USDA-ARS, United States Salinity Laboratory

TETrans

Trace Element Transport Model

Research Report 121

Solute Transport Modeling Software
Macintosh Version 1.6

USER'S GUIDE

United States
Department of
Agriculture

Agricultural
Research
Service

TETrans

Version 1.6

Dennis Corwin, Ph.D.
Soil Scientist
(714) 369-4819

Bart Waggoner, M.S.
Programmer
(714) 369-4875

USDA-ARS
U.S. Salinity Laboratory
4500 Glenwood Drive
Riverside, California 92501
March, 1990

Table of Contents

Before You Start

TETrans (acronym for Trace Element Transport) is a software package that allows the user to simulate the vertical movement of nonvolatile inorganic and organic chemicals through the vadose zone under transient-state conditions. It is specifically designed to use a minimum of input parameters and to use input parameters which are easy to obtain relative to the input parameters of previous transport models. The software package consists of two components: the solute transport model and the user-friendly I/O (i.e., input/output) interface. The TETrans User's Guide centers its discussion around these two components.

The User's Guide will provide a conceptual understanding of the solute transport model component of the software so the model's strengths and weaknesses are more apparent. A detailed description of the I/O interface is necessary to provide the user-friendly tools for creating the input file(s) and for getting the desired output. The I/O interface is the shell of TETrans. It is through the I/O interface that the necessary input information is entered for the transport model component to create its simulations. Subsequently, the output from the transport model component is displayed by the I/O interface as text or graphics.

WHAT THIS PACKAGE CONTAINS

Please check to see that your TETrans software package includes the following items:

- ◊ TETrans application software - version 1.6 (TETrans)
- ◊ TETrans MPW Fortran source code of the solute transport model only - version 1.6 (TETran1_sub.f)
- ◊ Sample parameter file (Sample Parameters)
- ◊ User's Survey Questionaire
- ◊ Optional upon request: TETrans journal publications (2 articles from Journal of Environmental Quality)

WHAT YOU NEED

- ◊ Minimum requirements: MacPlus with 1 MB (i.e., megabyte) of RAM (i.e., random access memory)
- ◊ Preferable: Mac II or Mac SE/30 with 4-5 MB of RAM
- ◊ Macintosh System 6.0 or greater.

ABOUT THIS MANUAL

The User's Guide consists of 5 chapters:

- ◊ **Chapter 1: Installing TETrans** discusses the installation of the TETrans software.

- ◊ **Chapters 2: Introduction to TETrans** describes the solute transport model.

- ◊ Chapters 3 & 4 discuss the I/O interface which allows for the entry of parameter data and the selection of the type of output desired. **Chapter 3: Sample Simulation** provides a simulation using sample data provided in the Sample Parameters file. **Chapter 4: Using TETrans** goes through the use of the user-friendly Apple Macintosh standard pull-down and pop-up menus, and dialogue boxes for the entry and display of data.

- ◊ Finally, **Chapter 5: Troubleshooting TETrans** contains some useful hints in troubleshooting TETrans by providing some "preventative medicine" suggestions and corrective actions to error messages.

CHAPTER 1: INSTALLING TETrans

INSTALLING TETrans

Before you do anything else, make a backup copy of the TETrans disk. Refer to you Macintosh owner's guide if you need help. In addition, make sure you have the Macintosh system software 6.0 or greater installed on your hard disk (TETrans requires Macintosh System 6.0 or greater).

To install TETrans create a new folder on your hard disk and name it "TETrans". Copy all of the files from the TETrans disk into this new folder. You are now ready to run.

