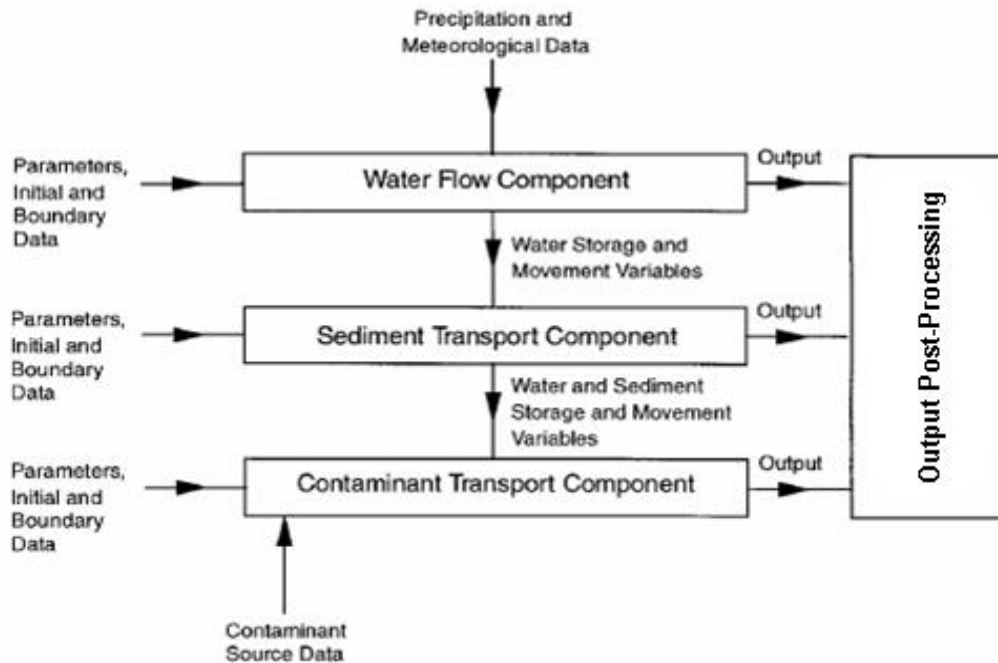
Figure 3-2. TREX Hierarchy and information flow (after Ewen et al 2000).

Within TREX, many features of the original CASC2D code were significantly enhanced and many new features were added. In particular, the TREX code is designed to simulate multiple watershed outlets and to also allow channel network branching in the upstream and downstream directions. This permits simulation of braided tributaries and distributary flows that might occur around alluvial fans or where a river system meets a large receiving waterbody on a low slope. Another significant enhancement is the addition of flow point sources and sinks. Note that TREX does not consider groundwater flow processes other than downward flow at the surface by infiltration or channel transmission loss and through defined soil and sediment layers. However, to account for other water losses or gains, groundwater interactions could be represented as a series of time-variable point sources and sinks. In effect, this feature allows TREX to be externally coupled with groundwater flow and transport modeling tools such as MODFLOW (Harbaugh et al. 2000), HST3D (Kipp, 1997), and MT3DMS (Zheng and Wang, 1999).

Another key feature of the enhanced TREX framework is the representation of the bed and bed processes. The bed itself is presented as a vertical stack (layers). Typical water quality modeling approaches use an Eulerian (fixed) frame of reference for all compartments. In contrast, TREX uses what is termed a "semi-Lagrangian" (floating) frame of reference (Velleux et al. 2001). In the Eulerian approach, the control volume for mass balance calculations is fixed in space and material is advected through the control volume. With respect to the bed, the deposition or erosion of material causes the entire

Table 3-1. Comparative overview of TREX features.

Model Component	Prior CASC2D Code	TREX Code	Status
General Model Controls			
Numerical integration time step	One time step limited to critical value regardless of flow	Series of time step values that can be optimized based on flow	Tested
Hydrologic Submodel			
Water depth initial condition	Assumed to be zero but recent code allowed a non-zero flow depth in channels (dry start)	User can specify any value for depth overland, in channels, or infiltrated (wet or dry start)	Tested
Flow outlets and downstream boundary conditions	Limited to one outlet, assumed normal depth	Any number of outlets possible, downstream control or normal depth can be specified	Partially tested
Floodplain interaction	Nascent features in initial code (Julien et al. 1995) but not in recent code (Rojas, 2002).	Restored feature and enhanced to compute flooding from water surface elevations	Tested
Channel topology: orientation	Channel connections limited to four (N-S or E-W) directions	Channel connections in all eight raster grid directions	Tested
Channel topology: branching	Converging branches, limited to two branches upstream	Converging and diverging branches with 2-7 branches	Tested
Flow point sources and sinks	None	Point sources for overland plane and channel network. Sources tested, sinks not.	Partially tested
Sediment Transport Submodel			
Number of particle types	Limited to three: sand, silt, clay	Unlimited number of types	Tested
Floodplain sediment transport	None: solids passing through overland part of a floodplain cell instantly move to channel	Occurs whenever water depth in overland part of floodplain cell exceeds zero	Tested
Channel erosion	Limited: only solids deposited during simulation erode; net bed elevation change never < 0	Not restricted; channels can incise and net change in bed elevation can be < 0	Tested
Solids point sources and sinks	None	Point sources for overland plane and channel network	Partially tested
Chemical Transport Submodel			
Number of chemical types	None	Unlimited	Tested
Chemical transport and fate	None	Three-phase partitioning with advection, erosion, deposition	Tested
Chemical point sources and sinks	None	Point sources for overland plane and channel network	Partially tested

frame of reference (all layers in the stack) to advect (upward or downward). As control volumes relocate, material is advected between adjacent layers. This advection can affect contaminant distributions in the bed. In the semi-Lagrangian approach, the control volume of the surface bed layer is allowed to expand or contract in response to erosion or deposition. Because the entire frame of reference for all bed layers is not relocated, the mixing that occurs with the Eulerian approach is eliminated. Velleux et al. (2001) and Imhoff et al. (2003) present further descriptions of the semi-Lagrangian approach and its details.

3.4 COMPUTATIONAL CONSIDERATIONS

The TREX source code is written in C and conforms to ANSI C99 conventions. The code was compiled and simulations executed on several computing systems to ensure a degree of portability. It is worth noting that TREX is a computationally intensive application. Simulations with hydrology, six solids types, and three chemical types on a domain with 34,000 elements required approximately 3.6 CPU hours on a system with an Intel Core i7-6700K 4 GHz quad-core processor with 32 GB of DDR4/2666 RAM with a 2TB (magnetic) hard disk drive running under the Linux operating system and with an executable created using the GNU (gcc) C compiler.

3.5 PROGRAM OPERATION

TREX is operated from a command line interface (the command prompt under the Windows operating system or shell prompt in Linux). In its most basic mode of operation, TREX requires that the user specify one argument. This argument is the path and file name of the TREX main input file. The main input file provides basic model input parameters that control a simulation and also contains names of ancillary model input files that delineate specific characteristics of the simulation such as the watershed boundary mask, elevations, soil classes, and land use, etc. Descriptions of the main and ancillary input files are presented in Section 4.0. When run from the command prompt under the Windows operating system, the command stream to begin execution of a TREX simulation is of the form:

C:\trex.exe *inputfilename.inp*

Similarly, when run from a shell prompt under the Linux operating system, the command stream is on the form:

/home/username/path/to/input/trex.x *inputfilename.inp*

TREX also has some automated warm/hot start capablities ("restart" options) that allow users to simulate a sequence of storm events where final conditions from one event are used as initial conditions for the next and also simulation recovery capabilities that allow a partially completed simulation to be resumed. To operate in a mode that permits a warm or hot start or a simulation recovery, TREX requires that the user specify additional arguments to identify the options to be used. These arguments are in addition to the main

input file. When run from the command prompt under the Windows operating system, the command stream to begin execution of a TREX simulation with a restart option selected is of the form:

C:\trex.exe *inputfilename.inp restart# recover ##.##*

C:\trex.exe *inputfilename.inp recover ##.## restart#*

Similarly, when run from a shell prompt under the Linux operating system, the command stream is on the form:

/home/username/path/to/input/trex.x *inputfilename.inp restart# recover*

/home/username/path/to/input/trex.x *inputfilename.inp recover ##.## restart#*

where: *restart#* is one of the values needed to invoke restart capabilities. The full list of permissible restart options is summarized in Table 3-2. Simulation recovery options are described in Table 3-3. **NOTE: TREX simulation recovery options are still under development and cannot be invoked at this time.**

During execution, TREX generates a series of output files. Depending on the number of cells in the spatial domain and number of state variables simulated and the frequency of reporting, the size of model output can be quite large (>5 GB). Descriptions of TREX output file types are presented in Section 5.0.

Table 3-2. TREX simulation restart option command line arguments.

Argument	*Description*
none	No restart files are read or written (default option when no argument is specified)
restart0	No initial conditions are read, but final conditions are written to restart files
restart1	Initial conditions for a "warm" start read from and final conditions written to restart files
restart2	Initial conditions for a "hot" start read from and final conditions written to restart files

Notes

1. For a warm start, initial conditions for soils and sediments (solids and chemical concentrations, number of layers, volume, surface area, thickness, etc.) are read from restart files.

2. For a hot start, initial conditions for soils, sediments, runoff, and surface water (solids and chemical concentrations, number of layers, volume, surface area, thickness, etc.) are read from restart files.

The warm start option is useful for simulating a sequence of events when the time between storms is so long that runoff depths on the overland plane are zero and surface water in streams returns to baseflow conditions. The hot start option is useful for simulating a sequence of events when the time between storms is short enough such that runoff depths are not zero and/or surface water conditions in have not yet returned to baseflow conditions.

It should be noted that neither of these restart options provide full simulation recovery capabilities because conditions for other processes (i.e. interception, infiltrations, transmission loss, etc.) are not written to or read from these restart files.

Use of any restart option (0, 1, or 2) requires that the user provide a "restart-info.txt" file in the "Restart" directory. **Note: The file name, directory name, and path are hardwired in the code.** This restart file contents and directory structure are decribed in Section 4.5.9.

Table 3-3. TREX simulation recovery option command line arguments.

Argument	*Description*
none	No simulation recovery occurs (default option when no argument is specified)
recover	the model runstate of a partially completed simulation

Notes

1. For a warm start, initial conditions for soils and sediments (solids and chemical concentrations, number of layers, volume, surface area, thickness, etc.) are read from restart files.

2. For a hot start, initial conditions for soils, sediments, runoff, and surface water (solids and chemical concentrations, number of layers, volume, surface area, thickness, etc.) are read from restart files.

The warm start option is useful for simulating a sequence of events when the time between storms is so long that runoff depths on the overland plane are zero and surface water in streams returns to baseflow conditions. The hot start option is useful for simulating a sequence of events when the time between storms is short enough such that runoff depths are not zero and/or surface water conditions in have not yet returned to baseflow conditions.

Use of any restart option (0, 1, or 2) requires that the user provide a "restart-info.txt" file in the "Restart" directory. **Note: The file name, directory name, and path are hardwired in the code.** This restart file contents and directory structure are decribed in Section 4.5.9.

4.0 DESCRIPTION AND ORGANIZATION OF MODEL INPUT FILES

4.1 INPUT FILE STRUCTURE OVERVIEW

TREX has a main input file that controls most aspects a simulation. Within this main input file, the inputs are divided into six groupings of related parameters (Data Groups A-F). The main input file also specifies a number of ancillary input files that are required to operate the model. The ancillary model input files are used to delineate specific characteristics of the simulation such as the watershed boundary mask, elevations, soil classes, and land use, etc. Ancillary files are each organized into Data Groups.

The organization and content of each Data Group is described in a series of tables presented in the following sections of this manual. Each Data Group is itself divided into records and fields. In general, the name of each variable, its type, and expected units is described. Variables names starting with "n" describe the number of elements associated with a parameter (i.e. nsolids = number of solids types, nchems = number of chemicals, ndt = number of time steps, etc.) Variables ending with "opt" are switches that toggle operation of model processes. Variables containing "ic", "bc", and "w" are associated with initial conditions (ic), boundary conditions (bc), and loads/forcing functions (w). Variable types include int (integer), float (floating point), char (an unbroken sequence of characters without a space or tab), and string (a sequence of characters that can include spaces and tabs). Inputs are typically specified in metric units (m, m/s, g/m^3, etc.).

Model controls for time steps (dt), printout, initial conditions (ICs), boundary conditions (BCs), and loads/forcing functions are input as paired values in a time series (i.e. pairs of {function value at time t, time t}). Time steps and print intervals are step functions (i.e. the input value is used until time t, after which the next value is used). ICs, BCs, and loads are piecewise linear functions (i.e. values are linearly interpolated between times specified).

4.2 MAIN MODEL INPUT FILE

Within this main input file, the inputs are divided into six groupings (Data Groups) of related parameters. Data Group A is used to specify general controls for the simulation such as the simulation type and the series of time steps to be used for numerical integration. Data Group B is used to specify parameters for hydrologic simulations. Data Group C is used to specify parameters for sediment transport simulations. Data Group D is used to specify parameters for chemical transport simulations. Data Group E is used to specify parameters for environmental conditions such as air temperature and wind speed. Data Group F is used to specify parameters for model output control. However, users should note that not all possible combinations of model inputs are fully implemented in the model at this time. Users are advised to review the TREX source code.

4.2.1   Data Group A: General Controls

Data Group A: General Controls

Record	Description
1	Header1 (string)
2	Header2 (string)
3	"KSIM" (char), ksim (int) {1 = hydrology, 2 = sediment, 3 = chemical}, "NROWS" (char), nrows (int), "NCOLS" (char), ncols (int), "DX" (char), dx (float) (m), "DY" (char), dy (float) (m), "TZERO" (char), tzero (double) (days) {used for environmental functions such as solar radiation}, "TSTART" (char), tstart (double) (hrs) {time beyond tzero}
Note	tzero is the decimal value of the Julian day at time zero (when the elapsed time is zero) and tstart is the time along the specified time functions where the simulation starts
4	"DTOPT" (char), dtopt (int) {0 = time steps input by user, 1 = time steps interatively calculated by model without simulation relaunch, 2 = time steps interatively calculated by model with simulation relaunch, 3 = time steps read from external file}
Note	The timestop option (dtopt) controls how numerical integration occurs. TREX solves its govering equations using a first-order, explicit solver (Euler's Method). When dtopt = 0, the user must specify the sequence (time series pairs) of time step (dt) values. If a time step is too large, numerical instability occurs and simulations will abort. When dtopt = 1, the model interatively calculates a sequence of time steps using Courant numbers determined in overland and channel flow routing routines. At any time level, simulations proceed using the time step for the last successful time iteration. If the time step is too large and the maximum Courant number is exceeded, calculations are repeated using a smaller time step until Courant number conditions are satisfied. If the time step is too small and Courant numbers fall below the minimum Courant number, calculations are repeated using a larger time step until Courant number conditions are satisfied. The final sequence of calculated time steps is written to file for later use. <u>Simulation relaunch does not occur for dtopt = 1 regardless of the ksim value input (i.e. ksim is overridden)</u>. When dtopt = 2, the model calculates a sequence of time steps using Courant conditions in the same manner as for dtopt = 1. <u>Simulation relaunch occurs if ksim > 1 (and dtopt = 2)</u>. For the case when ksim = 1 and dtopt = 2, the model will run the simulation, calculate time steps and end as specified. For the case when ksim > 1 and dtopt = 2, the model will override ksim, temporarily set ksim = 1, run the simulation and determine time steps while performing hydrology calculations only, write time steps to file, reset ksim to its initial value and then relaunch the full simulation (ksim > 1) using the time steps written to file.
Caution	Automated time-stepping can reduce effort to develop a full model simulation by reducing the need to manually repeat simulations to determine a sequence of time steps. However, run times for individual simulations will increase because calculations for any time level will be repeated as time steps are adjusted and written to file. *Once a successful time step sequence for a simulation is generated, users are advised to select dtopt = 3 and read those time step values from a file to minimize model run time impacts.*
	if dtopt = 0
5	"NDT" (char), ndt (int) {number of time step values}

Data Group A: General Controls (continued)

	for idt = 1, ndt
6	dt[idt] (float) (s), dttime[idt] (float) (hrs)
Note	Record 6 is repeated for idt = 1, ndt
	elseif dtopt = 1 or dtopt = 2
7	"DTMAX" (char), dtmax (float) (s), "MAXCOURANT" (char), maxcourant (float) (dimensionless), "RELAXATION" (char), relaxation (float) (dimensionless), "ENDTIME" (char), dttime[ndt] (float) (hours) {simulation end time in hours}
8	DTOUTPUTFILE (char), dtoutputfile (string)
	elseif dtopt = 3
9	"DTINPUTFILE" (char), dtinputfile (string)
	ReadDTFile
	endif dtopt = 0, 1 or 2, 3
10	"NPRINTOUT" (char), nprintout (int) {number of print intervals for tabular output}
	for iprntout = 1, nprintout
11	printout[iprntout] (float) (hrs), printouttime[iprntout] (float) (hrs)
Note	Record 11 is repeated for iprntout = 1, nprintout
12	"NPRINTGRID" (char), nprintgrid (int) {number of print intervals for grids output}, "GRIDSTART" (char), gridcount {starting value for grid count, extension for grid files}
	for iprntgrid = 1, nprintgrid
13	printgrid[iprntgrid] (float) (hrs), printgridtime[iprntgrid] (float) (hrs)
Note	Record 13 is repeated for iprntgrid = 1, nprintgrid
14	"NPRINTRECOVER" (char), nprintrecover (int) {number of print intervals for run state recovery data}
	for iprntrecover = 1, nprintrecover
15	printrecover[iprntrecover] (float) (hrs), printrsttime[iprntrecover] (float) (hrs)
Note	Record 13 is repeated for iprntrecover = 1, nprintrecover
16	"ECHO" (char), echofile (string) {required}

4.2.2   Data Group B: Hydrologic Simulation Parameters

Data Group B: Hydrologic Simulation Parameters

Record	Description
Record	*Description*
1	Header (string)
2	"MASK" (char), maskfile (string)
	call ReadMaskFile
3	"ELEVATION" (char), elevationfile (string) (m)
	call ReadElevationFile
4	"INFOPT" (char), infopt (int) (0 = no infiltration, 1 = infiltration)
	if ksim = 1 {for ksim > 1, see Data Group C}
	if infopt = 1 {there is infiltration but no sediment transport}
5	"NSOILS" (char), nsoils (int) (number of soil types)
	for isoil = 1, nsoils
6	kh[isoil] (float) (m/s), capsh[isoil] (float) (m), soilmd[isoil] (float) (dimensionless), soilname[isoil] (string)
Note	Record 6 is repeated for isoil = 1, nsoils
7	"SOIL_TYPES" (char), soilfile (string)
	call ReadSoilFile
	endif infiltopt
8	"NLANDS" (char), nlands (int) (number of land use types)
	for iland = 1, nlands
9	nmanningov[iland] (float) (n units), interceptionclass[iland] (float) (mm), landname[iland] (string)
Note	Record 9 is repeated for iland = 1, nlands
10	"LAND_USE" (char), landusefile (string)
	call ReadLandUseFile
	endif ksim = 1
11	"STORAGE_DEPTHS" (char), storagedepthfile (string) (m)
	call ReadStorageDepthFile
12	"CHNOPT" (char), chnopt (int) (0 = no channels, 1 = channels)

Data Group B: Hydrologic Simulation Parameters (continued)

	if chnopt = 1
13	"TPLGYOPT" (char), tplgyopt (int) {0 = compute topology from channel property file and link, and node masks, 1 = topology read from topology file}, "CTLOPT" (char), ctlopt (int) (0 = no transmission loss, 1 = transmission loss) {channel transmission loss option}, "FLDOPT" (char), fldopt (int) {0 = floodplain water transfer is from overland to channel only, 1= floodplain water transfer can be in either direction depending on water surface elevations}, "OUTOPT" (char), outopt (int) {0 = pour water from overland to channel portion of cell before routing overland at outlets, 1 = route water overland before pouring into channel at outlets}
	if tplgyopt = 0 then
14	"LINK" (char), linkfile (string)
	call ReadLinkFile
15	"NODE" (char), nodefile (string)
	call ReadNodeFile
16	"CHANNEL" (char), chanfile (string) {includes channel "dead" storage}
	call ReadChannelFile
	call ComputeTopology
Note	Records 14, 15, and 16 are only input if tplgyopt = 0.
	elseif tplgyopt = 1
17	"TOPOLOGY" (char), topofile (string) {combines the channel property, and the link and node masks} {also includes channel "dead" storage}
	call ReadTopologyFile {for future use...}
Note	Record 17 is only input if tplgyopt = 1.
Warning	Option not fully implemented
	endif tplgyopt
	if ksim = 1 {for ksim > 1, see Data Group C}
	if ctlopt = 1 {there is channel transmission loss but no sediment transport}
18	"TRANSMISSION_LOSS_PROPERTIES" (char), channeltlossfile (string)
	call ReadTransmissionLossProperties {read none-by-node transmission loss parameter values}
	endif ctlopt = 1
	endif ksim = 1
	endif chnopt