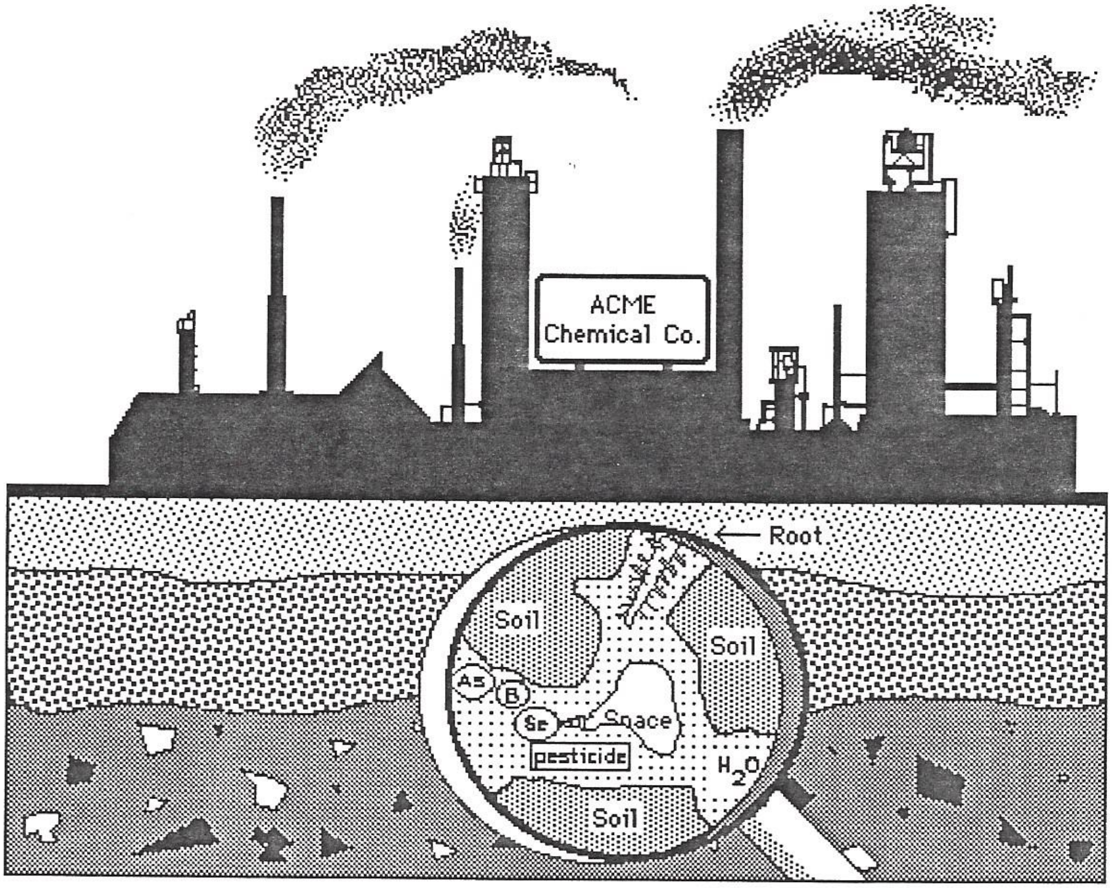
CHAPTER 2: INTRODUCTION TO TETrans

This chapter provides a general description of the functional solute transport model. It discusses the theory, preferential flow (i.e., hydraulic bypass) considerations, input parameter requirements, applications, strengths and weaknesses. Recently published literature which provides verification test data comparing measured data and simulated results for the movement of chloride and boron through a soil lysimeter column is presented in order to establish credibility for the transport model.

ABOUT TETrans

Groundwater is a major source of our drinking, industrial and agricultural water. Groundwater is becoming an even more important natural resource as surface water supplies are depleted and/or polluted.

The growing concern over health affects resulting from contaminated drinking water has brought the degradation of our groundwater to the forefront of public attention. Groundwater degradation is also an agricultural concern because of the decrease in crop productivity which can accompany the use of poor quality irrigation water. The ability to model the migration of pollutants through the vadose zone is an essential tool in combating the degradation of our groundwater.