Data Group B: Hydrologic Simulation Parameters (continued)

19 (ICov)	"INITIAL_WATER_OVERLAND" (char), initialwaterovfile (string) (m)
	call ReadInitialWaterOverlandFile
	if infopt = 1
20 (ICsoil)	"INITIAL_INFILTRATION" (char), initialinfiltrationfile (string) (m)
	call ReadInitialInfiltrationFile
	endif infopt = 1
	if (chnopt = 1)
21 (ICch)	"INITIAL_WATER_IN_CHANNELS" (char), initialwaterchfile (string) (m)
	call ReadInitialWaterChannelFile
	if(ctlopt > 0)
22 (ICsed)	"INITIAL_TRANSMISSION_LOSS" (char), initialtranslossfile (string) (m)
	call ReadInitalTransmissionLossFile
	endif ctlopt
	endif chnopt
23 (W)	"RAINOPT" (char), rainopt (int)
Note	rainopt: 0 = uniform in space, 1 = IDW spatial interpolation, 2 = design storm constant in space from grid index map, 3 = radar data: locations and radar-derived rainrates from files, 4 = design storm basin-average (DAD or SST) input via file and distributed via elliptical pattern, 5 = time series grid read.
	if rainopt = 1
24	"IDWradius" (char), idwradius (float) (m), "IDWexponent" (char), idwexponent (float) (dimensionless)
	endif rainopt =1
	if rainopt <= 2
25	"NRAINGAGES" (char), nrg (int) {by default rain is uniform if only one rain gage is specified and distributed if there is more than one gage}
Check	if rainopt = 0 and nrg > 1, warn and abort... {if rainopt is zero, then nrg must equal 1}
	endif rainopt <= 2
	if rainopt > 2 or nrg > 0
26	"CONV1" (char), rainconvunits (float), "CONV2" (char), rainconvtime (float), "SCALE" (char), rainscale (float) {Note: these scalars are globals as needed for rainopt = 5}
	endif rainopt > 2 or nrg > 0

Data Group B: Hydrologic Simulation Parameters (continued)

	if rainopt <=2
	if nrg > 0
	for irg = 1, nrg
27	"GAGE" (char), rgid[irg] (int), rgx[irg] (float) (m), rgy[irg] (float) (m), nrpairs[irg] (int)
	for irpairs = 1, nrpairs[irg]
28	rfintensity[irg][ipairs] (float) (m/s), rftime[irg][ipairs] (float) (hrs)
Note	Records 27 and 28 are repeated as a group for irg = 1, nrg. Record 26 is repeated for ipairs = 1, npairs[irg].
	endif nrg > 0
	if rainopt = 2
29	"DESIGN_RAIN_GRID" (char), designraingridfile (string)
	call ReadDesignRainGrid
	endif rainopt = 2
	elseif rainopt = 3
30	"RADAR_RAIN_LOC" (char), radarlocationfile (string)
31	"RADAR_RAIN_RATE" (char), radarrainfile (string)
32	"RADAR_VERIFY" (char), radarverifyfile (string)
Note	Reading routines called after we read the radar verify file so that we can write echo output to verify file.
	call ReadRadarRainLocations
	call ReadRadarRainRates (float float float) {pass in units conversions and scale}
	elseif rainopt = 4
33	"SPACE_TIME_STORM" (char), spacetimestormfile (string)
	call ReadSpaceTimeStorm (float float float) {pass in units conversions and scale}
34	"DAD_GRID" (char), dadstormgridfile (string) {output} {this is grid pointer map of rain cell locations in space, similar to generated designraingridfile}
	elseif rainopt = 5
35	"RAINGRIDFREQ" (char), raingridfreq (float) (hrs)
36	"RAINGRIDFILES" (char), raingridfileroot (string)
	endif rainopt <= 2 etc.

Data Group B: Hydrologic Simulation Parameters (continued)

37a (W)	"SNOWOPT" (char), snowopt (int), "MELTOPT" (char), meltopt (int)
Note	snowopt: 0 = no snowfall, 1 = compute snowfall from rain (precipitation) based on air temperature, 2 = snow from gages (IDW interpolation in space), 3 = snow on ground from gages (SNOTEL, SNODAS), 4 = time series grid read. Only options 0, 1, and 2 are implemented. meltopt: 0 = no snowmelt, 1 = compute snowmelt based on modified temperature index, 2+ = options under development. Only options 0 and 1 are implemented.
	if snowopt > 0 or meltopt > 0
37b	"TSNOW" (char), tsnow (float) (°C) {temperature below which precipitation falls as snow and above which snow begins to melt, can be any value but expected value is 0-1 °C}, "SWECOVER" (char), swecover (float) (m) {snow/ice SWE thickness threshold above which infiltration is zero (toggles infiltration on/off}
	endif snowopt > 0 or meltopt > 0
	if meltopt > 0
37c	"LATITUDE" (char), latitude (float) {decimal degrees of latitude for model domain. Positive values represent Degrees North and negative values Degrees South}
	endif meltopt > 0
	if snowopt = 2
38	"SnowIDWradius" (char), sidwradius (float) (m), "SnowIDWexponent" (char), sidwexponent (float) (dimensionless)
39	"NSNOWGAGES" (char), nsg (int) {by default snowfall is uniform if only one snow gage is specified and distributed if there is more than one gage}
	if nsg > 0
40	"CONV1" (char), convunits (float), "CONV2" (char), convtime (float), "SCALE" (char), scale (float)
	for isg = 1, nsg
41	"GAGE" (char), sgid[isg] (int), sgx[isg] (float) (m), sgy[isg] (float) (m), nspairs[isg] (int)
	for ispairs = 1, nspairs[isg]
42	sfintensity[isg][ipairs] (float) (m/s), sftime[irg][ipairs] (float) (hrs)
Note	Records 41 and 42 are repeated as a group for isg = 1, nsg. Record 42 is repeated for ipairs = 1, npairs[isg].
Also Note	Records 38-42 apply for snowopt involving spatial interpolation of point data.
	endif nsg > 0
	elseif snowopt = 3

Data Group B: Hydrologic Simulation Parameters (continued)

Warning	Warn user option(s) not implemented...
	elseif snowopt = 4
43	"SNOWGRIDFREQ" (char), snowgridfreq (float) (hrs)
44	"CONV1" (char), snowconvunits (float), "CONV2" (char), snowconvtime (float), "SCALE" (char), snowscale (float)
45	"SNOWGRIDFILES" (char), snowgridfileroot (string)
	endif snowopt = 2, 3, 4...
	if meltopt > 0
46	"ATMELT" (char), atmelt (float) (m/s/° C), "SRMELT" (char), srmelt (float) (m/s/(W/m2)) {we might want to switch to reading grid values rather than a constant...}
47	"DEM_SLOPE" (char), slopefile (string) (degrees)
	call ReadSlopeFile
48	"DEM_ASPECT" (char), aspectfile (string) (degrees)
	call ReadAspectFile
49	"SKYVIEW" (char), skyviewfile (string) (dimensionless)
	call ReadSkyViewFile
	endif meltopt > 0
	if snowopt > 0 or meltopt > 0
50 (ICsnw)	"INITIAL_SNOW_OVERLAND" (char), initialsnowovfile (string) (m)
	call ReadInitialSnowOverlandFile
	endif snowopt > 0 or meltopt > 0
51 (Wov)	"NUMBER_OF_OVERLAND_FLOW_POINT_SOURCES" (char) (Flow point sources that enter or leave the overland plane by means other than rainfall or runoff and which are specified for individual cells, i.e. well, spring, irrigation diversion, etc.), nqwov (int)
	if nqwov > 0
52	"CONV1" (char), convunits (float), "CONV2" (char), convtime (float), "SCALE" (char), scale (float)
	for i = 1, nqwov
53	qwovrow[i] (int), qwovcol[i] (int), nqwovpairs[i] (int), qwovdescription[i] (string) {qwovdescription is read to end of line as character}
	for ipairs = 1, nqovwpairs[i]
54	qwov[i][ipairs] (float) (m3/s), qwovtime[i][ipairs] (float) (hrs)

Data Group B: Hydrologic Simulation Parameters (continued)

Note	Records 53 and 54 are repeated as a group for i = 1, nqwov. Record 54 is repeated for ipairs = 1, nqwovpairs[i]. qwov units: m^3/s.
	endif nqwov > 0
	if chnopt = 1
55 (Wch)	"NUMBER_OF_CHANNEL_FLOW_POINT_SOURCES" (char) (Flow point sources that enter or leave the channel by means other than rainfall or runoff and which are specified for individual nodes, i.e. mine adit, spring, irrigation diversion, etc.), nqwch (int)
	if nqchw > 0
56	"CONV1" (char), convunits (float), "CONV2" (char), convtime (float), "SCALE" (char), scale (float)
	for i = 1, nqwch
57	qwchlink[i] (int), qwchnode[i] (int), nqwchpairs[i] (int), qwchdescription[i] (string) {qdescription is read to end of line as character}
	for ipairs = 1, nqwchpairs[i]
58	qwch[i][ipairs] (float) (m3/s), qwchtime[i][ipairs] (float) (hrs)
Note	Records 57 and 58 are repeated as a group for i = 1, nqwch. Record 58 is repeated for ipairs = 1, nqwchpairs[i]. qwch units: m^3/s.
	endif nqwch > 0
	endif chnopt = 1
59 (BC)	"NUMBER_OF_WATERSHED_OUTLETS/BOUNDARIES" (char) (Locations where flows leave the model domain via the overland plane and channel network), noutlets (int)
	for i = 1, noutlets
60	"OUTLET_CELL" (char), iout[i] (int) (m), jout[i] (int) (m), sovout[i] (float) (dimensionless), dbcopt[i] (int) {0 = normal depth ($s_f = s_o$), 1 = specified water depth time series}
	if dbcopt[i] > 0
61	"CONV1" (char), convunits (float), "CONV2" (char), convtime (float), "SCALE" (char), scale (float)
62	nqbcpairs[i] (int), qbcdescription[i] (string) {qbcdescription is read to end of line as character}
	for ipairs = 1, nqbcpairs[i]
63	qbc[i][ipairs] (float) (m), qbctime[i][ipairs] (float) (hrs)
Note	Records 60-63 are repeated as a group for i = 1, noutlets. Within this group, Records 61-63 are only input if dbcopt > 0. If dbcopt > 0, Records 61 and 62 are input once and Record 63 is repeated for ipairs = 1, nqbcpairs[i]. qbc units: m.

Data Group B: Hydrologic Simulation Parameters (continued)

64	"NQREPORTS" (char), nqreports (int)
	for iqreport = 1, nqreports
65	qreprow[iqreport] (int), qrepcol[iqreport] (int), qarea[iqreport] (float) (km2), qunitsopt[iqreport] (int) (1 = m3/s, 2= mm/hr), stationid (char)
Note	Record 65 is repeated for iqreport = 1, nqreports

4.2.3   Data Group C: Sediment Transport Simulation Parameters

Data Group C: Sediment Transport Simulation Parameters

Record	*Description*
	if ksim >= 2 then (if ksim >= 2, then sediment transport is simulated)
1	Header (string)
2	"NSOLIDS" (char), nsolids (int), "NSGROUPS" (char), nsgroups (int)
Note	nsolids = number of particle types, nsgroups = number of reporting groups for solids, particles in a reported group are summed for a group total
3	"ADVOVOPT" (char), advovopt (int), "ADVOVSCALE", advovscale (float), "DSPOVOPT" (char), dspovopt (int), "DSPOVSCALE", dspovscale (float), "DEPOVOPT" (char), depovopt (int), "DEPOVSCALE", depovscale (float), "ERSOVOPT" (char), ersovopt (int), "ERSOVSCALE", ersovscale (float), "TNSOVOPT" (char), tnsovopt (int), "TNSOVSCALE", tnsovscale (float), "ELEVOVOPT" (char), elevovopt (int)
Note	advovopt: 0 = advection bypassed, 1 = advection; dspovopt: 0 = dispersion bypassed, 1 = dispersion; depovopt: 0 = deposition bypassed, 1 = deposition using the fall velocity, 2 = deposition using the fall velocity and the probability of deposition; ersovopt: 0 = erosion bypassed, 1 = transport capacity limited erosion (Kilinc-Richardson), 2 = generalized form of transport capacity limited erosion, $k\,q^{\beta}\,S^{\gamma}$ (see Julien and Simons, 1985; Rustomji and Prosser, 2000) with grain shear stress partitioning, 3 = erosion from excess shear stress (untested); tnsovopt: 0 = no transformations bypassed, 1 = transformations occur; elevovopt: 0 = overland elevation update calculations bypassed, 1 = overland elevations updated
	if chnopt > 0
4	"ADVCHOPT" (char), advchopt (int), "ADVCHSCALE", advchscale (float), "DSPCHOPT" (char), dspchopt (int), "DSPCHSCALE", dspchscale (float), "DEPCHOPT" (char), depchopt (int), "DEPCHSCALE", depchscale (float), "ERSCHOPT" (char), erschopt (int), "ERSCHSCALE", erschscale (float), "TNSCHOPT" (char), tnschopt (int), "TNSCHSCALE", tnschscale (float), "ELEVCHOPT" (char), elevchopt (int)

Data Group C: Sediment Transport Simulation Parameters (continued)

Note	advchopt: 0 = advection bypassed, 1 = advection; dspchopt: 0 = dispersion bypassed, 1 = dispersion; depchopt: 0 = deposition bypassed, 1 = deposition using the fall velocity, 2 = deposition using the fall velocity and the probability of deposition; erschopt: 0 = erosion bypassed, 1 = transport capacity limited erosion (Engelund and Hansen), 2 = transport capacity limited erosion (Engelund and Hansen) with grain shear stress partitioning, 3 = erosion from excess shear stress (untested) {other options such as Ackers and White, Yang, etc. can be added here}; tnschopt: 0 = no transformations bypassed, 1 = transformations occur; elevchopt: 0 = channel bed elevation update calculations bypassed, 1 = channel bed elevations updated
	endif chnopt > 0
Also Note	The process options for Records 3 and 4 toggle use (on or off) of the different process routines (for sediment transport). However, when zero is selected for a process option, the process is bypassed but the user must still enter process parameters (as if the option = 1) as required in later records.
5	Header (string): "PARTICLE GROUP NAMES FOR REPORTING"
	for igroup = 1, nsgroups
6	"GROUPNUMBER" (char), particlegroupname[igroup] (string)
Note	Record 6 is repeated for igroup = 1, nsgroups.
7	Header (string): "PARTICLE CHARACTERISTICS: ds, G, ws," {depending on process options add:} "cncopt, tcdov, tcdch," ... "groupnumber, particlename"
Particles	for isolid = 1, nsolids
8a (general)	ds[isolid] (float) (m), spgravity[isolid] (float) (dimensionless), ws[isolid] (float) (m/s)
	if depovopt > 1 or depchopt > 1 or ersovopt > 2 or erschopt > 2
8b (type)	cncopt[isolid] (int) (cohesive/non-cohesive transport option) {0 = non-cohesive (Gessler), 1 = cohesive (Partheniades)}
	endif depovopt > 1 or depchopt > 1 or ersovopt > 2 or erschopt > 2
	if depovopt > 1
8c (depov)	tcdov[isolid] (float) (N/m2 = Pa)
	endif depovopt > 1
	if ersovopt > 2
8d (ersov)	tceov[isolid] (float) (N/m2 = Pa), zageov[isolid] (float) (dimensionless)
	endif ersovopt > 2
	if chnopt > 0
	if depchopt > 1

Data Group C: Sediment Transport Simulation Parameters (continued)

8e (depch)	tcdch[isolid] (float) (N/m2 = Pa)
	endif depchopt > 1
	if erschopt <= 2 {modified Engelund and Hansen}
8f (ersch ≤ 2)	vcch[isolid] (float) (m/s) {critical velocity motion threshold}
	elseif erschopt > 2
8g (ersch > 2)	tcech[isolid] (float) (N/m2 = Pa), zagech[isolid] (float) (dimensionless)
	endif erschopt <= 2
	endif chnopt > 0
8h	sgroupnumber[isolid] (int), particlename[isolid] (string)
Note	Record 8 (a-h) is repeated for isolid = 1, nsolids.
	if tnsovopt > 0 or tnschopt > 0
9	Header (string): "PARTICLE REACTIONS: abrasion, flocculation, dissolution, etc."
Reactions	for i = 1, nsolids
10	"SOLID" (char), isolid (int), "NFIELDS" (char), nfields (int)
	for ifields = 1, nfields[ichem]
11	ncns (int), fieldname (string)
	for icns = 1, ncns
12	sname (char), sid (int), svalue (float)
13	"NSYIELDS" (char), nsyields (int)
	for iyield = 1, nsyields
14	syldfrom[iyield] (int), syldto[iyield] (int), syldprocess[iyield] (int), syield[iyield] (float) (g/g)
Note	Records 10, 11, 12, and 13 are repeated as a group for i = 1, nsolids. Record 10 is input once. Records 11 and 12 are repeated as a group for iflds = 1, nflds[i]. Within this group, Record 12 is repeated for icns = 1, ncns. Record 13 is input once. Record 14 is repeated for iyield = 1, nyields.
	endif tnsovopt > 0 or tnschopt > 0
15 (ICov)	Header (string): "SOIL CHARACTERISTICS: erosion parameters, grain size distribution etc."
16	"NSOILS" (char), nsoils (int) {number of soil types}

Data Group C: Sediment Transport Simulation Parameters (continued)

Soils	for isoil = 1, nsoils
	if infiltopt = 0 {no infiltration, but there still is sediment transport}
	if ersovopt <= 2
17	kusle[isoil] (float) (dimensionless), vcov[isoil] (float) (m/s), porosityov[isoil] (float), soilname[isoil] (string) {for ksim >1, infopt = 0, and ersovopt = 1, kusle is the only soil parameter specified; cusle and pusle are land use parameters...}
	else (ersovopt > 2)
18	mexpov[isoil] (float) (dimensionless), porosityov[isoil] (float), soilname[isoil] (string)
	endif ersovopt <= 2
	elseif infopt > 0 {both infiltration and sediment transport}
	if ersovopt <= 2
19	kusle[isoil] (float) (dimensionless), vcov[isoil] (float) (m/s), porosityov[isoil] (float) (dimensionless), kh[isoil] (float) (m/s), capsh[isoil] (float) (m), soilmd[isoil] (float) (dimensionless), soilname[isoil] (string)
	else (ersovopt > 2)
20	mexpov[isolid] (float) (dimensionless), porosityov[isoil] (float) (dimensionless), kh[isoil] (float) (m/s), capsh[isoil] (float) (m), soilmd[isoil] (float) (dimensionless), soilname[isoil] (string)
	endif ersovopt <= 2
21a (label)	"SoilGSD:" (char)
	for isoil=1, nsolids
21b (gsd)	gsdov[isoil][isolid] (float)
Check	Compute gsdovtot += gsdov[isoil][isolid]. If gsdovtot != 1.0 then abort.
Note	Record 17/18/19/20 and 21 are repeated as a group. Only one Record (17 or 18 or 19 or 20) is entered. The record that is entered depends on the values of infopt and ersovopt. The group is repeated for isoil = 1, nsoils.
	endif infopt = 0
22	Header (string): "LAND USE CHARACTERISTICS: Cusle, grain size distribution"
23	"NLANDS" (char), nlands (int) {number of land use types}
	if ersovopt <= 2
	for iland = 1, nlands
24a	nmanningov[iland] (float) (n units), interceptionclass[iland] (float) (mm), cusle[iland] (float) (dimensionless), pusle[iland] (float) (dimensionless)

Data Group C: Sediment Transport Simulation Parameters (continued)

	if ersovopt = 2
24b	ktranscapov[iland] (float) (kg/m/s), betasov[iland] (float) (dimensionless), gammasfov [iland] (float) (dimensionless), barefractionov[iland] (float) (dimensionless), tcwexpov[iland] (float) (dimensionless)
	endif ersovopt = 2
24c	landname[iland] (string)
	else (ersovopt > 2)
	for iland = 1, nlands
25	nmanningov[iland] (float) (n units), interceptionclass[iland] (float) (mm), ayov[iland] (float) (g/cm2), landname[iland] (string)
	endif ersovopt <= 2
Note	Only one Record (24 or 25) is entered. The record that is entered depends on the value of ersovopt and is repeated for iland = 1, nlands.
26	"LAND_USE" (char), landusefile (string)
	call ReadLandUseFile
27	Header (string): "SOIL STACK AND LAYER PROPERTIES: thickness, grain size distribution"
28	"MAXSTACKOV" (char), maxstackov (int), "MINVOLOV" (char), minvolov (float) (dimensionless), "MAXVOLOV" (char), maxvolov (float) (dimensionless), "STKOVOPT" (char), stkovopt (int) {0 = stack does not collapse, 1 = stack collapses}
Note	maxstackov is the maximum number of layers that can exist in the soil stack. Not all layers need be occupied (i.e. have a thickness greater than zero) at the simulation start.
29	"SOIL_STACK" (char), soilstackfile (string) {initial number of layers in each grid cell}
Note	The soil stack grid specified the number of occupied soil layers that exist in the soil stack at the start of the simulation. The value input specifies nstackov0[i][j] for each cell.
	call ReadSoilStackFile
	for ilayer = maxstackov to 1 (reverse order)
30	"THICKNESS_LAYERn" (char), thicknessgrid (string) {thickness values in meters}
	Read SoilLayerThicknessFile
Note	A soil layer thickness grid must be specified for the maximum possible numberof layers in the soil stack defined by maxstackov. However, not all cells in each layer need to be occupied. A layer is occupied when it has a thickness value greater than zero. A layer is empty when it has a thickness value of zero. Empty layers may exist/occur as you move up the stack. Empty layers cannot exist in the middle or bottom of the stack.
31	"SOIL_TYPES_LAYERn" (char), soilfile (string)

Data Group C: Sediment Transport Simulation Parameters (continued)

	call ReadSoilTypeFile
Note	The information for Records 30 and 31 is read in reverse order (layer 1 is bottom and last).
32	Header (string): "INITIAL SUSPENDED SOLIDS CONCENTRATIONS IN THE OVERLAND PLANE: one grid file for each solids type"
	for isolid=1, nsolids
33	"GRIDFILE_FOR_SOLIDn" (char), initialssovfile (string)
	call ReadInitialSolidsOverland
Note	Record 32 is input once. Record 33 is repeated for isolid = 1, nsolids
	if chnopt > 0
34 (ICch)	Header (string): "SEDIMENT STACK CHARACTERISTICS AND LAYER PROPERTIES: number of layers, thickness, grain size distribution"
35	"MAXSTACKCH" (char), maxstackch (int), "MINVOLCH" (char), minvolch (float) (dimensionless), "MAXVOLCH" (char), maxvolch (float) (dimensionless), "STKCHOPT" (char), stkchopt (int) {0 = stack does not collapse, 1 = stack collapses}
36	"SEDIMENT_PROPERTIES_FILE" (char), sedimentpropertiesfile (string)
	call ReadSedimentPropertiesFile
37	"INITIAL_SUSP_SOLIDS_CHANNELS" (char), initialsschfile (string)
	call ReadInitialSolidsChannelFile
	endif chnopt > 0
	for isolid = 1, nsolids
38 (Wpov)	"NUMBER_OF_OVERLAND_LOADS_FOR_SOLIDSn" (char), nswpov[isolid] (int)
	if nswpov[isolid] > 0
39	"CONV1" (char), convunits (float), "CONV2" (char), convtime (float), "SCALE" (char), scale (float)
	for isw = 1, nswpov[isolid]
40	swpovrow[isolid][isw] (int), swpovcol[isolid][isw] (int), nswpovpairs[isolid][isw] (int), swpovopt[isolid][isw] (int), loadname (string) {read to end of line}
	for iswpair = 1, nswpovpairs[isolid][isw]
41	swpov[isolid][isw][iswpair] (float) (kg/day), swpovtime[isolid][isw][iswpair] (float) (hrs)
	endif nswch[isolid] > 0
Note	Records 38, 39, 40, and 41 are repeated as a group for isolid = 1, nsolids. Record 38 is input once. Record 39 is input once. Records 40 and 41 are repeated as a group for isw = 1, nswpov[isolid]. Record 41 is repeated for iswpair = 1, nswpovpairs[isolid][isw]

Data Group C: Sediment Transport Simulation Parameters (continued)

	for isolid = 1, nsolids
42 (Wdov)	"NUMBER_OF_OVERLAND_DISTRIBUTED_LOADS_FOR_SOLIDSn" (char), nswdov[isolid] (int)
	if nswdov[isolid] > 0
	for isw = 1, nswdov[isolid]
43	"AREAS_FOR_SOLIDSn_WASHOFF_FUNCTIONn" (char), swdovareafile[isolid][isw] (string)
	call ReadSWDovAreaFile
44	"CONV1" (char), convunits (float), "CONV2" (char), convtime (float), "SCALE" (char), scale (float)
45	nswdovpairs[isolid][isw] (int), loadname (string) {read to end of line}
	for iswpair = 1, nswdovpairs[isolid][isw]
46	swdov[isolid][isw][iswpair] (float) (g/m2/mm rain), swdovtime[isolid][isw][iswpair] (float) (hrs)
	endif nswdov[isolid] > 0
Note	Distributed loads are treated as rainfall-driven "washoff" loads where the load is specified as a mass per unit area per unit of rain. A grid of washoff areas for each cell must be input as well as a time series for the load.
Note	Records 42, 43, 44, 45, and 46 are repeated as a group for isolid = 1, nsolids. Record 42 is input once. Records 43, 44, 45, and 46 are repeated as a group for isw = 1, nswdov[isolid]. Record 46 is repeated for iswpair = 1, nswdovpairs[isolid][isw]
	if chnopt > 0
	for isolid = 1, nsolids
47 (Wpch)	"NUMBER_OF_CHANNEL_LOADS_FOR_SOLIDSn" (char), nswch[isolid] (int)
	if nswch[isolid] > 0
48	"CONV1" (char), convunits (float), "CONV2" (char), convtime (float), "SCALE" (char), scale (float)
	for isw = 1, nswch[isolid]
49	swchlink[isolid][isw] (int), swchnode[isolid][isw] (int), nswchpairs[isolid][isw] (int), loadname (string) {read to end of line}
	for iswpair = 1, nswchpairs[isolid][isw]
50	swch[isolid][isw][iswpair] (float) (kg/day), swchtime[isolid][isw][iswpair] (float) (hrs)

Data Group C: Sediment Transport Simulation Parameters (continued)

Note	Records 47, 48, 49, and 50 are repeated as a group for isolid = 1, nsolids. Record 47 is input once. Record 48 is input once. Records 49 and 50 are repeated as a group for isw = 1, nswch[isolid]. Record 50 is repeated for iswpair = 1, nswchpairs[isolid][isw]
	endif nswch[isolid] > 0
	endif chnopt > 0
	for ioutlet = 1, noutlets
51 (BC)	Header (string) {read to end of line} {Example: "SOLIDS BOUNDARY CONDITIONS FOR OUTLET ioutlet"}
	if dbcopt[ioutlet] > 0
52	"CONV1" (char), convunits (float), "CONV2" (char), convtime (float), "SCALE" (char), scale (float)
	for isolid = 1, nsolids
53	nsbcpairs[ioutlet][isolid] (int), bcname (string) {read to end of line} {Example: "BOUNDARY CONDITIONS FOR SOLID TYPE isolid"}
	for isbcpair = 1, nsbcpairs[ioutlet]
54	sbc[ioutlet][isolid][isbcpair] (float) (g/m3), sbctime[ioutlet][isolid][isbcpair] (float) (hrs)
Note	Records 51, 52, 53, and 54 are repeated as a group for ioutlet = 1, noutlets. Record 51 is input once. Records 52, 53, and 54 are only input if dbcopt[ioutlet] > 0. Record 52 is input once. Records 53 and 54 are repeated as a group for isolid = 1, nsolids. Within this group, Record 54 is repeated for isbcpairs = 1, nsbcpairs[ioutlet].
	endif dbcopt[ioutlet] > 0
55	"NSEDREPORTS" (char), nsedreports (int)
	for isedreport = 1, nsedreports
56	sedreprow[isedreport] (int), sedrepcol[isedreport] (int), sedarea[isedreport] (float) (km2), sedunitsopt[isedreport] (int) {1 = g/m3, 2 = kg (mass transported over the reporting interval)}, stationid (char)
Note	Record 56 is repeated for isedreport = 1, nsedreports. Reports are for all solids types simulated
	endif ksim >= 2

4.2.4   Data Group D: Contaminant Transport Simulation Parameters

Data Group D: Contaminant Transport Simulation Parameters

Record	Description
	if ksim >= 3 then (if ksim >= 3, then chemical transport is simulated)
1	Header (string)
2	"NCHEMICALS" (char), nchems (int), "NCGROUPS" (char), ncgroups (int)
Note	nchems = number of chemical types, ncgroups = number of reporting groups for chemicals, chemicals in a reported group are summed for a group total
3	Header (string): "CHEMICAL GROUP NAMES FOR REPORTING"
	for igroup = 1, ncgroups
4	chemgroupname[igroup] (char)
Note	Record 4 is repeated for igroup = 1, ncgroups.
5	Header (string): "PROPERTIES FOR CHEMICALS"
	for ichem = 1, nchems
6	"CHEMICAL n" (char), ichem (int), "NFIELDS" (char), nflds[ichem] (int), "GROUPNUMBER" (char), cgroupnumber[ichem] (int), chemname[ichem] (string)
	for ifields = 1, nfields[ichem]
7	ncns (int), fieldname (string)
	for icns = 1, ncns
8	cname (char), cid (int), cvalue (float)
9	"NCYIELDS" (char), ncyields[ichem] (int)
	for iyield = 1, ncyields
10	cyldfrom[iyield] (int), cyldto[iyield] (int), cyidprocess[iyield] (int), cyield[iyield] (float) (g/g)
Note	Records 5, 6, 7, 8, and 9 are repeated as a group for ichem = 1, nchems. Record 5 is input once. Records 6, 7, and 8 are repeated as a group for iflds = 1, nflds[ichem]. Within this group, Record 8 is repeated for icns = 1, ncns. Record 9 is input once. Record 10 is repeated for iyield = 1, ncyields.
Definitions	Chemical Properties and Reaction Pathways are defined in Section 4.3
11 (ICov)	Header (string): "INITIAL CHEMICAL CONCENTRATIONS IN SOIL"
	for ilayer = maxlayersov, 1, -1 (reverse order, all possible layers even if null)
	for ichem = 1, nchems

Data Group D: Contaminant Transport Simulation Parameters (continued)

12	"SOIL_ICs_LAYER_ilayer_CHEMICAL_n" (char), soilchemfile (string) (mg/kg)
	call ReadSoilLayerChemicalFile()
Note	Record 12 is repeated for for ilayer = maxlayersov, 1, -1 (reverse order) and ichem = 1, nchems. One grid file for each chemical repeated for each layer.
	for ichem = 1, nchems
13	Header (string): "INITIAL CHEMICAL CONCENTRATIONS IN THE OVERLAND PLANE WATER COLUMN: one grid file for each chemical type"
	for ichem=1, nchems
14	"GRIDFILE_FOR_CHEMICALn" (char), initialchemovfile (string)
	call ReadInitialChemicalOverland
Note	Record 13 is input once. Record 14 is repeated for ichem = 1, nchems
	if chnopt > 0
15 (ICch)	Header (string): "INITIAL CHEMICAL CONCENTRATIONS IN SEDIMENT"
16	"CHEMICAL_SEDIMENT_CONCENTRATION_FILE" (char), sedimentchemfile (string) (mg/kg)
	call ReadSedimentChemicalFile
17	"INITIAL_CHEMICAL_CHANNELS" (char), initialchemchfile (string)
	call ReadInitialChemicalChannelFile
	endif chnopt = 1
	for ichem = 1, nchems
18 (Wov)	"NUMBER_OF_OVERLAND_LOADS_FOR_CHEMICALn" (char), ncwpov[ichem] (int)
	if ncwpov[ichem] > 0
19	"CONV1" (char), conunit (float), "CONV2" (char), contime (float), "SCALE" (char), scale (float)
	endif ncwpov[ichem] > 0
	for ichem = 1, nchems
22 (Wdov)	"NUMBER_OF_OVERLAND_DISTRIBUTED_LOADS_FOR_CHEMICALn" (char), ncwdov[isolid] (int)
	if ncwdov[ichem] > 0
	for icw = 1, ncwdov[ichem]

Data Group D: Contaminant Transport Simulation Parameters (continued)

23	"AREAS_FOR_CHEMICALn_WASHOFF_FUNCTIONn" (char), cwdovareafile[ichem][icw] (string)
	call ReadCWDovAreaFile
24	"CONV1" (char), convunits (float), "CONV2" (char), convtime (float), "SCALE" (char), scale (float)
25	ncwdovpairs[isolid][isw] (int), loadname (string) {read to end of line}
	for icwpair = 1, ncwdovpairs[ichem][icw]
26	cwdov[ichem][icw][icwpair] (float) (g/m2/mm rain), cwdovtime[ichem][icw][icwpair] (float) (hrs)
	endif ncwdov[ichem] > 0
Note	Distributed loads are treated as rainfall-driven "washoff" loads where the load is specified as a mass per unit area per unit of rain. A grid of washoff areas for each cell must be input as well as a time series for the load.
Note	Records 22, 23, 24, 25, and 26 are repeated as a group for isolid = 1, nsolids. Record 22 is input once. Records 23, 24, 25, and 26 are repeated as a group for isw = 1, nswdov[isolid]. Record 26 is repeated for iswpair = 1, nswdovpairs[isolid][isw]
	if chnopt > 0
	for ichem = 1, nchems
27 (Wch)	"NUMBER_OF_CHANNEL_LOADS_FOR_CHEMICALn" (char), ncwch[ichem] (int)
	if ncwch[ichem] > 0
28	"CONV1" (char), conunit (float), "CONV2" (char), contime (float), "SCALE" (char), scale (float)
	for icw = 1, ncwch[ichem]
29	cwchlink[ichem][icw] (int), cwchnode[ichem][icw] (int), ncwchpairs[ichem][icw] (int), loadname (string)
	for icwpair = 1, ncwpovpairs[icw]
30	cwch[ichem][icw][icwpair] (float) (kg/day), cwchtime[ichem][icw][icwpair] (float) (hrs)
Note	Records 27, 28, 29, and 30 are repeated as a group for ichem = 1, nchems. Record 27 and 28 are each input once. Records 29 and 30 are repeated for icw = 1, ncwch[ichem]. Record 30 is repeated for icwpair = 1, ncwchpairs[ichem][icw]. Loads input as kg/day.
	endif ncwch[ichem] > 0
	endif chnopt > 0
	for ioutlet = 1, noutlets
	if dbcopt[ioutlet] > 0

Data Group D: Contaminant Transport Simulation Parameters (continued)

31 (BC)	Header (string) {read to end of line} {Example: "CHEMICAL BOUNDARY CONDITIONS FOR OUTLET ioutlet"}
32	"CONV1" (char), convunits (float), "CONV2" (char), convtime (float), "SCALE" (char), scale (float)
	for ichem = 1, nchems
33	nsbcpairs[ioutlet][isolid] (int), bcname (string) {read to end of line} {Example: "BOUNDARY CONDITIONS FOR CHEMICAL TYPE ichem"}
	for icbcpair = 1, ncbcpairs[ioutlet]
34	cbc[ioutlet][ichem][icbcpair] (float) (g/m3), cbctime[ioutlet][ichem][icbcpair] (float) (hrs)
Note	Records 31, 32, 33, and 34 are repeated as a group for ioutlet = 1, noutlets. Record 31 is input once. Records 32, 33, and 34 are only input if dbcopt[ioutlet] > 0. Record 32 is input once. Records 33 and 34 are repeated as a group for ichem = 1, nchems. Within this group, Record 34 is repeated for icbcpairs = 1, ncbcpairs[ioutlet].
	endif dbcopt[ioutlet] > 0
35	"NCHEMREPORTS" (char), nchemreports (int)
	for ichemreport = 1, nchemreports
36	chemreprow[ichemreport] (int), chemrepcol[ichemreport] (int), chemarea[ichemreport] (float) (km2), chemunitsopt[ichemreport] (int) {1 = g/m3, 2 = kg (mass transported over the reporting interval)}, stationid (char)
Note	Record 36 is repeated for: ichemreport = 1, nchemreports. Reports are for all chemical types simulated.
	endif ksim >= 3

4.2.5   Data Group E: Environmental Properties

Environmental properties are developmental features that are not fully implemented at this time. For hydrology and sediment transport simulations (ksim = 1 or 2), Records 1 and 2 must always be entered. For chemical transport simulations (ksim =3), Records 1, 2, 10, and 27 must be entered. If channels are simulated (chnopt > 0), Record 20 and 37 are also entered. For these records, the required input values should each be set to zero until the features of the relevant sections of Data Group E are fully implemented. A more detailed description of environmental functions is presented in Section 4.4

Data Group E: Environmental Properties (air temp, water temp, pH, etc)

Record	Description
1	Header (string) "Data Group E: Environmental Properties"
2	"General_Environmental_Properties_NPROPG" (char), npropg (int)
	for iprop=1, npropg
3	"PROPERTY" (char), pid (int), "CONVUNITS" (char), convunits (float), "SCALE" (char), scale (float), "NFUNCTIONS" (char), nenvgtf (int)
4	"Cell_Multiplier_Value_Grid" (char), parameterfile (string) {value for each cell is used as a multiplier of the time function value for that cell}
	call ReadGeneralEnvironmentFile(pid, convunits, scale)
	if nenvgtf[iprop] > 0
5	"TimeFunction_Pointer_Grid" (char), tfpointerfile (string) {identifies which time function (1–nenvgtf[]) is used by each cell in the model domain}
	call ReadGeneralFunctionPointerFile(iprop)
	for itf=1, nenvgtf[iprop]
6	"TimeFunctionID" (char), tfid (int), "STATIONELEV" (char), stnelevenvg (float), tfname (string)
Note	The station elevation is needed for laspe rate calculations for many parameters such as air temperature, vapor pressure...
7	"CONV1" (char), convunits (float), "CONV2" (char), convtime (float), "SCALE" (char), scale (float)
8	"NTFGPAIRS"(char), nenvgtfpairs[iprop][itf] (int)
	for itfpair = 1, nenvgtfpairs[iprop][itf]
9	envgtf[iprop][itf][itfpair] (float) (units vary), envgtftime[iprop][itf][itfpair] (float) (hrs)
Note	Units for time function values vary.
	if iprop = 2 (if the property is air temperature)
10	"ATLAPSERATE"(char), atlapse (float) (ºC/km)
	end if iprop = 2
	endif nenvgtf[iprop] > 0
Note	Records 3 through 9 are repeated as a group for iprop = 1, npropg. Records 3 and 4 are input once. If nenvgtf[iprop] > 0, Records 5 through 9 are input. Record 5 is input once. Records 6, 7, 8, and 9 are repeated as a group for itf = 1, nenvovtf[iprop]. Within this group, Records 6, 7, and 8 are each input once and Record 9 is repeated for itfpairs = 1, nenvgtfpairs[iprop][itf]. Record 10 is only entered if iprop = 2 (air temperature).

Data Group E: Environmental Properties (continued)

11	"Overland_Environmental_Properties_NPROPOV" (char), npropov (int)
	for iprop=1, npropov
12	"PROPERTY" (char), pidov (int), "CONV1" (char), convunits (float), "SCALE" (char), scale (float), "NFUNCTIONS" (char), nenvovtf (int)
13	"Cell_Value_Grid_Layer_0" (char), parameterfile (string) {grid file for water column}
	call ReadOverlandEnvironmentFile(pidov, 0, conv1, scale)
	if nenvovtf[iprop] > 0
14	"TimeFunction_Pointer_Grid" (char), tfpointerfile (string)
	call ReadOverlandEnvironmentTFPointerFile(pidov, 0)
	endif nenvovtf[iprop] > 0
	for ilayer = maxstackov, 1, -1 {Reverse Order!!}
15	"Cell_Multiplier_Grid_Layer_n" (char), parameterfile (string) {grid file with multiplier values for each soil layer} [Usage: cell value = multiplier * time function value]
	call ReadOverlandEnvironmentFile(pidov, ilayer, conv, scale)
	if nenvovtf[iprop] > 0
16	"TimeFunction_Pointer_Grid" (char), tfpointerfile (string)
	call ReadOverlandEnvironmentTFPointerFile(pidov, ilayer)
	endif nenvovtf[iprop] > 0
	if nenvovtf[iprop] > 0
	for itf=1, nenvovtf[iprop]
17	"TimeFunctionID" (char), tfid (int), tfname (string)
18	"CONV1" (char), convunits (float), "CONV2" (char), convtime (float), "SCALE" (char), scale (float)
19	"NTFOVPAIRS"(char), nenvovtfpairs[iprop][itf] (int)
	for itfpair = 1, nenvovtfpairs[iprop][itf]
20	envovtf[iprop][itf][itfpair] (float) (units vary), envovtftime[iprop][itf][itfpair] (float) (hrs)
Note	Units for time function values vary.
Note	Records 12 through 20 are repeated as a group for iprop = 1, npropov. Records 12 and 13 are input once. If nenvovtf[iprop] > 0, Record 14 is input once. Records 15 and 16 are input and are repeated as a group for ilayer = maxstackov, 1, -1 (in reverse order). Within this group, Record 16 is only input if nenvovtf[iprop] > 0. Records 17, 18, 19 and 20 are repeated as a group for itf = 1, nenvovtf[iprop]. Within this group, Records 17, 18, and 19 are each input once and Record 20 is repeated for itfpairs = 1, nenvovtfpairs[iprop][itf].