Over the past three decades numerous conceptual models for the movement of solutes through the unsaturated zone have been developed. Several reviews of these transport models have recently appeared in the literature (Addiscott and Wagenet, 1985; Nielsen et al., 1986; van Genuchten and Jury, 1987; Engesgaard and Christensen, 1988; Feddes et al., 1988). Basically two groups of transport models are recognized: deterministic and stochastic. Within these two categories there are additional subcategories of models. Functional models are a group of deterministic models which utilize simplified treatments of solute and water flow while making no claim to a fundamental description of the mechanisms involved in the transport process. Functional models require less input data and computer expertise for their application. As such, functional models are management-oriented models. Several functional models have been presented in the literature (Bressler, 1967; Tanji et al., 1972; Burns, 1975; Burns, 1976; Addiscott, 1977; Rose et al., 1982; Bond and Smiles, 1988).

More than ever before, there is a need for simple, management-oriented models for interpreting and simulating solute movement by leaching. This need for functional transport models arises from two limitations found in more theoretically rigorous mechanistic models of transport. First, the soil data needed for sophisticated analytical and numerical models are typically well beyond the capacity of most real-world users, such as the EPA, Soil Conservation Service or Agricultural Extension Service. Second, the spatial variability typical of field soils limits the accuracy of application of exact transport theory under field management situations. As pointed out by Bond and Smiles (1988), "... the assumptions used to derive most flow equations presented in the literature are not satisfied in field soils, and analytical solutions of these equations are appropriate only to a very restricted set of initial and boundary conditions." Stochastic transport models do not provide a viable alternative for most real-world applications since, again, the data upon which they are derived are too labor and cost intensive to be practical.

In many well-drained soils, water movement responsible for solute transport can be approximately and simply calculated using water-balance accounting and a knowledge of field capacity. Using this simplified approach, no quantitative knowledge of the soil's hydraulic conductivity or of its moisture retention curves is required. Previous models have been developed on this premise by Burns (1975), Rose et al. (1982), and Bond and Smiles (1988). However, in each of these models it was assumed that no bypass flow of water and solute through large pores had occurred. In aggregated soil or soil high in clay, bypass flow paths through macropores and cracks has a definite effect on the flow of solutes. In this case, flow can deviate significantly from the near piston-type flow exhibited in some well-drained soils. In addition, soil water may actually be composed of two phases: an immobile water phase and a mobile water phase (Turner, 1958; Coats and Smith, 1964; Deans, 1964). It is only the mobile water phase which is miscibly displaced by incoming precipitation or irrigation water. The immobile water phase is bypassed.

TETrans (acronym for Trace Element Transport) was developed on the premise of mass balance applied by Burns (1975) and on a consideration of bypass flow. TETrans is a simplified mathematical model of one-dimensional transport which simulates the vertical movement of trace elements and nonvolatile organic and inorganic chemicals through the vadose zone. The model is specifically designed for real-world applications where little transport information is known by or is available to the user. The model utilizes a mass-balance approach to determine solution and adsorbed (or exchangeable) solute concentration distributions. Several modeling options are offered for simulating such transport-influencing factors as plant water uptake, hydraulic bypass, exchange and adsorption.

THEORY

TETrans is a capacity model which defines changes in amounts of solute and water content rather than rates of change. As such, it is driven by the amounts of rainfall, irrigation or evapotranspiration (ET) and only considers time indirectly by using the time from one irrigation or precipitation event to another. From a knowledge of water inputs and losses, and of soil-solute chemical interactions, TETrans predicts the average movement of reactive and nonreactive solutes in the unsaturated zone of the soil. In TETrans, transport through the soil profile is modeled as a series of events or processes for a finite collection of discrete depth intervals. These sequential events or processes include: (1) infiltration and drainage to field capacity, (2) instantaneous chemical equilibration for reactive solutes, (3) water uptake by the plant root resulting from transpiration and evaporative losses from the soil surface, and (4) instantaneous chemical reequilibration. Each process is assumed to occur in sequence within a given depth interval as opposed to reality where transport is a collection of simultaneous processes.