Data Group E: Environmental Properties (continued)

Note	Cell multiplier and time function pointer grids are used to define values for each grid cell in combination with specified time functions. Individual cell values are the product of cell multiplier and time function values. Time function pointer grids identify which time function value is applied to each grid cell. If no time functions are input, cell multipler values are treated as constants applied to each cell. Cell multiplier and time function pointer grids are read for the water column (overland runoff) and each soil layer, dependent on the environmental function specified. Some environmental factors do not occur at depth in soils (e.g. light extinction) and only require water column and surface soil layer values. Input requirements for each function are identified in Section 4.4.
	end if nenvovtf[iprop] > 0
	if chnopt > 0
21	"Channel_Environmental_Properties_NPROPCH" (char), npropch (int)
	for iprop=1, npropch
22	"PROPERTY" (char), pidch (int), fname (char), "CONV1" (char), convunits (float), "SCALE" (char), scale (float), "NFUNCTIONS" (char) (dummy), nenvchtf (int)
23	"Node_Multiplier_File_for_Channel_Water" (char), parameterfile (string) {values for water column in a channel format file for each node of each link}
	call ReadChannelEnvironmentFile(pidch, conv1, scale)
	if nenvchtf[iprop] > 0
24	"Node_Time_Function_Pointer_for_Channel_Water" (char), parameterfile (string) {values for water column in a single channel format file for each node of each link}
	call ReadChannelEnvironmentTFPointerFile(pidch, conv1, scale)
	endif nenvchtf[iprop] > 0
25	"Node_Multiplier_File_for_Channel_Sediment" (char), parameterfile (string) {values for water column in a channel format file for each node of each link}
	call ReadChannelEnvironmentFile(pidch, conv1, scale)
	if nenvchtf[iprop] > 0
26	"Node_Time_Function_Pointer_for_Channel_Sediment" (char), parameterfile (string) {values for water column in a single channel format file for each node of each link}
	call ReadChannelEnvironmentTFPointerFile(pidch, conv1, scale)
	endif nenvchtf[iprop] > 0
	if nenvchtf[iprop] > 0
	for itf=1, nenvchtf[iprop]
27	"Node_TimeFunctionID_File" (char), tfid (int), tfname (string)

Data Group E: Environmental Properties (continued)

28	"CONV1" (char), convunits (float), "CONV2" (char), convtime (float), "SCALE" (char), scale (float)
29	"NTFCHPAIRS"(char), nenvchtfpairs[iprop][itf] (int)
	for itfpair = 1, nenvchtfpairs[iprop][itf]
30	envchtf[iprop][itf][itfpair] (float) (units vary), envchtftime[iprop][itf][itfpair] (float) (hrs)
Note	Units for time function values vary.
Note	Records 22 through 27 are repeated as a group for iprop = 1, npropch. Records 22 and 23 are input once. Records 24, 25, 26, and 27 are repeated as a group for itf = nenvchtf[iprop]. Within this group, Records 24, 25, and 26 are input once and Record 27 is repeated for itfpair = 1, nenvchtfpairs[itf].
Note	Node multiplier and time function pointers are used to define values for each node in combination with specified time functions. Individual node values are the product of node multiplier and time function values. Time function pointers identify which time function value is applied to each node. If no time functions are input, node multipler values are treated as constants applied to each node. Node multiplier and time function pointers are read for the water column (stream water) and each sediment layer, dependent on the environmental function specified. Some environmental factors do not occur at depth in sediments (e.g. light extinction) and only require water column and surface sediment layer values. Input requirements for each function are identified in Section 4.4.
	end if nenvchtf > 0
	if ksim > 2
31	"Overland_Particle_Organic_Carbon_FPOCOVOPT" (char), fpocovopt (int)
Note	fpocovopt is the option for specification of particle organic carbon content: 0 = fpoc for the overland plane not specified (the default value is 1.0, i.e. fpocov[isolid][row][col][layer] = 1.0 and is contant in space and time) 1 = a grid of values for each particle type with associated time functions and pointers is entered for the overland plane (water column and soil stack) and a channel environmental property file with associated time functions and pointers is entered for the channel network
	if focopt > 0
	for isolid=1, nsolids
32	"SOLIDTYPE" (char), pid (int), "SCALE" (char), scale (float), "NFUNCTIONS" (char), nfpocovtf (int) {number of FOC time functions for the specified solids type...}
33	"Cell_Value_Grid_Layer_n" (char), parameterfile (string) {cell multiplier grid for water}
	call ReadOverlandFpocFile(isolid, 0, conv1, scale)
	if nfpocovtf[isolid] > 0
34	"TimeFunction_Pointer_Grid" (char), tfpointerfile (string)

Data Group E: Environmental Properties (continued)

	call ReadOverlandFpocTFPointerFile(pid, ilayer, conv1, scale)
	endif nfpocovtf[isolid] > 0
	for ilayer = maxstackov, 1, -1 {Reverse Order!!}
35	"Cell_Value_Grid_Layer_n" (char), parameterfile (string)
	call ReadOverlandFpocFile
	if nfpocovtf[isolid] > 0
36	"TimeFunction_Pointer_Grid" (char), tfpointerfile (string)
	call ReadOverlandFpocTFPointerFile(isolid)
	endif nfpocovtf[isolid] > 0
Note	Records 35 and 36 are repeated as a block for ilayer=maxstackov, 1, -1 (reverse order).
	if nfpocovtf[isolid] > 0
	for itf=1, nfpovovtf[isolid]
37	"TimeFunctionID" (char), tfid (int), tfname (string)
38	"CONV1" (char), convunits (float), "CONV2" (char), convtime (float), "SCALE" (char), scale (float)
39	"NTFOVPAIRS"(char), nfpocovtfpairs[isolid][itf] (int)
	for itfpair = 1, nfpocovtfpairs[isolid][itf]
40	fpocovtf[isolid][itf][itfpair] (float) (dimensionless: 0–1), fpocovtftime[isolid][itf][itfpair] (float) (hrs)
	endif nfpocovtf[isolid] > 0
Note	Records 28 through 36 are repeated as a group for isolid = 1, nsolids. Within this group, Records 28, 29, and 30 are input once. Records 31 and 32 are repeated as a group for ilayer = maxstackov, 1, -1 (in reverse order). If nfpocovtf[isolid] > 0, Record 37 though 40 are input. Within this group, Records 33, 34, 35 are each input once and Record 40 is repeated for itfpairs = 1, nfpocovtfpairs[isolid][itf].
	endif fpocovopt > 0
	if chnopt > 0
41	"Channel_Particle_Organic_Carbon_FPOCCHOPT" (char), fpocchopt (int)
Note	fpocchopt is the option for specification of particle organic carbon content: 0 = fpoc for the channel is not specified (the default value is 1.0, i.e. fpocch[isolid][link][node][layer] = 1.0 and is contant in space and time) 1 = a channel environmental property file with associated time functions and pointers is entered for the channel network

Data Group E: Environmental Properties (continued)

	if fpocchopt > 0
39	pname (char), pid (int), "SCALE" (char), scale (float), "NFUNCTIONS" (char) (dummy), nfpocchtf (int)
40	"Fpoc_PROPERTY_FILE_FOR_CHANNEL" (char), parameterfile (string) {water column and sediment stack}
	call ReadChannelFpocFile(pid, 0, conv1, scale)
	if nfpocchtf[iprop] > 0
	for itf=1, nfpocchtf[iprop]
41	"TimeFunctionID" (char), tfid (int), tfname (string)
42	"CONV1" (char), convunits (float), "CONV2" (char), convtime (float), "SCALE" (char), scale (float)
43	"NTFCHPAIRS"(char), nfpocchtfpairs[iprop][itf] (int)
	for itfpair = 1, nfpocchtfpairs[iprop][itf]
44	fpocchtf[iprop][itf][itfpair] (float) (dimensionless: 0–1), fpocchtftime[iprop][itf][itfpair] (float) (hrs)
	endif fpocchopt > 0
	endif ksim > 2

4.2.6   *Data Group F: Ouput Specification Controls*

Data Group F: Ouput Specification Controls

Record	*Description*
1	Header (string)
	if nqreports > 0
2	Header (string) such as "EXPORT TIME SERIES OUPUTS"
3	"WATER_EXPORT" (char), waterexpfile (string)
	endif nqreports > 0
	if ksim > 1
	if nsedreports > 0
4	"SEDIMENT_EXPORT_ROOT" (char), sedexprootfile (string)
5	"SEDIMENT_EXPORT_EXT" (char), sedextension (string)

Data Group F: Ouput Specification Controls (continued)

	endif nsedreports > 0
	if ksim > 2
	if nchemreports > 0
6	"CHEMICAL_EXPORT_ROOT" (char), chemexpfile (string)
7	"CHEMICAL_EXPORT_EXT" (char), chemextention (string)
	endif nchemreports > 0
	endif ksim > 2
	endif ksim > 1
8	Header (string) such as "POINT-IN-TIME GRID OUPUTS"
9	"RAINFALL_RATES" (char), rainrategrid (string) (Path and file name) (mm/hr)
10	"RAINFALL_DEPTH" (char), raindepthgrid (string) (Path and file name) (mm)
11	"INFILTRATION_RATE" (char), infrategrid (string) (Path and file name) (mm/hr)
12	"INFILTRATION_DEPTH" (char), infdepthgrid (string) (Path and file name) (mm)
13	"WATER_DISCHARGE" (char), qgrid (string) (Path and file name) (m^3/s)
14	"WATER_DEPTH" (char), waterdepthgrid (string) (Path and file name) (m)
	if snowopt > 1
15	"SWEFALL_RATE" (char), swefallgrid (string) (Path and file name) (mm/hr)
16	"SWEFALL_DEPTH" (char), swefalldepthgrid (string) (Path and file name) (mm)
	endif snowopt > 1
	if meltopt > 0
17	"SWEMELT_RATE" (char), swemeltrategrid (string) (Path and file name) (mm/hr)
18	"SWEMELT_DEPTH" (char), swemeltdepthgrid (string) (Path and file name) (mm)
	endif meltopt > 0
	if snowopt > 0 or meltopt > 0
19	"SWE_DEPTH" (char), swedepthgrid (string) (Path and file name) (m)
	endif snowopt > 0 or meltopt > 0
	if ksim > 1
20	"SOLID_CONC_WATER_ROOT" (char), solidsconcwatergridroot (string) (report for total and groups) (Path...\root) (g/m^3)

Data Group F: Ouput Specification Controls (continued)

21	"SOLID_CONC_SURFACE_LAYER_ROOT" (char), solidsconcsurfgridroot (string) (report for total and groups) (Path...\root) (g/m^3)
	if ksim > 2
22	"TOTCHEM_CONC_WATER_ROOT" (char), totchemconcwatergridroot (string) (report for total and groups) (Path...\root) (g/m^3)
23	"DISCHEM_CONC_WATER_ROOT" (char), dischemconcwatergridroot (string) (report for groups) (Path...\root) (g/m^3)
24	"BNDCHEM_CONC_WATER_ROOT" (char), bndchemconcwatergridroot (string) (report for groups) (Path...\root) (g/m^3)
25	"PRTCHEM_CONC_WATER_ROOT" (char), prtchemconcwatergridroot (string) (report for groups) (Path...\root) (g/m^3)
26	"SRBCHEM_CONC_WATER_ROOT" (char), srbchemconcwatergridroot (string) (report for groups) (Path...\root) (mg/kg)
27	"TOTCHEM_CONC_SURFACE_LAYER_ROOT" (char), totchemconcsurfgridroot (string) (report for total and groups) (Path...\root) (g/m^3)
28	"DISCHEM_CONC_SURFACE_LAYER_ROOT" (char), dischemconcsurfgridroot (string) (report for groups) (Path...\root) (g/m^3)
29	"BNDCHEM_CONC_SURFACE_LAYER_ROOT" (char), bndchemconcsurfgridroot (string) (report for groups) (Path...\root) (g/m^3)
30	"PRTCHEM_CONC_SURFACE_LAYER_ROOT" (char), prtchemconcsurfgridroot (string) (report for groups) (Path...\root) (g/m^3)
31	"SRBCHEM_CONC_SURFACE_LAYER_ROOT" (char), srbchemconcsurfgridroot (string) (report for groups) (Path...\root) (mg/kg)
32	"TOTCHEM_CONC_SUBSURF_LAYER_ROOT" (char), totchemconcsubgridroot (string) (report for total and groups) (Path...\root) (g/m^3)
33	"DISCHEM_CONC_SUBSURF_LAYER_ROOT" (char), dischemconcsubgridroot (string) (report for groups) (Path...\root) (g/m^3)
34	"BNDCHEM_CONC_SUBSURF_LAYER_ROOT" (char), bndchemconcsubgridroot (string) (report for groups) (Path...\root) (g/m^3)
35	"PRTCHEM_CONC_SUBSURF_LAYER_ROOT" (char), prtchemconcsubgridroot (string) (report for groups) (Path...\root) (g/m^3)
36	"SRBCHEM_CONC_SUBSURF_LAYER_ROOT" (char), srbchemconcsubgridroot (string) (report for groups) (Path...\root) (mg/kg)
Note:	The subsurface reporting flag (ssrflag) is set to a non-zero value if one or more of file names for subsurface reporting is specified (i.e. not null). Otherwise, ssrflag = 0.
	if ssrflag > 0

Data Group F: Ouput Specification Controls (continued)

37	"REPORTING_HORIZON" (char), horizon (float) (m), "SSROPT" (char), ssropt (int) (0 = point concentration; 1 = depth-weighted average concentration)
Note	The reporting horizon represents the depth below grade level that is used to determine the soil/sediment stack layers where reporting occurs. When ssropt = 0 (point concentrations), the horizon is used to determine which element in the stack is reported. When ssropt = 1 (depth-weighted average), the concentration reported represents an average calculated based on values for all soil/sediment stack layers in the stack from the surface down to the level of the reporting horizon depth. When the reporting horizon falls within the surface layer of the stack, the point concentration and depth concention will be equal (and will be equal to surface soil/sediment concentrations specified in Records 27-31.
	endif ssrflag > 0
38	"DISCHEM_FRAC_WATER_ROOT" (char), dischemfracwatergridroot (string) (report for total and groups) (Path...\root) (dimensionless)
39	"BNDCHEM_FRAC_WATER_ROOT" (char), bndchemfracwatergridroot (string) (report for groups) (Path...\root) (dimensionless)
40	"MBLCHEM_FRAC_WATER_ROOT" (char), mblchemfracwatergridroot (string) (report for groups) (Path...\root) (dimensionless) {mobile = dissolved + bound}
41	"PRTCHEM_FRAC_WATER_ROOT" (char), prtchemfracwatergridroot (string) (report for groups) (Path...\root) (dimensionless)
42	"DISCHEM_FRAC_SURFACE_LAYER_ROOT" (char), dischemfracsurfgridroot (string) (report for total and groups) (Path...\root) (dimensionless)
43	"BNDCHEM_FRAC_SURFACE_LAYER_ROOT" (char), bndchemfracsurfgridroot (string) (report for groups) (Path...\root) (dimensionless)
44	"MOBCHEM_FRAC_SURFACE_LAYER_ROOT" (char), mobchemfracsurfgridroot (string) (report for groups) (Path...\root) (dimensionless) {mobile = dissolved + bound}
45	"PRTCHEM_FRAC_SURFACE_LAYER_ROOT" (char), prtchemfracsurfgridroot (string) (report for groups) (Path...\root) (dimensionless)
46	"CHEMICAL_INFILTRATION_GRID_ROOT" (char), infchemfluxgridroot (string) (report for groups) (Path...\root) (dimensionless)
	endif ksim > 2
47	Header (string) such as "CUMULATIVE TIME GRID OUTPUTS"
48	"NET_ELEVATION_CHANGE" (char), netelevationgrid (string) (Path and filename) (cm)
49	"SOLIDS_GROSS_EROSION" (char), solidserosiongridroot (string) (report for groups) (Path...\root) (kg)
50	"SOLIDS_GROSS_DEPOSITION" (char), solidsdepositiongridroot (string) (report for groups) (Path...\root) (kg)

Data Group F: Ouput Specification Controls (continued)

51	"SOLIDS_NET_ACCUMULATION" (char), solidsnetaccumgridroot (string) (report for groups) (Path...\root) (kg)
	if ksim > 2
52	"CHEMICAL_GROSS_EROSION" (char), chemerosiongridroot (string) (report for groups) (Path...\root) (kg)
53	"CHEMICAL_GROSS_DEPOSITION" (char), chemdepositiongridroot (string) (report for groups) (Path...\root) (kg)
54	"CHEMICAL_NET_ACCUMULATION" (char), chemnetaccumgridroot (string) (report for groups) (Path...\root) (kg)
	endif ksim > 2
	endif ksim > 1
49	Header (string) such as "SIMULATION SUMMARY OUPUTS"
	endif ksim > 2
	endif ksim > 1
49	Header (string) such as "SIMULATION SUMMARY OUPUTS"
50	"DUMP_FILE" (char), dmpfile (string) (Path and file name) {optional, not implemented}
51	"MASS_BALANCE" (char), msbfile (string) (Path and file name) {optional}
52	"SUMMARY_STATISTICS" (char), statsfile (string) (Path and file name) {required}

In Data Group F, the units for model output are presented as noted above (i.e. mm/hr, mm, m, g/m^3, g/s, kg, etc.).

4.3   DEFINITIONS FOR CHEMICAL PROPERTIES AND REACTION PATHWAYS

At this time, the following chemical properties can be specified in Data Group D. It should be noted that not all chemical processes routines that use these properties are implemented. Properties are summarized in groups according to process and pathway: partitioning, biodegradation, dissolution (and precipitation), hydrolysis, oxidation (and reduction), photodegradation, radioactive decay, volatilization, and user-defined reaction.

Chemical Properties and Reaction Pathways (Data Group D)

CID	Name	Description	Units	Default Value
Process/Pathway Options (Switches to Activate/Deactivate Chemical Processes)				
100	partopt	partitioning: 0 = off, 1 = on	N/A	0

Chemical Properties and Reaction Pathways (Data Group D) (continued)

200	bioopt	biodegradation: 0 = off, 1 = on	N/A	0
300	dslopt	dissolution: 0 = off, 1 = on	N/A	0
400	hydoopt	hydrolysis: 0 = off, 1 = on	N/A	0
500	oxiopt	oxidation: 0 = off, 1 = on	N/A	0
600	photoopt	photodegradation: 0 = off, 1 = on	N/A	0
700	radopt	radioactive decay: 0 = off, 1 = on	N/A	0
800	volopt	volatilization: 0 = off, 1 = on	N/A	0
900	udropt	user-defined reaction: 0 = off, 1 = on	N/A	0
Partitioning (100 series)				
100	partopt	partitioning: 0 = off, 1 = on	N/A	0
110	lkp	log partition coefficient	L/kg	N/A
120	lkb	log binding doefficient {partitioning to dissolved organic compounds)	L/kg	N/A
130	lkoc	log organic carbon partition coefficient	L/kg OC	N/A
140	nux	particle interaction paramter	dimensionless	1.0e+20
Biodegradation (200 series)				
200	bioopt	biodegradation: 0 = off, 1 = on	N/A	0
210	kbiowov	first-order biodegradation rate for overland water	1/day	none
215	kbiowov	second-order biodegradation rate for overland water	mL/cells/day	none
220	kbiosov	first-order biodegradation rate for overland soil	1/day	none
225	kbiosov	second-order biodegradation rate for overland soil	mL/cells/day	none
230	kbiowch	first-order biodegradation rate for channel water	1/day	none
235	kbiowch	second-order biodegradation rate for channel water	mL/cells/day	none
240	kbiosch	first-order biodegradation rate for channel sediment	1/day	none

Chemical Properties and Reaction Pathways (Data Group D) (continued)

245	kbiosch	second-order biodegradation rate for channel sediment	mL/cells/day	none
Dissolution (300 series)				
300	dslopt	dissolution: 0 = off, 1 = on	N/A	0
Hydrolysis (400 series)				
400	hydoopt	hydrolysis: 0 = off, 1 = on	N/A	0
Oxidation (500 series)				
500	oxiopt	oxidation: 0 = off, 1 = on	N/A	0
Photodegradation (600 series)				
600	photoopt	photodegradation: 0 = off, 1 = on	N/A	0
Radioactive Decay (700 series)				
700	radopt	radioactive decay: 0 = off, 1 = on	N/A	0
Volatilization (800 series)				
800	volopt	volatilization: 0 = off, 1 = on	N/A	0
User-Defined Reaction (900 series)				
900	udropt	user-defined reaction: 0 = off, 1 = on	N/A	0

4.4   DEFINITIONS FOR ENVIRONMENTAL PROPERTIES

At this time, the following environmental properties can be specified in Data Group E. It should nonetheless be noted that not all environmental or chemical processes routines that use these environmental properties are implemented

Environmental Properties (Data Group E)

PID	Name	Description	Units	Default Value
General Properties				
1	windspeed	wind speed	m/s	0
2	airtemp	air temperature	°C	15
3	solarrad	incident net solar radiation	w/m^2	0
4	cloudcover	fraction of sky covered by clouds	dimensionless	0
5	albedo	fraction of incident net solar rediation reflected by the land surface	dimensionless	0
Overland/Channel Properties (Runoff/Surface Water and Soil/Sediment)				
1	cdoc	concentration of dissolved organic compounds (doc) (C_{doc})	g/m^3	0
2	fdoc	effective fraction of doc for binding (f_{doc})	dimensionless	1
3	hardness	Hardness	g/m^3	0
4	ph	pH	dimensionless	7
5	temperature	water, soil/sediment temperature	°C	10
6	oxrad	oxidant/radical concentration	g/m^3	0
7	bacteria	bacteria concentration	g/m^3	0
8	extinction	light extinction coefficient	1/m	0
9	udrprop	user-defined reaction property	vary	0
Overland/Channel Particle Properties for Chemical Transport and Fate (Particles in Runoff/Surface Water and Soil/Sediment)				
	fpocov	fraction particulate organic carbon (POC) of particles in overland water or soil (f_{poc})	dimensionless	1
	fpocch	fraction particulate organic carbon (POC) of particles in channel water or sediment (f_{poc})	dimensionless	1

4.5   ANCILLARY MODEL INPUT FILES

Ancillary input files are used to describe characteristics of the model domain such as watershed boundary mask, elevations, soil classes, and land uses, etc. Ancillary files for the overland plane are organized as grid files (such as those that can be exported from ESRI's ArcGIS software) that include a header block and a block of grid values specified by row and column. While the exact grid values specified in the ancillary files for the

overland plane differ, the format for each file is the same. Users should note that the header used for grid files differs slightly from the native format exported from ArcGIS software. Ancillary files for the channel network are organized as "channel property files" and "sediment property files". These files specify conditions for each link of the network on a node by node basis. The exact format of each channel or sediment property files differs slightly according to the type of data input.

4.5.1 General Format for Spatial Domain Characteristics Files (Grid Files)

General Format for Spatial Domain Characteristics Files (Grid Files)

Record	Description
1	Header1 (string)
2	"NCOLS" (char), gridcols (int)
3	"NROWS" (char), gridrows (int)
4	"XLLCORNER" (char), xllcorner (float)
5	"YLLCORNER" (char), yllcorner (float)
6	"CELLSIZE" (char), cellsize (float)
7	"NODATAVALUE" (char), nodatavalue (int) {**Note: always use integer -9999**}
	for i = 1, gridrows
	for j = 1, gridcols
8	girdvalue[i][j] (int or float depending on grid) {gridvalue is a sample name...}
Note	Record 8 is repeated for j = 1, grid cols and then repeated again for i = 1, gridrows.
Also Note	Data input is unformatted. However, a typical file will have gridrows number of lines with gridcols number of entries on each line.
Grid Types	Grid files are input for: the simulation mask (int) (imask[][]), ground elevation (float) (elevation[][]), soil types (int) (soiltype[][]), land use classes (int) (landuse[][]), links (int) (link[][]), nodes (int) (nodes[][]), storage depths in the overland plane (float), initial water depths in the overland plane (float), optional design rainfall grid (rainopt = 2) (int) (designraingridindex[][]),and soil stack elements (int).

4.5.2 Description and Organization of Channel Property and Topology Files

Channel Property File {input for tplgyopt = 0}

Record	Description
1	Header {string}
2	CHANLINKS (char), chanlinks (int)
Note	The number of links in the network (nlinks) is already known from the link file. This information is used to check that the channel properties file is compatible with the link file.
	for ilink = 1, nlinks
3	linknum (int) {dummy}, nnodes[ilink] (int)
	for inode = 1, nnodes[ilink]
4	bwidth[ilink][inode] (float) (m), sideslope[ilink][inode] (float) (dimensionless), hbank[ilink][inode] (float) (m), nmanning_ch[ilink][inode] (float) (n units), sinuosity[ilink][inode] (float) (dimensionless), deadstoragedepth[ilink][inode] (float) (m)
Note	Records 3 and 4 are repeated as a group for ilink = 1, nlinks. Record 4 is repeated for inode = 1, nnodes[ilink].

External Channel Topology File {input for tplgyopt = 1} (**Not fully implemented**)

Record	Description
1	Header {string}
2	nlinks (int) {number of links in network}
	for ilink = 1, nlinks
3	linknum (int) {dummy}, nnodes[ilink] (int), downstreamlink[ilink] (int) {the downstream link always starts with the first element of the downstream link...}
	for inode = 1, nnodes[ilink]
4	crow (int), ccol (int), bwidth[ilink][inode] (float), sideslope[ilink][inode] (float), hbank[ilink][inode] (float), nmanningch[ilink][inode] (float), sinuosity[ilink][inode] (float), deadstorage[ilink][inode] (float)
Note	Records 3 and 4 are repeated as a group for ilink = 1, nlinks. Record 4 is repeated for inode = 1, nnodes[ilink].

Channel Initial Water Depth File

Record	Description
1	Header {string}
2	CHANLINKS (char), chanlinks (int)
Note	The number of links in the network (nlinks) is already known from the link file. This information is used to check that the initial water file is compatible with the link file.
	for ilink = 1, nlinks
3	linknum (int) {dummy}, nnodes[ilink] (int)
	for inode = 1, nnodes[ilink]
4	hch0[ilink][inode] (float) (m)
Note	Records 3 and 4 are repeated as a group for ilink = 1, nlinks. Record 4 is repeated for inode = 1, nnodes[ilink].

Channel Transmission Loss Property File (used only when ksim = 1 and ctlopt > 0)

Record	Description
1	Header {string}
2	CHANLINKS (char), chanlinks (int)
Note	The number of links in the network (nlinks) is already known from the link file. This information is used to check that the transmission loss file is compatible with the link file.
	for ilink = 1, nlinks
3	linknum (int) {dummy}, nnodes[ilink] (int)
	for inode = 1, nnodes[ilink]
4	khsed[ilink][inode] (float) (m/s), capshsed[ilink][inode] (float) (m), sedmd[ilink][inode] (float) (dimensionless) {usually zero for initially saturated bed}
Note	Records 3 and 4 are repeated as a group for ilink = 1, nlinks. Record 4 is repeated for inode = 1, nnodes[ilink].

Channel Initial Transmission Loss Depth File

Record	Description
Record	*Description*
1	Header {string}
2	CHANLINKS (char), chanlinks (int)
Note	The number of links in the network (nlinks) is already known from the link file. This information is used to check that the initial transmission loss file is compatible with the link file.
	for ilink = 1, nlinks
3	linknum (int) {dummy}, nnodes[ilink] (int)
	for inode = 1, nnodes[ilink]
4	translossdepth0[ilink][inode] (float) (m) {depth to wetting front}
Note	Records 3 and 4 are repeated as a group for ilink = 1, nlinks. Record 4 is repeated for inode = 1, nnodes[ilink].

4.5.3 *Description and Organization of Sediment Properties and Channel Solids Initial Conditions Files*

Sediment Properties File: Stack Extent, Thickness, Grain Size Distribution, etc.

Record	Description
Record	*Description*
1	Header {string}
2	"CHANLINKS" (char), chanlinks (int), "CHANSOLIDS" (char), chansolids (int), "CHANERSCHOPT" (char), chanerschopt (int) {dummy}, "CHANCTLOPT" (char), chanctlopt (int) {dummy}
Note	The number of links in the network (nlinks) and number of particle types (nsolids) are already known from the link file and main input file. This information is input to check that the sediment properties file is compatible with the channel network and particle types.
	for ilink = 1, nlinks
3	"LINKNUMBER" (char), linknum (int) {dummy}, "NUMNODES" (char), channodes (int) {dummy}
	for inode = 1, nnodes[ilink]
4a	"NODE" (char), nodenum (int) {dummy), "NSTACK" (char), nstackch0[ilink][inode] (int)
	if erschopt > 1
4b	"YIELD" (char), aych[ilink][inode] (float) (g/cm2), "EXPONENT" (char), mexpch[ilink][inode] (float) (dimensionless)

Sediment Properties File: Stack Extent, Thickness, Grain Size Distribution (continued)

	endif erschopt > 1
	if ctlopt > 0
4c	"KHSED" (char), khsed[ilink][inode] (float) (m/s), "CAPSHSED" (char), capshsed[ilink][inode] (float) (m), "SEDMD" (char), sedmd[ilink][inode] (float) (dimensionless)
	endif ctlopt > 0
Note	Record 4 has up to three components. Record 4a is always input. Record 4b is input only if erschopt > 1. Record 4b is input only if ctlopt > 1.
	for ilayer = nstackch0[ilink][inode], 1; ilayer-- {reverse order}
5	"LAYER" (char), layernum (int) {dummy}, "THICKNESS" (char), hlayerch0[ilink][inode][ilayer] (float) (m) {initial thickness of the layer}, "WIDTH" (char), bwlayerch0[ilink][inode][ilayer] (float) (m) {initial bottom width}, "POROSITY" (char), porositych[ilink][inode][ilayer] (float) (dimensionless)
Note	The width of each layer in the stack (bwlayerch0) must be less than or equal to the width of each overlying layer and also less than or equal to the channel bottom width specified in the channel properties file for every link and node in the network.
6a	"GSD:" (char)
	for isolid = 1, nsolids
6b	gsdch[isolid][ilink][inode][layer] (float)
Note	Records 3, 4a/b/c, 5, and 6a/b are repeated as a group for ilink = 1, nlinks. Within this outer block, Records 4a/b/c, 5, and 6a/b are repeated for inode=1, nnodes[ilink]. Within this inner block, Records 5 and 6a/b are repeated for ilayer = nstackch0[ilink][inode], 1 (ilayer--). Within this innermost block, Record 6b is repeated for isolid = 1, nsolids.
Check	Compute gsdchtot =+ gsd[ilink][inode][isolid][ilayer] and check that gsdchtot = 1.0. Abort if gsdchtot != 1.0 for each node of each link...

Channel Initial Suspended Solids File: solids concentration in channel water

Record	*Description*
1	Header {string}
2	"CHANLINKS" (char), chanlinks (int), "CHANSOLIDS" (char), chansolids (int)
Note	The number of links in the network (nlinks) and number of particle types (nsolids) are already known from the link file and main input file. This information is input as a check that the initial solids file is compatible with the channel network and particle types.
	for ilink = 1, nlinks

Channel Initial Suspended Solids File: solids concentration in channel water (continued)

3	"LINKNUMBER" (char), linknum (int) {dummy}, "NUMNODES" (char), channodes (int) {dummy}
	for inode = 1, nnodes[ilink]
4	"NODE" (char), nodenum (int) {dummy)
	for isolid = 1, nsolids
5	csedch[isolid][ilink][inode][0] (float) (g/m^3)
Note	Records 3, 4, and 5 are repeated as a group for ilink = 1, nlinks. Within this block, Record 4 and 5 are repeated as a group for inode = 1, nnodes[ilink]. Within this innermost block, Record 5 is repeated for isolid = 1, nsolids.

4.5.4 *Description and Organization of Channel Chemical Initial Conditions Files*

Channel Initial Conditions File: Chemical in Channel Water

Record	*Description*
1	Header {string}
2	"CHANLINKS" (char), chanlinks (int), "CHANCHEMS" (char), chanchems (int)
Note	The number of links in the network (nlinks) and number of chemical types (nchems) are already known from the link file and main input file. This information is input as a check that the initial chemical file is compatible with the channel network and chemical types.
	for ilink = 1, nlinks
3	"LINKNUMBER" (char), linknum (int) {dummy}, "NUMNODES" (char), channodes (int) {dummy}
	for inode = 1, nnodes[ilink]
4	"NODE" (char), nodenum (int) {dummy)
	for ichem = 1, nchems
5	cchemch[isolid][ilink][inode][0] (float) (g/m^3)
Note	Records 3, 4, and 5 are repeated as a group for ilink = 1, nlinks. Within this block, Record 4 and 5 are repeated as a group for inode = 1, nnodes[ilink]. Within this innermost block, Record 5 is repeated for ichem = 1, nchems.

Chemical Initial Conditions File: Chemical in the Sediment Bed

Record	Description
Record	*Description*
1	Header {string}
2	"CHANLINKS" (char), chanlinks (int), "CHANCHEMS" (char), chanchems (int)
Note	The number of links in the network (nlinks) and number of chemicals (nchems) are already known from the link file and main input file. This information is input to check that the chemical conditions file is compatible with the channel network and chemicals.
	for ilink = 1, nlinks
3	"LINKNUMBER" (char), linknum (int) {dummy}, "NUMNODES" (char), channodes (int) {dummy}
	for inode = 1, nnodes[ilink]
4	"NODE" (char), nodenum (int) {dummy), "NSTACK" (char), nstackch0[ilink][inode] (int) {dummy}
	for ilayer = nstackch0[ilink][inode], 1; ilayer-- {reverse order}
5a	"LAYER" (char), layernum (int) {dummy}
	for ichem = 1, nchem
5b	"Chemical#" (char) {dummy}, cchemch[isolid][ilink][inode][layer] (float) (mg/kg)
Note	Records 3, 4, and 5a/b are repeated as a group for ilink = 1, nlinks. Within this outer block, Records 4, and 5a/b are repeated for inode=1, nnodes[ilink]. Within this inner block, Records 5a/b are repeated for ilayer = nstackch0[ilink][inode], 1 (ilayer--). Within this innermost block, Record 5b is repeated for ichem = 1, nchem.

4.5.5 *Description and Organization of Environmental Properties Files*

Environmental multiplier value and time function identification files must be provided for each environmental property specified. Files for the overland plane follow the general gird file format used for spatial domain characteristics (see Section 4.5.1). Files for the channel network follow a modified form of the file format used for channel properties. The specific modification for channels is that environmental multiplier values and time function identifications must be specified for every layer in the sediment stack (i.e. layers 1-maxstackch) even if some layers are empty (nulls) at the start of the simulation.

Channel Multiplier or Time Function Identification File for Water Column

Record	Description
Record	*Description*
1	Header {string}

Channel Multiplier or Time Function Identification File for Water Column (continued)

2	*If environmental property multiplier values are being specified:* "CHANLINKS" (char), chanlinks (int), "Multiplier_Values_for_ENVPROPCH_ID" (char), pidch (int) *If environmental property time function identifiers are being specified:* "CHANLINKS" (char), chanlinks (int), "TFIdentifer_Values_for_ENVPROPCH_ID" (char), pidch (int)
Note	The number of links in the network (nlinks) and channel property identifier (pidch) are already known from the main input file. This information is input as a check that the file contents match the conditions intended.
	for ilink = 1, nlinks
3	"LINKNUMBER" (char), linknum (int) {dummy}, "NUMNODES" (char), channodes (int) {dummy}
	for inode = 1, nnodes[ilink]
4	"NODE" (char), nodenum (int) {dummy}
5	*If environmental property multiplier values are being specified:* "Multiplier_for_Water", envch [pidch][ilink][inode][0] (float) (units vary) *If environmental property time function identifiers are being specified:* "TFID_for_Water",envchtfid [pidch][ilink][inode][0] (int)
Note	Records 3, 4, and 5 are repeated as a group for ilink = 1, nlinks. Within this block, Record 4 and 5 are repeated as a group for inode = 1, nnodes[ilink]. Within this innermost block, Record 5 is repeated for ichem = 1, nchems.

Channel Multiplier or Time Function Identification File for Sediment Bed

Record	*Description*
1	Header {string}
2	*If environmental property multiplier values are being specified:* "CHANLINKS" (char), chanlinks (int), "Multiplier_Values_for_ENVPROPCH_ID" (char), pidch (int) *If environmental property time function identifiers are being specified:* "CHANLINKS" (char), chanlinks (int), "TFIdentifer_Values_for_ENVPROPCH_ID" (char), pidch (int)

Channel Multiplier or Time Function Identification File for Sediment Bed (continued)

Note	The number of links in the network (nlinks) and channel property identifier (pidch) are already known from the main input file. This information is input as a check that the file contents are compatible with the channel network and chemicals.
	for ilink = 1, nlinks
3	"LINKNUMBER" (char), linknum (int) {dummy}, "NUMNODES" (char), channodes (int) {dummy}
	for inode = 1, nnodes[ilink]
4	"NODE" (char), nodenum (int) {dummy), "MAXSTACK" (char), maxstackch {dummy}
	for ilayer = maxstackch, 1; ilayer-- {reverse order}
5a	"LAYER" (char), layernum (int) {dummy}, "TFID",
5b	"Chemical#" (char) {dummy}, cchemch[isolid][ilink][inode][layer] (float) (mg/kg) *If environmental property multiplier values are being specified:* "CHANLINKS" (char), chanlinks (int), "Multiplier_Values_for_ENVPROPCH_ID" (char), pidch (int) *If environmental property time function identifiers are being specified:* "CHANLINKS" (char), chanlinks (int), "TFIdentifer_Values_for_ENVPROPCH_ID" (char), pidch (int)
Note	Records 3, 4, and 5a/b are repeated as a group for ilink = 1, nlinks. Within this outer block, Records 4, and 5a/b are repeated for inode=1, nnodes[ilink]. Within this inner block, Records 5a/b are repeated for ilayer = nstackch0[ilink][inode], 1 (ilayer--). Within this innermost block, Record 5b is repeated for ichem = 1, nchem.

Evironmental functions are still being developed. Examine the TREX source for details until code development and documentation efforts are completed.

4.5.6 Description and Organization of Radar Rainfall Locations and Radar Rainfall Rates Files

These ASCII file formats were developed by Julie Javier, Jim Smith and Mary Lynn Baeck at Princeton University. Radar location and rainfall rate files are generated by intersecting data from a radar with a watershed and producing a field with constant x- and y-coordinate spacing. This is typically done in-part with TITAN software from Mike Dixon at NCAR. Radar locations are typically supplied in geographic coordinates (latitude and longitude). These must be converted UTM coordinates for use with TREX. There is no check of the radar coordinates relative to the watershed location. The user must ensure that at least one radar cell intersects the watershed. This is typically done in a GIS. These files must be input when rainfall option 3 is selected in Data Group B.

Radar Locations File [ReadRadarRainLocations.c]

Record	*Description*
1	Header {string}
2	"RADARCELLSPACING" (char), radarcellw (float), "NRADARCELLS" (char), nrg (int)
Note	radar cell spacing is used for the maximum distance on restricted nearest neighbor interpolation

Radar Locations File [ReadRadarRainLocations.c] (continued)

	for igage = 1, nrg
3	rgx (float), rgy (float)
Note	rgx and rgy are x and y coordinates of each radar cell. Record 3 generally consists of a two column format in x and y. Read is unformatted. An example file is a 200 km x 200 km field at 1km spacing. There are 40000 locations. The file is generally a 40000 x 2 array (rows/cols). It has 2 values (x and y) (cols) on each row, with 40000 rows.

Radar Rain Rates File [ReadRadarRainRates.c]

Record	*Description*
1	Header {string}
2	"RADARTIMEINC" (char), timeincrement (float) {minutes}, "RADARDURATION" (char), stormduration (int) {hours}
Note	Units for timeincrement must be in minutes. Use of this fixed increment automatically assign the time for each intensity value so that we have a "pair" of values (intensity, time). Also, stormduration is a simple surrogate for number of rows in file. The
Note	the time increment is used in conjunction with the duration to estimate how many rows there are to read in the file. Later version will just scan entire file to determine properties.
NOTE	The file input is currently limited in that nrpairs is a FIXED length for all radar cell locations. The current supplied radar fields are uniform in time (same duration and increment at all locations). Unlike the rain data input into the main TREX input file, we have no current provision to read nrpairs[irg] so it is fixed. Later versions may possibly eliminate this restriction.
	for jrpairs = 1, nrpairs {nrpairs currently input as stormduration}
	for irgage = 1, nrg
3	rfintensity (float)[irgage][jrpairs]

Radar Rain Rates File [ReadRadarRainRates.c] (continued)

NOTE!	The radar rain rate data file is formatted as follows: each value on a single row (line) represents a rainfall intensity at location irgage. The entire row is the intensity at all locations at one time. Each column represents a different location irgage. Each row represents a single time step (jrpair). Record 3 looping order is "opposite" that of rain gages entered into the main TREX input file. Here, we loop over i fastest (inner loop) then j. We loop over columns in the file first then rows, so loop order is OPPOSITE to other rain inputs. However, we assign them to the same array locations. In order to keep consistency throughout rain routines, we have the TREX reading convention: rfintensity[row][column] = rfintensity[i][j] = rfintensity[nrg][nrpairs]. Record 3 is repeated for jrpairs=1,nrpairs and irgage=1,nrg. An example is a 200 km x 200 km field for one hour at 5 minutes duration. There are 40,000 locations (200x200) = nrg on each row. There are 12 rows (60 minutes/5 minutes = 12), so nrpairs =12. The 2D array is then 12 (nrows) x 40000 (ncols).