Aside from conceptualizing transport as a sequence of processes, five major theoretical assumptions are made in TETrans. First, drainage occurs through the soil profile to a depth-variable field capacity. Second, for a given depth interval the depletion of stored water by evaporation and transpiration processes does not go below a field-observable minimum water content which lies above the water content associated with the wilting point of the crop. Third, dispersion is assumed to be either negligible or part of the phenomenon of bypass. Fourth, the chemical processes of adsorption-desorption are nonhysteretic and instantaneous. Fifth, the soil profile can be divided into a finite series of discrete depth intervals with each interval having homogeneous physical and chemical characteristics.

On a microscopic scale, bypass can result when water moves through pores where stagnant areas of immobile water exist. This immobile soil water can exist as either a thin film around soil particles resulting from adhesive and cohesive forces, or as stagnant water in dead-end pores (Turner, 1958; Coats and Smith, 1964; Deans, 1964). Adjacent to the immobile film of water around a soil particle lies a mobile water phase. During an irrigation (or precipitation) event the incoming water miscibly displaces the mobile water while the immobile layer is bypassed. Bypass can also result from the movement of water through large pores, thereby bypassing the smaller pores. The net effect of bypass is that some resident soil water containing a solute is not miscibly displaced by incoming water. This subsequently affects the amount of solute within a soil-depth increment. Previous functional transport models have not dealt with the problem of bypass, which, in certain soils, can have a very profound effect upon the movement and distribution of a solute.

Bypass flow in flux-based transport models (i.e., mechanistic models) is distinguished by differences in pore-water velocity from one point to the next. In contrast, bypass flow in capacity-based models, such as a simple mass-balance approach, can be approximated by the spatial variation in the quantity of the resident pore-water which is not involved in piston-type displacement following an irrigation or precipitation event. The quantity, or more specifically, the fraction of the total resident soil solution which is not miscibly displaced by incoming irrigation or precipitation water is subject to bypass. In order to address the problem posed by bypass in the most simplistic manner, a single term, the mobility coefficient (γ), which accounts for the effects of bypass due to the presence of immobile water and preferential flow through large pores and cracks is used in TETrans. (NOTE: the mobility coefficient is the fractional representation of the percent water bypass. *Water bypass %* is actually the prompt used by the TETrans I/O interface - see the "Input Parameters" section of this chapter and the "Hydraulics" subsection of Chapter 4: Using TETrans for further details). The mobility coefficient is defined as the fraction of the resident soil water which is subject to displacement; therefore, $1-\gamma$ represents the fraction of soil water which is bypassed.

Model Description

The following model steps outline the sequence of events for the transport process within a defined depth interval, z_1 to z_2 (see Appendix A for any undefined terms):

A. Infiltration and drainage

1. Before an irrigation (BI) or precipitation, the total amount of solute, T_{BI}, in a layer of soil, z_1 to z_2, is represented as,

$$T_{BI} = T_{sw} + T_{ad} = V_t [\theta_{BI}C_{BI} + \rho_b C_{ad}] = V_{BI}C_{BI} + V_t\rho_b C_{ad} \quad [1]$$

where: V_t is a unit volume of soil within the depth interval z_1 to z_2 (m^3); T_{BI} is the total amount of solute in V_t immediately before an irrigation (kg); T_{sw} is the total amount of solute in the soil water of V_t (kg); T_{ad} is the total amount of adsorbed solute in V_t (kg); θ_{BI} is the volumetric water content immediately before an irrigation (cm^3/cm^3); C_{BI} is the concentration of solute in the soil water immediately before an irrigation (kg/m^3); ρ_b is the soil bulk density (kg/m^3); C_{ad} is the adsorbed solute concentration (kg/m^3); and V_{BI} is the volume of soil water in V_t immediately before an irrigation (m^3).