4.5.7 *Description and Organization of Space-Time Storm Files*

This ASCII file format was developed by John England while at the Bureau of Reclamation, to read in depth-area duration (DAD) data from the USACE/USBR Storm Catalog. This file must be input when rainfall option 4 is selected in Data Group B.

Space-Time Storm File [ReadSpaceTimeStorm.c]

Record	*Description*
1	Header {string}
2	"NUM_DURATIONS" (char), ndurations (int), "MAX_DUR" (char), maxdur (float)
Note	maxdur is used to limit the duration of the input storm. We may implement this first by trial with the user knowing the table explicitly and only reading in the ndurations.
3	"NUM_AREAS" (char), nareas (int), "MAX_AREA" (char), maxarea (float)
Note	maxara is used to limit the area size for the input storm. We may implement this first by trial with the user knowing the table explicitly and only reading in the nareas of interest.
4	"STORM_CENTER_LOCATION" (char), raincenterx (float), raincentery (float) {raincenterx and raincentery are UTM storm center coordinates; these are fixed scalars for rgx, rgy} {estimate from lat-long in decimal degrees using CORPSCON}
5	"STORM_SHAPE_RATIO" (char), stormelong (float), "STORM_ORIENT" (char), stormorient (float) {decimal degrees}
Note	stormelong is limited from [1.0, 8.0] where 1.0 will give a circle. stormorient is limited from [0.0, 360.0] and is defined from positive ordinate (y-axis) (North) in clockwise direction; see HMR 52 for definitions.
	for iduration = 2, ndurations {start time is shifted by one so zero is time 1}
6	dadtime[iduration] (float) {hours} {read n values on one line; local, passed to rftime}

Space-Time Storm File [ReadSpaceTimeStorm.c] (continued)

	end loop over ndurations
	for iareas = 1, nareas
7	rainarea[iarea] (float) {mi2} {read one value as the first column on the line}
	for iduration = 2, ndurations
8	daddepth[iarea][iduration] (float) {inches} {read n values on one line, local, pass to rfintensity after conversion to rate}
	end loop over ndurations
	end loop over nareas
Note	See standard D-A-D sheet to understand format. Data are in U.S. English units as source is USACE/USBR. Record 6 is on one line. Record 7 is the first column where nareas equals the number of rows. Record 8 is the cumulative rainfall depths (ndurations long) for that iarea, on the same line as that iarea.

4.5.8   Description and Organization of Environmental Properties Files

At this time, ancillary files for environmental properties as specified in Data Group E are not fully implemented or documented. Examine the TREX source for details until code development and documentation efforts are completed.

4.5.9   Restart Information File and Restart Directory Structure

To use simulation restart capabilities, the user must specify a restart option (see Section 3.5 and Table 3-3) as well as an primary restart information file. The name and location of the primary restart information file are hardwired in the TREX code. When a restart option is used, TREX reads from a file named "restart-info.txt" that must be located in a directory named "Restart", such that the path to this file is: "Restart/restart-info.txt". The restart-info.txt file contains the path and names and header information for all ancillary files created or read by TREX. Note that the restart information file and restart directory names are case sensitive. The use of any subdirectory paths within the "Restart" directory is at the discretion of the user.

Restart Information File: restart-info.txt (this file name is case sensitive)

Record	*Description*
1	Header1 (string) {descriptive header for restart-info.txt file}
	if ksim >= 2 then (if ksim >= 2, then sediment transport is simulated)

Restart Information File: restart-info.txt (this file name is case sensitive) (continued)

2	Header (string) {header for soil stack properties}
3	"SOIL_STACK" (char) {dummy}, restartfile (string) {soil stack file path and name}
4	Header (string) {header for soil stack file} {used when restart files are written}
	for ilayer = maxstackov to 1 (reverse order)
5	"SOIL_LAYER_THICKNESS" (char) {dummy}, restartfile (string) {soil layer thickness file path and name}
6	Header (string) {header for soil layer thickness file} {used when restart files are written}
7	"SOIL_LAYER_VOLUME" (char) {dummy}, restartfile (string) {soil layer volume file path and name}
8	Header (string) {header for soil layer volume file} {used when restart files are written}
9	"SOIL_LAYER_MINIMUM_VOLUME" (char) {dummy}, restartfile (string) {soil layer minimum volume file path and name} {soil stack control data}
10	Header (string) {header for soil layer minimum volume file} {used when restart files are written}
11	"SOIL_LAYER_MAXIMUM_VOLUME" (char) {dummy}, restartfile (string) {soil layer maximum volume file path and name} {soil stack control data}
12	Header (string) {header for soil layer maximum volume file} {used when restart files are written}
13	"SOIL_LAYER_ELEVATION" (char) {dummy}, restartfile (string) {soil layer elevation file path and name} {soil stack control data}
14	Header (string) {header for soil layer elevation file} {used when restart files are written}
15	"SOIL_LAYER_TYPE" (char) {dummy}, restartfile (string) {soil type file path and name} {soil type for each cell of this layer}
16	Header (string) {header for soil layer type file} {used when restart files are written}
	for isolid = 1 to nsolids
17	"SOIL_LAYER_SOLIDS_CONC" (char) {dummy}, restartfile (string) {soil layer solids concentration file path and name} {concentration of solids type "isolid" in each cell of this layer}
18	Header (string) {header for soil layer solids concentration file} {used when restart files are written}
	end loop over solids
	if ksim >= 3 then (if ksim >= 3, then chemical transport is simulated)
	for ichem = 1 to nchems

Restart Information File: restart-info.txt (this file name is case sensitive) (continued)

19	"SOIL_LAYER_CHEM_CONC" (char) {dummy}, restartfile (string) {soil layer chemical concentration file path and name} {concentration of chemical type "ichem" in each cell of this layer}
20	Header (string) {header for soil layer chemical concentration file} {used when restart files are written}
	end loop over chemicals
	endif ksim >= 3
	end loop over soil layers
	endif ksim >= 2
Note	Records 5-18 are repeated as a group for ilayer = maxstackov to 1. Within this outer block, Records 5-16 are each entered once. Records 17-18 are repeated as a block for isolid = 1 to nsolids. Records 19-20 are repeated as a block for ichem = 1 to nchems. The entire block is then repeated for each soil layer.
21	Header (string) {header for sediment stack properties}
22	"SEDIMENT_STACK" (char) {dummy}, restartfile (string) {sediment stack propeties file path and name} {ancillary file containing all sediment stack information such as sediment layer geometry, solids and chemical concentrations, etc. for each node of each link}
23	Header (string) {header for sediment stack properties file} {used when restart files are written}
	if rstopt = 2
24	Header (string) {header for overland water properties}
25	"OVERLAND_WATER_DEPTH" (char) {dummy}, restartfile (string) {overland water depth file path and name} {depth of water in overland plane, including overland portion of channel cells}
26	Header (string) {header for overland water depth file} {used when restart files are written}
	for isolid = 1 to nsolids
27	"OVERLAND_WATER_SOLIDS" (char) {dummy}, restartfile (string) {overland water solids concentration file path and name} {concentration of solid type "isolid" in overland plane, including overland portion of channel cells }
28	Header (string) {header for overland water solids concentration file} {used when restart files are written}
	end loop over solids
Note	Records 27-28 are repeated as a group for isolid = 1 to nsolids.
	for ichem = 1 to nchems

Restart Information File: restart-info.txt (this file name is case sensitive) (continued)

29	"OVERLAND_WATER_CHEM" (char) {dummy}, restartfile (string) {overland water chemical concentration file path and name} {concentration of chemical type "ichem" in overland plane, including overland portion of channel cells }
30	Header (string) { header for overland water chemical concentration file} {used when restart files are written}
	end loop over chemicals
Note	Records 29-30 are repeated as a group for ichem = 1 to nchems.
31	Header (string) {header for channel water properties}
32	"CHANNEL_WATER" (char) {dummy}, restartfile (string) {channel properties file path and name} {ancillary file containing water depth, solids concentrations, and chemical concentrations for each node of each link}
33	Header (string) {header for sediment stack properties file} {used when restart files are written}

5.0 DESCRIPTION AND ORGANIZATION OF MODEL OUTPUT FILES

TREX produces six categories of output files that echo model inputs, report simulation errors, and present a variety of simulation results. These six categories of output are:

1. Input data echo report;
2. Simulation error report;
3. Export times series;
4. Point-in-time grids;
5. Cumulative time grids; and
6. Simulation performance summaries.

The names of the specific output files in each category are specified by the user in the main model input file. Descriptions of the output in each category follow.

5.1   INPUT DATA ECHO REPORT

The input data echo report presents a summary (echo) of all model inputs read by TREX for a simulation. The echo file is useful for debugging model inputs. If input data are misaligned or other problems with data specified in the main model input file or any ancillary file are detected, the data summarized in the echo file will differ from the data in the input files.

A secondary input data echo report, the radar verification file, is created when rainfall option (rainopt) 3 is selected in Data Group B and radar rainfall data are used to define precipitation forcing functions. The radar verification file is used to echo the input radar locations and rainfall rates read by the model. The file is also used to print selected intermediate computations so the user can check that input data are correctly read by the model and are assigned to the correct locations. The following routines write output to the verification file: ReadDataGroupB, ReadRadarRainLocations, ReadRadarRainRates, and InitializeWater. Intermediate computations and checks that are written to the verification file include: a radar rainfall field in same units and format as read in (ReadRadarRainRates); rate and time pairs (complete time series) for first and last radar locations to check arrays (ReadRadarRainRates); radar grid pointer that determines the TREX model domain cells that each radar gage applies to (InitializeWater).

5.2   SIMULATION ERROR REPORT

The simulation error report presents a summary of numerical integration errors detected during a run. This file provide information that may be helpful for diagnosing model stability errors such as the simulation time at which an error was detected, which model

routine detected the error, and the cell in which the error occurred. For a successful simulation, the error file will be empty except for an echo of the main model input file used for the simulation.

5.3 EXPORT TIME SERIES OUTPUTS

Export times series are tabular outputs of values of model state variables at specified points in space (reporting stations) reported over time. Results are output according to the print interval specified in Data Group A of the main model input file. Export outputs can be specified for hydrology (water depth or flow at a station), sediment transport (solids concentration or load at a station), and chemical transport (chemical concentration or load at a station). Export output files are written in a tab-delimited format to facilitate post-processing and analysis in spreadsheet or statistical software.

5.4 POINT-IN-TIME OUTPUTS

Point-in-time outputs are row-column grid files of model state variables over the entire spatial domain reported at a specific point in time. Results are output according to the grid print interval specified in Data Group A of the main model input file. Point-in-time outputs can be specified for hydrology (water depth or flow), sediment transport (solids concentrations including the sum of all solids types), and chemical transport (chemical concentrations by specific phase and phase distribution fractions). Point-in-time output files are written in a format that can be directly imported into geographic information system (GIS) software (in particular ESRI's ArcGIS or Arc/Info) for individual display and the creation of animations (sequential displays of grids for consecutive points in time).

5.5 CUMULATIVE TIME OUTPUTS

Cumulative time outputs are row-column grid files of model state variables over the entire spatial domain reported at the end of the simulation. Results are output once for the entire time period simulated. Cumulative time outputs can be specified for a number of parameters such as the net elevation change and the gross erosion, gross deposition, and net accumulation of solids or chemicals. Cumulative time output files are written in a format that can be directly imported into geographic information system (GIS) software (in particular ESRI's ArcGIS or Arc/Info).

5.6 SIMULATION SUMMARY OUTPUTS

Simulation summary outputs are divided into three formats: 1) a model dump file; 2) a detailed mass balance file; and 3) a summary statistics file. Each of these formats is described below.

5.6.1   The Model Dump File

Note: this output format is not yet implemented. Computer code to control when data is written to the dump file exists. However, the dump file write module is just a skeleton. Once implemented, the model dump file will be a binary (direct access) file that contains an element-by-element, process-by-process, direction-by-direction report (dump) for each state variable for all points in space and all points in time in the model domain. A post-processing program will need to be created to extract binary output and write it in tabular or grid form for futher post-processing and analysis.

5.6.2   Detailed Mass Balance File

The detailed mass balance file is a file that contains an element-by-element, process-by-process, direction-by-direction cumulative summary of mass (or volume) that passed through each element (overland cell or channel network node) in the model demain during the simulation period. Mass balance files are written in a tab-delimited format to facilitate post-processing and analysis in spreadsheet or statistical software.

5.6.3   Summary Statistics File

Summary statistics file is a simple text file that contains brief summaries of model performance and output for hydrology, sediment transport for the sum of all solids types and each solids type simulated, and chemical transport for each chemical type simulated. For each output class a simple mass balance is presented along with minimum and maximum values for the overland plane and channel network. The overall simulation runtime (wall clock, not CPU time) is also presented at the end of the file.

6.0 DEVELOPMENTAL FEATURES

The TREX framework is designed to be modular to allow future development and addition of expanded features. Future development plans call for addition of a wide range of chemical and particle transformation processes. In particular, chemical biodegradation, hydrolysis, oxidation, photolysis, radioactive decay, volatilization, and dissolution processes may be added to the basic chemical transport submodel. Further development plans also include addition of a nutrient submodel to simulate nitrogen, phosphorus, and carbon cycles at the watershed scale.

To facilitate future expansion, a number of developmental features are present in TREX. These developmental features include provisions for:

1. Additional mass transfer and transformation processes for chemicals;

2. Mass transformation processes for solids (such as mineralization and abrasion);

3. Soil and sediment erosion rate formulations (erosion rates are presently estimated from transport capacity relationships); and

4. Environmental conditions (such as temperature, wind speed, solar radiation, etc.).

While not fully implemented, code for these features already exists in TREX in order to provide a template and speed full development. Descriptions of a number of these developmental features follow.

6.1 CHEMICAL MASS TRANSFER AND TRANSFORMATION PROCESSES

As previously noted, future development plans call for addition of a wide range of chemical and particle transformation processes. In particular, chemical biodegradation, hydrolysis, oxidation, photolysis, radioactive decay, volatilization, and dissolution processes may be added to the basic chemical transport submodel. Overviews of most of these processes are presented in the WASP5 (Ambrose et al. 1993) and IPX (Velleux et al. 2001) manuals. However, the basic computer code needed to implement all of these processes already exists within the TREX framework. Further, code for chemical dissolution and simple first-order biodegradation is fully developed but is untested. Because of their advanced state of development in TREX, overviews of chemical biodegradation and dissolution follow.

6.1.1 *Chemical Biodegradation*

Chemicals that can be metabolized or co-metabolized by bacteria or other microbes may be subject to transformation by biodegradation. The chemical biodegradation flux may be expressed as a simple first-order process as (after Ambrose et al. 1993):

$$\hat{J}_{bc} = k_b C_c V \tag{6.1}$$

where: $\hat{J}_{bc}$ = chemical biodegradation flux [M/T]

k_b = first-order chemical biodegradation rate [1/T]

C_c = total chemical concentration in the water column or porewater [M/L^3]

V = bulk volume (water and particles) [L^3]

6.1.2 Chemical Dissolution

Chemicals that exist in a pure solid phase and are not sorbed to particles can enter solution by dissolution. The chemical dissolution flux may be expressed as a mass rate of transfer from the pure solid phase to the dissolved (aqueous) phase as (Cussler, 1997; Lynch et al. 2003):

$$\hat{J}_{sc} = k_d \alpha (S - f_d C_c) \tag{6.2}$$

where: $\hat{J}_{sc}$ = chemical dissolution mass flux [M/T]

k_d = mass transfer coefficient for chemical dissolution [L/T]

= $\dfrac{D}{\delta}$

D = aqueous phase chemical diffusion coefficient for dissolution [L^2/T]

δ = boundary layer film thickness [L]

α = surface area available for mass transfer between solid and liquid [L^2]

S = aqueous solubility of the chemical [M/L^3]

f_d = fraction of the total chemical in the dissolved phase [dimensionless]

C_c = total chemical concentration in the water column or porewater [M/L^3]

Assuming pure solid phase chemical particles are spherical, the surface area available for mass transfer can be expressed as a function of the pure solid phase chemical concentration, particle diameter, and particle density as (after Lynch et al. 2003):

$$\alpha = \frac{6 C_{cp} V}{d_s \rho_p} \tag{6.3}$$

where: C_{cp} =   pure solid phase chemical concentration (in particle form) [M/L^3]

              V =   bulk volume (water and particles) [L^3]

              d_p =   particle diameter [L]

              ρ_p =   particle density of pure solid phase chemical [M/L^3]

Following dissolution from the pure solid phase, dissolved chemicals are available for mass transfer (i.e. sorption) or transformation. Under equilibrium conditions, the maximum dissolved phase concentration a chemical can attain is the solubility limit. The solubility of a chemical is influenced by several factors including temperature and the concentrations (activities) of other ions in solution. The chemical dissolution reaction pathway may be useful for more detailed simulation of metal precipitates or explosive chemical compounds such as TNT or RDX that can be present in a granular, pure solid form.

6.2 ENVIRONMENTAL CONDITIONS

The rates at which chemical mass transfer and transformation processes occur often change as environmental conditions vary. Basic hydrologic processes such as evapotranspiration and soil moisture can change and environmental conditions vary. To facilitate development of more detailed chemical fate process representations or long-term hydrological impacts on chemical transport and fate, provisions to add a series of spatially and temporally varying time functions to represent environmental conditions exist in the TREX framework. While further development is needed to fully activate these features, code exists to represent environmental conditions such as wind speed, air temperature, solar radiation, the concentration of dissolved organic carbon (DOC) or other binding agents, the fraction organic carbon or partitioning effectiveness of both dissolved and particulate sorbents for partitioning, hardness, pH, water temperature, the concentration of oxidants, the concentration (population density) of bacteria, and light extinction properties. The developmental code for environmental conditions is organized to permit representation of any number or type of environmental properties as needed. For example, additional properties such as relative humidity or the atmospheric concentration of a chemical can be readily added within the existing framework structure.

7.0 PROGRAMMING GUIDE

7.1 TREX PROGRAMMING OVERVIEW

TREX has a modular design to support future expansions and enhancements. With a basic degree of familiarity, users can customize most functions of the frameworks to create highly specialized application to address a broad range of watershed hydrological, sediment transport, and chemical transport and fate issues. However, some caution is appropriate. TREX is a complex program. No guarantees are made regarding either the suitability of TREX for constructing a specific watershed model application or the computational performance of the code. Although substantial efforts have been taken to examine all aspects of the code, users are strongly advised to carefully examine the TREX source code and all model inputs and outputs to ensure proper operation.

The authors would appreciate receiving notification of any comments, potential programming errors, bug reports, other problems encountered with the TREX program, or desired enhancements. Notification may be sent by standard mail, telephone, or email to:

Mark Velleux	John F. England, Jr.	Dr. Pierre Julien
HDR Engineering, Inc.	U.S. Army Corps of Engineers	Department of Civil Engineering
1 International Boulevard	Risk Management Center	Colorado State University
10th Floor, Suite 1000	12596 West Bayaud Ave, Suite 400	B205 Engineering Research Center
Mahwah, NJ 07495	Lakewood, CO 80228	Fort Collins, CO 80523
(201) 335-5151	(303) 963-4524	(970) 491-8450
Mark.Velleux@hdrinc.com	john.f.england@usace.army.mil	pierre@engr.colostate.edu

7.2 TREX AVAILABILITY, ONLINE RESOURCES, AND SUPPORT

The TREX source code, some example input and output files, and user's manual are available for download from Dr. Julien's web site at Colorado State University:

http://www.engr.colostate.edu/%7Epierre/ce_old/Projects/TREX%20Web%20Pages/TREX-Home.html

The web site also includes references to reports and journal articles on TREX and applications with TREX. Velleux et al. (2008) present an overview of TREX. A metals transport study using TREX is described in Velleux (2005) and Velleux et al. (2006). Extreme flood modeling of the Arkansas River with TREX is presented in England (2006) and England et al. (2007).

The authors may periodically update the TREX distribution bundle on this web site as development continues. Example application files (test cases) are also included. The current examples include hydrology and sediment transport for the October 17, 1981 storm event on the Goodwin Creek, Mississippi watershed, a hypothetical chemical transport example based on the Goodwin Creek setting, and demonstrations of extreme storm rainfall options applied to the Arkansas River, Colorado. The hydrology and sediment transport examples for the Goodwin Creek site are of note because they allow comparison to results obtained using CASC2D-SED described by Johnson et al. (2000), Julien and Rojas (2002), and Rojas et al. (2008).

Please note that TREX was developed for academic and research use. TREX is not officially supported by any of the organizations with which the authors are affiliated. This means that users should not expect support beyond receiving the program, example files, and user's manual via the web site unless special arrangements are made with the authors.

7.3   TREX PROGRAM ORGANIZATION AND DESCRIPTION

TREX is a modular program and is comprised of subprocess groups that are organized into functional units for input, initialization, time function update, transport process, integration, output, and deallocation. The main program (trex) calls processes to:

1. Read inputs;

2. Initialize variables;

3. Compute hydrologic (water) transport, sediment (solids) transport, and chemical transport from the simulation iterative time loop;

4. Compute hydrologic (water) mass balance, sediment (solids) mass balance, and chemical mass balance from the simulation iterative time loop;

5. Write outputs; and

6. Deallocate memory and end execution.

The transport process functional units for hydrologic transport, sediment transport, and chemical transport and fate are the core of TREX. These units appear within the model iterative time loop of TREX. The organization of the transport process functional units is presented in Figure 7-1. Descriptions of the main components of each functional unit within TREX follow. Descriptions of data included in global header files also follows.

7.3.1   *Input Functional Unit*

The input functional unit reads user-specified data from the main model input file, as organized into Data Groups A-F, as well as any ancillary input files required for a simulation. The major modules within this functional unit are:
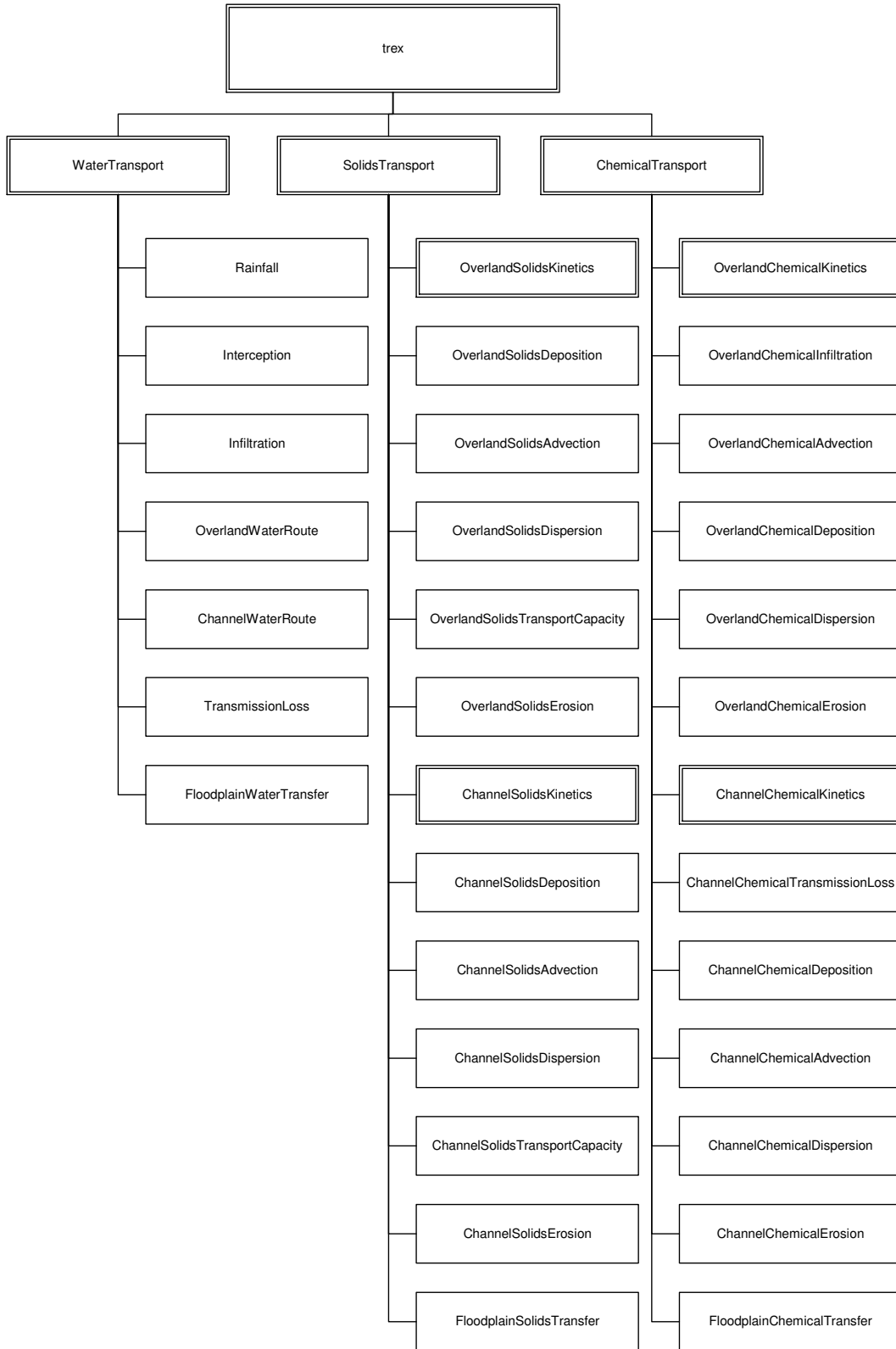
Figure 7-1. Organization of transport process functional units in TREX.

- ReadInputFile
- ReadDataGroupA
- ReadDataGroupB
- ReadDataGroupC
- ReadDataGroupD
- ReadDataGroupE
- ReadDataGroupF

At the time TREX is executed, the user must specify the name of the main model input file (and may also specify a restart option). The main model input file is specified as an argument to the program as described in Section 3.5. The main input file name is stored in global memory and TREX calls ReadInputFile to initiate data input processing operations. As TREX reads Data Groups A-F, a series of utility and secondary modules are called to read any required ancillary input files. The name assigned to each of these secondary modules provides a shorthand description of the module function and use. For example, ReadMaskFile is called to read the row-column grid format file that defines the spatial domain of a simulation. The model echo file is produced as output by the this unit.

7.3.2 *Initialization Functional Unit*

The initialization functional unit allocates memory and assigns initial values to variables needed for simulation. The modules within this functional unit, organized by the transport process (water, solids, chemical) for which they initialize variables, are:

- Initialize
- InitializeWater
- InitializeSolids
- InitializeChemical
- InitializeEnvironment
- TimeFunctionInit
- TimeFunctionInitWater
- TimeFunctionInitSolids
- TimeFunctionInitChemical
- TimeFunctionInitEnvironment
- ComputeInitialState
- ComputeInitialStateWater
- ComputeInitialStateSolids
- ComputeInitialStateChemical

Once all model inputs are read, the initialization functional unit initializes all remaining primary and secondary variables that are not defined at the time the main model inputs are read. In addition, this functional unit also creates all basic output file types that will be used during the simulation. For example, the export, detailed mass balance, and summary statistics files are created by this functional unit.

7.3.3 Time Function Update Functional Unit

The time function update functional unit assigns values in time for each time function specified by the user. The modules within this functional unit, organized by transport process (water, solids, chemical), are:

- UpdateTimeFunction
- UpdateTimeFunctionWater
- UpdateTimeFunctionSolids
- UpdateTimeFunctionChemical
- UpdateEnvironment

This functional unit is called from the simulation iterative time loop within TREX. User-specified time functions such as the rainfall intensity (rate) as a gage location, the flow from a point source of water, or the load from a chemical point source are updated for use each time step during the simulation. Values in time are updated between times defined by user input using linear interpolation. Outputs from this functional unit may be subsequently interpolated over space or applied at a point as needed by the modules comprising the transport functional unit.

7.3.4 Transport Process Functional Unit

The transport process functional unit computes rates and fluxes for each hydrological, sediment transport, and chemical transport process specified by the user. The modules within this functional unit, organized by transport process (water, solids, chemical), are:

- WaterTransport
- Rainfall
- Interception
- Infiltration, TransmissionLoss
- OverlandWaterRoute, ChannelWaterRoute
- FloodplainWaterTransfer
- SolidsTransport
- OverlandSolidsKinetics, ChannelSolidsKinetics
- OverlandSolidsDeposition, ChannelSolidsDeposition

- OverlandSolidsAdvection, ChannelSolidsAdvection

- OverlandSolidsDispersion, ChannelSolidsDispersion

- OverlandSolidsTransportCapacity, ChannelSolidsTransportCapacity

- OverlandSolidsErosion, ChannelSolidsErosion

- FloodplainSolidsTransfer

- ChemicalTransport

- OverlandChemicalKinetics, ChannelChemicalKinetics

- OverlandChemicalInfiltration, ChannelChemicalInfiltration

- OverlandChemicalDeposition, ChannelChemicalDeposition

- OverlandChemicalAdvection, ChannelChemicalAdvection

- OverlandChemicalDispersion, ChannelChemicalDispersion

- OverlandChemicalErosion, ChannelChemicalErosion

- FloodplainChemicalTransfer

This functional unit is called from the simulation iterative time loop within TREX. Transport rates and fluxes for each process for each state variable in each overland cell and each channel node are computed for the surface water as well as all layers in the soil and sediment column for each time step during the simulation. Further details regarding the organization of the kinetics (mass transfer and transformation reactions) modules for chemical fate are presented in Figure 7-2. the chemical kinetics subunits include process modules for partitioning, biodegradation, hydrolysis, oxidation, photolysis, radioactive decay, a user-defined reaction, volatilization, dissolution, and transformation yields between chemical state variables.

7.3.5 Integration Functional Unit

The integration functional unit performs numerical integration to compute values for each state variable over time using the rates and fluxes for each hydrological, sediment transport, and chemical transport process computed by the transport process and time function update functional units. The modules within this functional unit, organized by transport process (water, solids, chemical), are:

- WaterBalance

- OverlandWaterDepth

- ChannelWaterDepth

- SolidsBalance

- OverlandSolidsConcentration

- ChannelSolidsConcentration

Figure 7-2. Organization of chemical kinetics process modules within TREX.

- SolidsBalance

- OverlandSolidsConcentration

- ChannelSolidsConcentration

- ChemicalBalance

- OverlandChemicalConcentration

- ChannelChemicalConcentration

- NewState

- NewStateWater

- NewStateSolids

- NewStateChemical

- NewStateStack

This functional unit is called from the simulation iterative time loop within TREX. The model state variables are water depth, solids concentration, and chemical concentration. Mass balances are performed by summing the volume and mass fluxes for each state variable to computing new values for depth or concentration. The new state of the domain (water depths, solids concentrations, chemical concentrations, as well as soil or sediment stack volumes) is then stored. At the end of the simulation iterative time loop, utility functions to determine maximum and minimum depths and concentrations are called, the new domain state is set as the current state, and then the simulated time is advanced one time step.

7.3.6 *Output Functional Unit*

The output functional unit writes program results to a range of user-specified output files. The modules within this functional unit are:

- WriteTimeSeries

- WriteTimeSeriesWater, WriteTimeSeriesSolids, WriteTimeSeriesChemical

- WriteGrids

- WriteGridsWater, WriteGridsSolids, WriteGridsChemical

- WriteDumpFile

- WriteEndGrids

- WriteEndGridsWater, WriteEndGridsSolids, WriteEndGridsChemical

- ComputeFinalState

- ComputeFinalStateWater, ComputeFinalStateSolids, ComputeFinalStateChemical

- WriteMassBalance

- WriteMassBalanceWater, WriteMassBalanceSolids, WriteMassBalanceChemical

- WriteSummary

- WriteSummaryWater, WriteSummarySolids, WriteSummaryChemical

With the exception of the WriteEndGrid series of modules, this functional unit is called from the simulation iterative time loop within TREX. The WriteEndGrids modules are once at the end of the simulation immediately following the simulation iterative time loop. Each time these modules are called, user-specified output is written to file. The WriteTimesSeries modules write to files that hold comma separated values written line by line to form a sequence of model results at specified points in space organized by time. For example, one possible time series output is the water depth or total suspended solids concentrations at a reporting station. The WriteGrid modules write to grid (row-column) output files that hold values for all points in space for a single time in a format. Grid outputs are written in sequence at a user-specified frequency. For example, one possible sequence of grid output is the water depths over the model domain at simulation times of the simulation start (time = 0), 10 minutes, 20 minutes, etc. The WriteEndGrid modules write to files similar to regular grid outputs except that a single grid that holds the difference between start and end conditions over the model domain for the entire simulation is written. The ComputeFinalState modules compute conditions for primary and secondary model variables as needed to prepare detailed mass balance and summary statistics reports. The WriteMassBalance modules write to a single file that holds a detailed mass balance for all state variables. The WriteSummary nodules write to a single file that holds summary detail and simulation statistics for all state variables.

7.3.7   *Deallocation Functional Unit*

The deallocation functional unit performs end of simulation memory deallocation simulation clean-up tasks. The modules within this functional unit are:

- FreeMemory

- FreeMemoryWater

- FreeMemorySolids

- FreeMemoryChemical

- FreeMemoryEnvironment

- SimulationError

- RunTime

The FreeMemory series of modules of functional unit are called from SimulationError when an trapped error condition occurs or following the simulation iterative time loop within TREX. These modules deallocate memory allocated for primary and secondary variables. At the end of a successful simulation, RunTime is called to determine the wall clock time (not CPU time) elapsed from the start to the end of the simulation.

7.3.8 Global Declaration and Definition Header Files

TREX is designed to operate around large, globally available (common) data blocks. The categories of global data blocks, organized by transport function, are:

- trex_general_declarations, trex_general_definitions

- trex_water_declarations, trex_water_definitions

- trex_solids_declarations, trex_solids_definitions

- trex_chemical_declarations, trex_chemical_definitions

- trex_environmental_declarations, trex_environmental_definitions

Each category of global data is used by a specific layer of the framework. The framework layers are: general model controls, hydrology, sediment transport, and chemical transport. In TREX, information is always (and only) passed forward from one layer to the next.

Declarations files specify included C library files (e.g. stdio.h, math.h, etc.), function prototypes, macros, and external file pointers and variable declarations. Definitions files provide definitions for all externally declared file pointers and variables. General declarations specify information regarding general model controls. Water declarations specify information regarding hydrology. Solids declarations specify information regarding sediment transport. Chemical declarations specify information regarding chemical transport and fate. Environmental declarations specify information regarding environmental conditions.

To make information available to a layer, the global declarations file for that layer must be included in all modules of the layer. The general declarations are common to, and used by, all program layers. The hydrology layer uses the general and water declarations. The sediment transport layer uses the general, water, and solids declarations. The chemical transport layer uses the general, water, solids, chemical, and environmental declarations.

Note that for the present state of development, environmental conditions only affect chemical transport and fate. As environmental condition feature developments continue, the environmental category may be divided into separate groups for hydrology, sediment transport, and chemical transport. For example, wind speed, air temperature, and relative humidity functions might be used by the hydrology layer to compute evapotranspiration and evaporation of surface water while soil temperature might be used by the sediment transport layer to compute mineralization rates of organic particles.

It is also important to note that information is never passed backwards between layers. For example, the hydrology layer does not and should not have access to global data for sediment or chemical transport. Similarly, sediment transport does not and should not have access to global data for chemical transport and fate. This layered data access and management approach is useful for maintaining framework modularity. Future layers, such as nutrient transport or eutrophication, can be readily added without establishing confusing crosslinks between layers.

7.4   PROGRAMMING STYLE

To enhance readability, comprehension, and facilitate future development, a consistent programming style has been used for all TREX code development. The authors believe continued use of a consistent programming style is critical for future code maintenance and development. Descriptions of the key conventions of the TREX programming style follow.

7.4.1   Naming Conventions

All variable names in the code are lower case (e.g. nsolids, advsedchoutflux, etc.). All programs module names are a mixture of upper and lower case (e.g. WaterTransport, ChannelChemicalKinetics, etc.). All macros are upper case (e.g. MAXNAMESIZE, TOLERANCE, etc.). Variable names are descriptive. As a short hand, variables for the overland plane include "ov" and those for the channel network include "ch". Variables for water depth have "h" and flow have "q" in their names. Variables for sediment transport variables include "sed" and those for chemical transport include "chem". Flux and mass terms include "flux" and "mass", respectively. Transport process variables also include three character identifiers to denote the specific process with which the variable is associated: "adv" for advection, "dsp" for dispersion, "dep" for deposition, "ers" for erosion, or "dsl" for dissolution, etc. In addition to identifiers such as "ov", "adv", and "flux", transport variables also include "out" or "in" to identify the direction of transport. Each of these name elements are typically concatenated to form the full variable name. For example, the flux of suspended solids transported out of an overland cell by advection would be "advsedovoutflux". Through use of a consistent naming convention, new state variables and process modules can be added to the framework by using existing modules as templates and using a simple search and replace to include the names of new variables.

7.4.2   Internal Documentation and Comments

The TREX code is extensively documented. Every module includes initial comments that identify the module name, purpose and methods, inputs, outputs, controls, modules called, calling module(s), routine author(s), revision history (if any), and date. Further, virtually every line of code has a comment to explain the line-by-line operation of the module. Wherever additional information is needed to explain the basis of an operation, a comment block delimited by the string "Note:" occurs and is followed by more in-depth documentation. Comment blocks delimited by the string "Developer's Note:" identify areas that may be the subject of future framework development efforts.

The beginning, interior, and concluding brackets of loop and if structures each have comments to clearly identify the structures. Also, each unit of code within a loop or if structure is indented with tabs to provide further visual cues as to which structure controls the code. The start and end of all loop structures are identified the strings "loop" and "end loop", respectively. The start and end of all if structures are identified the strings "if" and "end if", respectively.

7.4.3 Maintaining Consistent Programming Style During Development

In many settings, it is common for a computer program code development team to change over time. This is particularly true in a setting such as a university where computer code is shared among different generations of students and projects. Under such conditions, it is common for code to rapidly mutate until it becomes so unintelligible that sometimes even the immediate code author(s) cannot clearly follow or explain its operation. The experience of the authors is that computer code handled on an informal or ad hoc basis by a changing array of developers quickly becomes unmanageable. To achieve a higher degree of long-term manageability and maintainability over time, the authors strongly recommend that future developers continue to use the programming style conventions employed during initial TREX development.

Continued adherence to naming conventions and use of extensive, line-by-line comments in the code is essential for future maintainability. In addition to providing information regarding program operation, use of consistent comments and variable names also greatly facilitates program testing and debugging. Use of established programming conventions allows newly developed modules to be rapidly screened for the occurrence potential bugs such as incorrect variable name references. For example a common programming error is to reference variables for the channel network in a module for overland plane (or vice versa). Consider a solids transport module where variables csedch and advsedchoutflux have been incorrectly used instead of csedov and advsedovoutflux. By adhering to variable naming conventions, a module can be searched (case sensitive) for the occurrence of the string "sedch" since overland process variables use the string "sedov" and it is very rare for an overland module to ever reference conditions in the channel network. Similarly, the occurrence of the string "sedov" rather than "chemov" would be rare in chemical process modules.

Care should also be taken to prevent "cross-wiring" of program layers. The global data structure needed for any model layer (hydrology, sediment transport, etc.) should contain all variables needed for that layer to operate. In terms of program flow, information should only be passed forward. The global data structure for a later model layer should never be used or available to an earlier model layer.

It is worth reiterating that an important feature of the TREX framework is that all source code for the program is extensively documented. The importance of comprehensive, line-by-line program documentation for future code maintenance and development cannot be understated. All too often during development efforts, internal code documentation is neglected when programmers add code without adding corresponding, detailed comments for the perceived "speed and ease" of development. This inevitably leads to generation of poorly written, undocumented code. Even if the original authors of undocumented code are available and can still decipher it in the future, it becomes exceedingly difficult for other programmers to manage such code over time. The challenges of managing poorly written and poorly documented code are often difficult to overcome and can be costly in terms of lengthened development time cycles. The consequence of earlier shortcuts taken for "speed and ease" is often that the same body of code ends up being repeatedly redeveloped by subsequent generations of developers. It the further experience of the

authors that code that cannot be fully documented at the time it is first written is in many instances poorly conceived, often poorly structured, and typically leads to effort wasted redeveloping code. For these and many other reasons, the importance of maintaining complete and comprehensive internal documentation of all program code cannot be overemphasized.

8.0 REFERENCES

Abdullrazzak, M., and Morel-Seytoux, H. 1983. Recharge from an ephemeral stream following wetting front arrival to water table. Water Resources Research, 19(1):194-200.

Abramowitz, M. and Stegun, I.A. 1972. Handbook of Mathematical Functions. Applied Mathematics Series 55. National Bureau of Standards, Washington, D.C.

Aksoy, H., and Kavvas, M.L. 2005. A review of hillslope and watershed scale erosion and sediment transport models. Catena, 64(2-3):247-271.