2. After an irrigation and drainage to field capacity (AI), the total amount of solute, T_{AI}, is the result of what was present before the irrigation plus what entered minus what went out,

$$T_{AI} = T_{BI} + T_{in} - T_{out} = T_{BI} + V_{in}C_{in} - V_{out}C_{out} \qquad [2]$$

a. where if $0 < \gamma \leq 1$ and

(1.) if $V_{in} > V_{fc} - (1.0 - \gamma) V_{BI}$, then

$$V_{out} = V_{in} - V_{fc} + V_{BI} \qquad [3]$$

$$C_{out} = \{ \gamma V_{BI}C_{BI} - V_{fc}C_{in} + V_{in}C_{in} + (1.0 - \gamma) V_{BI}C_{in} \} / V_{out} \qquad [4]$$

$$V_{AI} = V_{fc} \qquad [5]$$

$$C_{AI} = \{ (1.0 - \gamma) V_{BI}C_{BI} + (V_{fc} - (1.0 - \gamma) V_{BI}) C_{in} \} / V_{fc} \qquad [6]$$

(2.) else if $V_{fc} - V_{BI} < V_{in} \leq V_{fc} - (1.0 - \gamma) V_{BI}$, then

$$V_{out} = V_{in} - V_{fc} + V_{BI} \qquad [7]$$

$$C_{out} = C_{BI} \qquad [8]$$

$$V_{AI} = V_{fc} \qquad [9]$$

$$C_{AI} = \{ (V_{fc} - V_{in}) C_{BI} + V_{in}C_{in} \} / V_{fc} \qquad [10]$$

(3.) else if $V_{in} \leq V_{fc} - V_{BI}$, then

$$V_{out} = 0 \qquad [11]$$

$$C_{out} = 0 \qquad [12]$$

$$V_{AI} = V_{BI} + V_{in} \qquad [13]$$

$$C_{AI} = (V_{BI}C_{BI} + V_{in}C_{in}) / (V_{BI} + V_{in}) \qquad [14]$$

b. otherwise if $\gamma = 0$ and

(1.) if $V_{in} > V_{fc} - V_{BI}$, then Eqs. [3], [4], [5] and [6] are applied.

(2.) else if $V_{in} \leq V_{fc} - V_{BI}$, then Eqs. [11], [12], [13] and [14] are applied.

where: T_{AI} represents the total amount of solute in a volume, V_t, of soil after an irrigation (kg); T_{in} is the total amount of solute entering V_t (kg); T_{out} is the total amount of solute leaving V_t (kg); V_{in} is the volume of water entering V_t (m^3); C_{AI} is the concentration of solute in the soil water after an irrigation (kg/m^3); C_{in} is the solute concentration of the entering water (kg/m^3); V_{out} is the volume of water leaving V_t (m^3); C_{out} is the solute concentration of the exiting water (kg/m^3); γ is the mobility coefficient, or more specifically, the fraction of V_{BI} which is subject to piston-flow (where, $0 \leq \gamma \leq 1$, $\gamma = 0$ represents total bypass, $\gamma = 1$ represents complete piston-type flow); $1.0 - \gamma$ is the fraction of V_{BI} which is subject to bypass; V_{AI} is the volume of soil water in V_t after an irrigation (m^3); and V_{fc} is volume of water in V_t at field capacity (m^3).

B. Chemical equilibration

Chemical equilibration involves the partitioning of a reactive solute into the solution and adsorbed or exchange phases. Based on the conservation of mass, the total amount of a reactive solute in a layer of soil, z_1 to z_2, of thickness z (total volume V_t) after an irrigation is equal to the amount of reactive solute present before the irrigation, plus the total solute entering the layer minus the total solute leaving the layer. This is represented in simple mathematical form as Eq. [2]. After an irrigation the volume of water in V_t equals either field capacity, $V_{Al} = V_{fc}$, or $V_{Al} = V_{Bl} + V_{in}$. Assuming instantaneous adsorption-desorption and assuming the reactive solute obeys the Langmuir adsorption isotherm model, T_{Al} is partitioned into the adsorbed and the solution phases according to Eq. [15],

$$T_{Al} = V_{Al} C_{Al} + V_t \rho_b [kbC_{Al} / (1 + kC_{Al})] \qquad [15]$$

where: C_{Al} is the concentration of the solute in the soil water after an irrigation (kg/m^3), ρ_b is the bulk density (kg/m^3), k is the Langmuir affinity coefficient (m^3/kg) and b is the adsorption maximum (mg/kg). Using Eqs. [2] and [15] and solving for C_{Al}, Eq. [16] is obtained,

$$C_{Al} = \frac{-(V_{Al} + V_t \rho_b kb - T_{Al} k) \pm \sqrt{(V_{Al} + V_t \rho_b kb - T_{Al} k)^2 + 4V_{Al} kT_{Al}}}{2 V_{Al} k} \qquad [16]$$

where the positive solution is the one needed. The total amount of solute in the adsorbed phase is represented by Eq. [17],

$$T_{ad} = V_t C_{ad} = V_t \rho_b [kbC_{Al} / (1 + kC_{Al})] \qquad [17]$$

and therefore, the adsorbed solute concentration, C_{ad}, is,

$$C_{ad} = \rho_b [kbC_{Al} / (1 + kC_{Al})] \qquad [18]$$

Equations [16] and [18] determine the solution and adsorbed phase solute concentrations following an irrigation by using the value of T_{Al} from the mass-balance equation (Eq. [2]) and the appropriate value for V_{Al}, either field capacity, $V_{Al} = V_{fc}$, or $V_{Al} = V_{Bl} + V_{in}$. The same logic can be followed for other chemical models including the Freundlich or linear adsorption model and the Mezuman-Keren boron adsorption model.

C. Plant water uptake

The total amount of evapotranspiration between irrigation events and the plant root distribution of a crop are needed as input for TETrans. The plant water uptake model simulates the net loss of water from each depth increment within the root zone of a maturing plant. Root growth is assumed to occur linearly from the date of planting to the date of maturity. If the plant is harvested and the root system is terminated, all subsequent loss of water from the root zone occurs by a simulation of evaporative loss from the soil surface. TETrans does not account for the upward movement of solute resulting from the processes of evaporation or transpiration. Evapotranspiration is only viewed as a sink for water loss which results in the concentration of the solute within the root zone. It is not viewed as creating a potential gradient which results in the net movement of the solute.

In TETrans the distribution of the removal of water by the plant root is fitted with the option of two models: linear or exponential distribution. The linear root water uptake model is that of Perrochet (1987) which is a synthesis of previous models and work presented by Molz and Remson (1970), Feddes et al. (1978), Hoagland et al. (1981), Ritchie (1984), Ritchie et al. (1984) and Prasad (1988). On a capacity basis, the volumetric root extraction function, S, is expressed by,

$$S(\psi,z) = r(\psi)\, g(z)\, T_p \qquad [19]$$

where, ψ is the soil-water suction head (m), z is the soil depth (m), $r(\psi)$ is the reducing factor, $g(z)$ is the root distribution function and T_p is the potential volumetric transpiration (m^3). Perrochet (1987) expresses the linear root distribution function by,

$$g(z) = \{ \alpha_1 (2z - L) + L \} / L^2 \qquad [20]$$

where, α_1 is the linear plant root disrtibution coefficient ($-1 \leq \alpha_1 \leq 1$) , $z \leq L$ and L is the plant root depth (m). The root distribution function must be normalized so that its integral over L is unity. It is assumed for the model that moisture conditions are optimal; consequently, the reducing factor, $r(\psi)$, is equal to 1. Since the actual volumetric transpiration, T_a, is the integral of the volumetric extraction function from the soil surface (z = 0) to the depth of root penetration (z=L), then the relative water uptake for a linear root distribution, $U_l(z)$, over the soil depth interval z_1 to z_2 (where, $0 \leq z_1 < z_2 \leq L$) becomes,

$$U_l(z) = [\alpha_1 / L^2] (z_2^2 - z_1^2) - [(\alpha_1 / L) - (1/L)] (z_2 - z_1) . \qquad [21]$$