Al-Hamdan, O.Z., Pierson, F.B., Nearing, M.A., Stone, J.J., Williams, C.J., Kormos, P.R., Boll, J., and Weltz, M.A. 2011, Shear stress partitioning of overland flow on disturbed and undisturbed rangelands, Paper 11134, paper presented at International Proceedings of the Symposium on Erosion and Landscape Evolution, September 18–21, 2011, Anchorage, Alaska, USA, American Society of Agricultural and Biological Engineers and Association of Environmental and Engineering Geologist.

Al-Hamden, O.Z., Pierson, F.B., Nearing, M.A., Williams, C.J., Stone, J.J., Kormos, P.R., Boll, J. and Weltz, M.A. 2012. Concentrated flow erodibility for physically based erosion models: Temporal variability in disturbed and undisturbed rangelands. Water Resources Research, 48, W07504, doi:10.1029/2011WR011464.

Al-Hamden, O.Z., Pierson, F.B., Nearing, M.A., Williams, C.J., Stone, J.J., Kormos, P.R., Boll, J. and Weltz, M.A. 2013. Risk assessment of erosion from concentrated flow on rangelands using overland flow distribution and shear stress partitioning. American Society of Agricultural and Biological Engineers, 56(2)-539-548.

Ambrose, R.B., Martin, J.L. and Wool, T.A. 1993. WASP5, A hydrodynamic and water quality model -- Model theory, user's manual, and programmer's guide. U.S. Environmental Protection Agency, Office of Research and Development, Environmental Research Laboratory, Athens, Georgia.

Atkinson, J.F., Abrahams, A.D., Krishnan, C., Li, G. 2000. Shear stress partitioning and sediment transport by overland flow. Journal of Hydraulic Research, 38(1):37-40.

Beuselinck, L., Govers, G., Steegen, A., and Quine, T.A. 1999. Sediment transport by overland flow over an area of net deposition. Hydrological Processes, 13(17):2769-2782.

Burban, P.Y., Xu, Y., McNeil, J., and W. Lick. 1990. Settling speeds of flocs in fresh and sea waters. Journal of Geophysical Research (C) Oceans, 95(C10):18213-18220.

Capel, P. and S. Eisenriech, S. 1990.Relationship between chlorinated hydrocarbons and organic carbon in sediment and porewater. Journal of Great Lakes Research, 16(2):245-257.

Chapra, S.C. 1997. Surface Water-Quality Modeling. McGraw-Hill Companies, Inc. New York, New York. 844 pp.

Chapra, S.C., and Canale, R.P. 1985. Numerical Methods for Engineers with Personal Computer Applications (First Edition). McGraw-Hill, Inc., New York, New York. 570 pp.

Cheng, N.S. 1997. Simplified settling velocity formula for sediment particle. Journal of Hydraulic Engineering, 123(2):149-152.

Cussler, E. L. 1997. Diffusion: mass transfer in fluid systems (Second E, 2nd edition. Cambridge University Press, New York, New York. 580 p.

Dingman, S.L. 2002. Physical Hydrology (Second Edition). Prentice Hall, Inc., Upper Saddle River, New Jersey. 646 p.

Di Toro, D.M. 1985. A particle interaction model of reversible organic chemical sorption. Chemosphere, 14(9-10):1503-1538.

Eadie, B., N. Morehead, and P. Landrum. 1990. Three-phase partitioning of hydrophobic organic compounds in Great Lakes waters. Chemosphere, 20(1-2):161-178.

Eadie, B., N. Morehead, V. Klump, and P. Landrum. 1992. Distribution of hydrophobic organic compounds between dissolved and particulate organic matter in Green Bay waters. Journal of Great Lakes Research, 18(1):91-97.

Eagleson, P.S. 1970. Dynamic Hydrology. McGraw-Hill Book Company, New York, New York. 462 pp.

Einstein, H.A., and Banks, R.B. 1950. Fluid resistance of composite roughness. Transactions of the American Geophysical Union, 31(4):603-610.

Eliasson, I., and Svensson, M.K. 2003. Spatial air temperature variations and urban land use – a statistical approach. Meteorological Applications, 10(2):135-149.

Engelund, F., and Hansen, E. 1967. A monograph on sediment transport in alluvial streams. Teknisk Vorlag, Copenhagen, Denmark. 62 pp.

England, J.F. Jr. 2006. Frequency Analysis and Two-Dimensional Simulations of Extreme Floods on a Large Watershed. Ph.D. dissertation, Department of Civil Engineering, Colorado State University, Fort Collins, Colorado. 237 p.

England, J.F. Jr., Velleux, M.L. and Julien, P.Y. 2007. Two-dimensional simulations of extreme floods on a large watershed. Journal of Hydrology, 347(1-2):229-241 (doi:10.1016/j.jhydrol.2007.09.034).

Ewen, J., Parkin, G., and O'Connell, P.E. 2000. SHETRAN: distributed river basin flow and transport modeling system. Journal of Hydrologic Engineering, 5(3):250-258.

Exner, F.M. 1925, Über die wechselwirkung zwischen wasser und geschiebe in flüssen, Sitzungber. Acad. Wissenscaften Wien Math. Naturwiss. Abt. 2a, 134:165–180.

Fetter, C.W. 2001. Applied Hydrogeology, Fourth Edition. Prentice-Hall, Inc. Upper Saddle River, New Jersey. 598 p.

Freyberg, D. 1983. Modeling the effects of time-dependent wetted perimeter on infiltration from ephemeral channels. Water Resources Research, 19(2):559-566.

Gessler, J. 1965. The Beginning of Bedload Movement of Mixtures Investigated as Natural Armouring in Channels. Technical report No. 69, The Laboratory of Hydraulic Research and Soil Mechanics, Swiss Federal Institute of Technology, Zurich (translation by W. M. Keck Laboratory of Hydraulics and Water Resources, California Institute of Technology, Pasadena, California).

Gessler, J. 1967. The beginning of bedload movement of mixtures investigated as natural armoring in channels. California Institute of Technology, Pasadena, California. 89pp.

Gessler, J. 1971. Beginning and ceasing of sediment motion. In: River Mechanics, Shen, H.W., ed. H.W. Shen, Fort Collins, Colorado. pp. 7:1–7:22.

Gray, D. M., and Prowse, T. D. 1993. "Chapter 7: Snow and floating ice." In: Handbook of Hydrology, Maidment, D.R., ed. McGraw-Hill, Inc., New York, New York.

Green, W.H. and Ampt, G.A. 1911. Studies on soil physics, 1: the flow of air and water through soils. Journal of Agricultural Sciences 4(1):11-24.

Haralampides, K., McCourquodale, J.A., Krishnappan, B.G. 2003. Deposition properties of fine sediment. Journal of Hydraulic Engineering, 129(3):230-234.

Harbaugh, A.W., Banta, E.R., Hill, M.C., and McDonald, M.G. 2000. MODFLOW-2000, the U.S. Geological Survey modular ground-water model -- User guide to modularization concepts and the Ground-Water Flow Process. U.S. Geological survey, Denver, Colorado. Open-File Report 00-92. 121 p.

Holley, E.R. 1969. Unified view of diffusion and dispersion. Journal of the Hydraulics Division, American Society of Civil Engineers, 95(2):621-631.

Holm, P.E., Rootzén, H., Borggaard, O.K., Møberg, J.P., and Christensen, T.H. 2003. Correlation of cadmium distribution coefficients to soil characteristics. Journal of Environmental Quality, 32(1):138-145.

Imhoff, J.C., Stoddard, A., Buchak, E.M., and Hayter, E. 2003. Evaluation of Contaminated Sediment Fate and Transport Models: Final Report. U.S. Environmental Protection Agency, Office of Research and Development, National Exposure Research Laboratory, Athens, Georgia. Contract Number 68-C-01-037, Work Assignment No. 1-10. 153 p.

Johnson, B.E., Julien, P.Y., Molnár, D.K., and Watson, C.C. 2000. The two-dimensional upland erosion model CASC2D-SED. Journal of the American Water Resources Association, 36(1):31-42.

Julien, P.Y. 1998. Erosion and Sedimentation (First Paperback Edition). Cambridge University Press, Cambridge, UK. 280 p.

Julien, P.Y. 2002. River Mechanics. Cambridge University Press, Cambridge, UK. 434 p.

Julien, P.Y. 2010. Erosion and Sedimentation (Second Edition). Cambridge University Press, Cambridge, UK. 371 p.

Julien, P.Y. and Saghafian, B. 1991. CASC2D User's Manual - A Two Dimensional Watershed Rainfall-Runoff Model. Department of Civil Engineering, Colorado State University, Fort Collins, Colorado. Report CER90-91PYJ-BS-12. 66 p.

Julien, P.Y., Saghafian, B., and Ogden, F.L. 1995. Raster-Based hydrologic modeling of spatially-varied surface runoff. Water Resources Bulletin, AWRA, 31(3):523-536.

Julien, P.Y. and Rojas, R. 2002. Upland erosion modeling with CASC2D-SED. International. Journal of Sediment Research, 17(4):265-274.

Julien, P.Y., and Frenette, M. 1985. Modeling of rainfall erosion. Journal of Hydraulic Engineering, 11(10):1344-1359.

Julien, P.Y., and Simons, D.B. 1985. Sediment transport capacity of overland flow. Transactions of the American Society of Agricultural Engineers, 28(3):755-762.

Karickhoff, S.W., Brown, D.S., and Scott, T.A. 1979. Sorption of hydrophobic pollutants on natural sediments. Water Research, 13(3):241-248.

Kashian, D.R., B. Prusha and W. H. Clements. 2004. The influence of dissolved organic matter and UV-B radiation on zinc toxicity: effects on aquatic communities. Environmental Science and Technology. (Invited Paper). In Press.

Kilinc, M.Y., and Richardson, E.V. 1973. Mechanics of soil erosion from overland flow generated by simulated rainfall. Hydrology Papers Number 63. Colorado State University, Fort Collins, Colorado.

Kinnell, P.I.A. 2010. Event soil loss, runoff and the Universal Soil Loss Equation family of models: A Review. Journal of Hydrology, 385(1-4):384-397.

Kipp, K.L, Jr. 1997. Guide to the Revised Heat and Solute Transport Simulator: HST3D Version 2. U.S. Geological Survey, Denver, Colorado. Water Resources Investigations Report 97-4157. 149 p.

Krishnappan, B.G. 2000. In situ distribution of suspended particles in the Frasier River. Journal of Hydraulic Engineering, 126(8):561-569.

Krone, R.B. 1962. Flume studies of the transport of sediments in estuarial shoaling processes. Final Report. Hydraulic Engineering Laboratory and Sanitary Engineering Research Laboratory, University of California, Berkeley, California.

Landrum, P., M. Rheinhold, S. Nihart, and B. Eadie. 1985. Predicting bioavailability of xenobiotics to Pontoporeia Hoya in the presence of humic and fulvic materials and natural dissolved organic matter. Environmental Toxicology and Chemistry, 4(4):459-467.

Landrum, P., S. Nihart, B. Eadie, and Herche, L. 1987. Reduction in bioavailability of organic contaminants to the amphipod Pontoporeia Hoya by dissolved organic matter of sediment interstitial waters. Environmental Toxicology and Chemistry, 6(1):11-20.

Lane, L.J. 1983. Chapter 19: Transmission Losses. In: Soil Conservation Service National Engineering Handbook, Section 4: Hydrology, U.S. Government Printing Office, Washington, D.C. pp.19-1–19-21

Li, R.M., Stevens, M.A., and Simons, D.B. 1976. Solutions to Green-Ampt infiltration equations. Journal of Irrigation and Drainage Division, ASCE, 102(IR2):239-248.

Linsley, R.K., Kohler, M.A., and Paulhus, J.L.H. 1982. Hydrology for Engineers (Third Edition). McGraw-Hill Book Company, New York, New York. 508 p.

Liston, G.E., and Elder, K. 2006. A meteorological distribution system for high-resolution terrestrial modeling (MicroMet). Journal of Hydrometeorology, 7(2):217-234.

Lu, Y., and Allen, H. 2001. Partitioning of copper onto suspended particulate matter in river waters. Science of the Total Environment 277(1-3):119-132.

Lynch, J., Brannon, J., Hatfield, K., and Delfino, J. 2003. An exploratory approach to modeling explosive coumpond persistence and flux using dissolution kinetics. Journal of Contaminant Hydrology, 66(3-4):147-153.

Mehta, A., McAnally, W., Hayter, E., Teeter, A., Heltzel, S., and Carey, W. 1989. Cohesive sediment transport. II: application. Journal of Hydraulic Engineering, 115(8):1094-1112.

Merritt, W.S., Letcher, R.A., and Jakeman. A.J. 2003. A review of erosion and sediment transport models. Environmental Modelling and Software, 18(8-9):761-799.

Meyer, L.D., and Weischmeier, W.H. 1969. Mathematical simulation of the process of soil erosion by water. Transactions of the American Society of Agricultural Engineers, 12(6):754-762.

Partheniades, E. 1992. Estuarine sediment dynamics and shoaling processes. In: Handbook of Coastal and Ocean Engineering, Volume 3: Harbours, Navigation Channels, Estuaries, and Environmental Effects, pp. 985-1071. Herbich, J. B., Ed. Gulf Publishing Company, Houston, Texas.

Pellicciotti, F., Brock, B., Strasser, U., Burlando, P., Funk, M., and Corripio, J. 2005. An enhanced temperature-index glacier melt model including the shortwave radiation balance: development and testing for Haut Glacier d'Arolla, Switzerland. Journal of Glaciology, 51(175):573-587.

Phillip, J.R. 1957a. The theory of infiltration: 1. The infiltration equation and its solution. Soil Science, 83:345-357.

Prosser, I.P., Dietrich, W.E., and Stevenson, J. 1995. Flow resistance and sediment transport by concentrated overland flow in a grassland valley. Geomorphology, 13(1):71-86.

Prosser, I.P., and Rustomji, P. 2000. Sediment transport capacity relations for overland flow. Progress in Physical Geography 24(2):179-193.

QEA. 1999. PCBs in the Upper Hudson River, Volume 2: A model of PCB Fate, Transport, and Bioaccumulation. Prepared for General Electric, Albany, New York. Prepared by Quantitative Environmental Analysis, LLC, Montvale, New Jersey. Job Number GENhud:131. May.

Rauws, G. 1988. Laboratory experiments on resistance to overland flow due to composite roughness. Journal of Hydrology, 103(1-2):37-52.

Rawls, W.J., Brakensiek, D.L., and Miller, N. 1983. Green-Ampt infiltration parameters from soils data. Journal of Hydraulic Engineering, 109(1):62-69.

Rawls, W.J, Ahuja, L.R., Brakensiek, D.L., and Shirmohammadi, A. 1993. "Infiltration and Soil Movement." In: Handbook of Hydrology, Maidment, D.R., ed. McGraw-Hill, Inc., New York, New York. pp 5.1-5.51.

Renard, K.G., Foster, G.R., Weesies, G.A., and Porter, J.P. 1991. RUSLE: revised universal soil loss equation. Journal of Soil and Water Conservation, 46(1):30-33.

Renard, K.G., Foster, G.R., Yoder, D.C., McCool, D.K., 1994. RUSLE revisited: status, questions, answers, and the future. Journal of Soil and Water Conservation, 49(3):213-220.

Renard, K.G., Foster, G.R., Weesies, G.A., McCool, D.K., and Yoder, D.C. 1997. Predicting Soil Erosion by Water: A Guide to Conservation Planning with the

Revised Universal Soil Loss Equation (RUSLE), Agricultural Handbook No. 703. U.S. Department of Agriculture, Washington, D.C. 407 p.

Richards, L.A. 1931. Capillary conduction of liquids in porous mediums. Physics, 1:318-333.

Rojas, R. 2002. GIS-based Upland Erosion Modeling, Geovisualization and Grid Size Effects on Erosion Simulations with CASC2D-SED. Ph.D. dissertation, Department of Civil Engineering, Colorado State University, Fort Collins, Colorado.

Rojas, R., Velleux, M., Julien, P., and Johnson, B. 2008. Grid scale effects on watershed soil erosion models. Journal of Hydrologic Engineering, 13(9):793-802.

Sauvé, S.F., Hendershot, W., and Allen, H.E. 2000. Solid-solution partitioning of metals in contaminated soils: dependence on pH, total metal burden, and organic matter. Environmental Science and Technology 34(7):1125-1131.

Sauvé, S.F., Manna, S., Turmel, M.C., Roy, A.G., and Courchesne, F. 2003. Solid-solution partitioning of Cd, Cu, Ni, Pb, and Zn in the organic horizons of a forest soil. Environmental Science and Technology 37(22):5191-5196.

Schnorbus, M., and Alila, Y. 2004. Generation of an Hourly Meteorological Time Series for an Alpine Basin in British Columbia for Use in Numerical Hydrologic Modeling. Journal of Hydrometeorology, 5(5):862-882.

Schwarzenbach, R.P., Geschwend, P.M., and Imboden, D.M. 1993. Environmental Organic Chemistry. Wiley-Interscience, New York.

Simons, D.B., and Sentürk, F. 1992. Sediment Transport Technology – Water and Sediment Dynamics (Revised Edition). Water Resources Publications, Littleton, Colorado.

Smith, R.E., and Parlange, J.-Y. 1978. A parameter efficient hydrologic infiltrations model. Water Resources Research, 14(3):533-538.

Svensson, M.K., and Eliasson, I. 2002. Diurnal air temperatures in built-up areas in relation to urban planning. Landscape and Urban Planning, 61(1):37-54.

Thomann, R.V. and J.A. Meuller. 1987. Principles of Surface Water Quality Modeling and Control. Harper and Row Publishers, Inc., New York, New York. 644 pp.

van Rijn, L.C. 1984a. Sediment transport, part I: bed load transport. Journal of Hydraulic Engineering, 110(10):1431-1456.

van Rijn, L.C. 1984b. Sediment transport, part II: suspended load transport. Journal of Hydraulic Engineering, 110(11):1612-1638.

van Rijn, L.C. 1993. Principles of Sediment Transport in Rivers, Estuaries, and Coastal Seas. Aqua Publications, Amsterdam, The Netherlands.

van Rijn, L.C. 2007a. A unified view of sediment transport by currents and waves, I: Initiation of motion, bed roughness and bed load transport. Journal of Hydraulic Engineering, 133(6):649-667.

van Rijn, L.C. 2007b. A unified view of sediment transport by currents and waves, II: Suspended transport. Journal of Hydraulic Engineering, 133(6):668-689.

van Rijn, L.C. 2007c. A unified view of sediment transport by currents and waves, III: Graded beds. Journal of Hydraulic Engineering, 133(7):761-775.

Velleux, M. 2005. Spatially distributed model to assess watershed contaminant transport and fate. Ph.D. dissertation, Department of Civil Engineering, Colorado State University, Fort Collins, Coloardo. 261 p.

Velleux, M., Gailani, J., and Endicott, D. 1996. Screening-level approach for estimating contaminant export from tributaries. Journal of Environmental Engineering, 122(6):503-514.

Velleux, M., Westenbroek, S., Ruppel, J., Settles, M., and Endicott, D. 2001. A User's Guide to IPX, the In-Place Pollutant Export Water Quality Modeling Framework, Version 2.7.4. U.S. Environmental Protection Agency, Office of Research and Development, National Health and Environmental Effects Research Laboratory, Mid-Continent Ecology Division, Large Lakes Research Station, Grosse Ile, Michigan. 179 pp. EPA/600/R-01/079.

Velleux, M., England, J.F. Jr. and Julien, P.Y. 2008. TREX: Spatially distributed model to assess watershed contaminant transport and fate. Science of the Total Environment, 404 (1):113-128 (doi:10.1016/j.scitotenv.2008.05.053).

Velleux, M., Julien, P.Y., Rojas-Sanchez, R., Clements, W.H. and England, J. 2006. Simulation of metals transport and toxicity at a mine-impacted watershed: California Gulch, Colorado. Environmental Science and Technology, 40(22):6996-7004.

Williams, J.R. 1975. Sediment routing for agricultural watersheds. Water Resources Bulletin, 11(5):965-974.

Wischmeier, W.H., and Smith, D.D. 1978. Predicting Soil Erosion Losses: A Guide to Conservation Planning, Agricultural Handbook No. 537. U.S. Department of Agriculture, Washington, D.C. 58 pp.

Yang, C. T. 1996. Sediment Transport: Practice and Theory. McGraw-Hill, Inc. New York, New York. 480 pp.

Zheng, C. and Wang, P.P. 1999. MT3DMS, A modular three-dimensional multi-species transport model for simulation of advection, dispersion and chemical reactions of

contaminants in groundwater systems; documentation and user's guide. U.S. Army Engineer Research and Development Center Contract Report SERDP-99-1, Vicksburg, Mississippi. 202 p.