Following the same logic, the relative water uptake for an exponential root distribution, $U_e(z)$, over the soil depth interval z_1 to z_2 becomes,

$$U_e(z) = (e^{-az_1} - e^{-az_2}) / (1 - e^{-aL}) \qquad [22]$$

where, $a = \alpha_2 / L$, and α_2 is the exponential plant root distribution coefficient. Therefore, the water loss within the z_1 to z_2 depth interval is equal to the actual volumetric transpiration multiplied by the relative water uptake. V_{Al} is adjusted to V_{et} which represents the volume of soil water in V_t following the removal of water by root uptake to meet transpiration needs. So, the water withdrawn by plant roots for any given depth increment, z_1 to z_2, is removed in a manner which corresponds to the relative plant root water uptake expressed by either Eq. [17] or [18] which are a reflection of the plant root distribution. Within any given depth increment, the residing soil water cannot be withdrawn below a minimum volume of water, V_{min}, by the plant root. V_{min} is an empirical value which lies above the water content at the wilting point and represents the lowest volume of water which is observed to occur within V_t for all ET events.

Concomitant with the removal of water by the roots is the concentration of the solute. During the extraction of soil water, roots behave similar to a semipermeable membrane. Solutes remain behind as the water is extracted. Therefore, evapotranspiration results in the concentration of solutes within the root zone.

D. Chemical reequilibration

Following the removal of water by the roots the volume of water in V_t just before the next irrigation becomes,

$$V_{BI} = V_{AI} - V_{et} \qquad [23]$$

where V_{et} is the volume of water in V_t removed by evapotranspiration (m^3). Concomitantly, the solute is concentrated by the factor V_{AI} / V_{et} and another chemical equilibration occurs. Once again, Eqs. [16] and [18] are used to determine the solution and adsorbed phase solute concentrations except V_{BI} is used in place of V_{AI}.

Bypass Considerations

In the current version of TETrans, the mobility coefficient (or fraction of water involved in bypass flow) can have values which are constant over time and depth, or constant over time and variable with depth. In the future an additional option of the mobility coefficient will be variable over time and with depth. The easiest way of determining mobility coefficient values for the first two cases (i.e., constant over time and depth, or constant over time and variable with depth) is simply by trial-and-error. Mobility coefficient value(s) are simply selected which result in the best simulation of measured chloride concentrations in the soil solution.

Corwin et al. (1990a) have already shown how mobility coefficient values which vary both temporally and spatially can be determined. (To obtain the software for determining spatially- and temporally-variable bypass coefficient values contact D. L. Corwin). The determination of temporally- and spatially-variable values for the mobility coefficient is based upon the deviation of measured soil solution chloride concentrations from calculated concentrations assuming complete piston-type displacement of solute. In TETrans this deviation is assumed to be attributed in large part to bypass resulting from preferential movement through macropores and from the movement of a mobile water phase, thereby bypassing small dead-end pores and a stagnant immobile phase of water. Though dispersion and anion exclusion would also account for the deviation of chloride transport from strict piston flow, these effects are assumed to be neglible. If dispersion is a significant factor then its effects are assumed to be inclusive within the bypass phenomenon, and compensated for in the mobility coefficient. To calculate the mobility coefficient, γ, for each irrigation and for each depth increment, Eqs. [2] and [4] are used. Eq. [2] can be rearranged to give Eq. [24],

$$C_{out} = (T_{BI} + V_{in}C_{in} - T_{AI}) / V_{out} \qquad [24]$$

and Eq. [4] can be rearranged to give Eq. [25],

$$\gamma = (V_{out}C_{out} - V_{out}C_{in}) / (V_{BI}C_{BI} - V_{BI}C_{in}) \ . \qquad [25]$$

Since T_{AI} can be calculated from the measurement of the chloride concentration of the soil solution at field capacity (for a nonreactive solute, $T_{AI} = V_{fc}C_{fc}$), then γ can be determined by substituting Eq. [24] into Eq. [25],

$$\gamma = (V_{in}C_{in} + T_{BI} - V_{fc}C_{fc} - V_{out}C_{in}) / (V_{BI}C_{BI} - V_{BI}C_{in}) \qquad [26]$$

which is the same as multiplying Eqs. [5] and [6] and solving for γ. Equation [26] holds for the situation where $V_{in} > V_{fc} - (1.0 - \gamma) V_{BI}$ for $0 < \gamma \leq 1$. However, since γ is precisely the term that is being determined, then Eq. [26] can only realistically be used when $V_{in} > V_{fc}$. If it is found that the total chloride, T_{AI}, measured for a depth increment is equal to Eq. [9] multiplied by Eq. [10] (i.e., $T_{AI} = (V_{fc} - V_{in}) C_{BI} + V_{in}C_{in}$), then it is known that the condition $V_{fc} - V_{BI} < V_{in} < V_{fc} - (1.0 - \gamma) V_{BI}$ for $0 < \gamma \leq 1$ is the case; consequently, it is assumed that,

$$\gamma = (V_{in} - V_{fc} + V_{BI}) / V_{BI} \qquad [27]$$

since it is impossible to determine γ explicitly and this represents the closest logical approximation. If the total measured chloride is equal to Eq. [13] multiplied by Eq. [14] (i.e., $T_{AI} = V_{BI}C_{BI} + V_{in}C_{in}$), then it is known that the condition $V_{in} \leq V_{fc} - V_{BI}$ exists; consequently, $\gamma = 0$ is assumed. If $T_{AI} = V_{BI}C_{BI} + (V_{fc} - V_{BI})C_{in}$ (i.e., T_{AI} equals Eq. [5] multiplied by Eq. [6] for $\gamma = 0$) when $V_{in} > V_{fc} - V_{BI}$, then $\gamma = 0$. The only condition for which γ has not been determined is when $V_{fc} - (1.0 - \gamma) V_{BI} \leq V_{in} \leq V_{fc}$ for $0 < \gamma \leq 1$. If all other conditions are not met, then this condition is assumed to be the case and Eq. [26] is invoked. Anomalous situations could arise where γ is calculated by Eq. [26] to be outside its defined range. By definition $0 \leq \gamma \leq 1$, so if γ is calculated to be less than 0, then γ is set equal to 0. Similarly, if γ is calculated to be greater than 1, then γ is set equal to 1. The possible reasons for γ extending beyond the range of 0 to 1 would be soil solution sampling errors and errors in the measurement of soil solution chloride levels.

INPUT PARAMETERS

The input parameters needed by TETrans for each soil layer include: soil layer thickness; water bypass percentage; initial solute concentration in the soil water; initial water content of the soil; adsorption coefficients; dates and amount of irrigation water applied; total evapotranspiration between irrigations; bulk density; water content at field capacity; minimum water content (or water content just above the wilting point); solute concentration of each irrigation water applied; and date of planting, days to maturity, date of harvest, maximum depth of root penetration and plant root distribution for each crop.

Though seemingly formidable in its input requirements, TETrans is far less parameter intensive than other transient-state models of solute transport, especially previous analytical and numerical deterministic models. In addition, the input parameters are more readily obtained since TETrans is a capacity-based model; consequently, complex parameters for rate-based models such as hydraulic conductivity - matric potential relations, dispersion - pore water velocity relations and water content - matric potential relations are not needed. Furthermore, capacity parameters are known to be less spatially variable than rate parameters.

A discussion of some of the less intuitive input parameters is provided for the user. The appropriate units of measurement for each input parameter are provided for the user within each input dialog box.

Percent Water Bypass

The most difficult parameter to determine is the fraction (or percentage) of water which is involved in bypass flow, referred to in the "Theory" section of this chapter as the mobility coefficient. A detailed description of how the mobility coefficient can be established is given in the "Bypass Considerations" section. A mobility coefficient of 0.50 (or 50% water bypass) was found by Corwin et al. (1990a) for a loam soil.

Immobile Water Fraction

The immobile water fraction is another means of handling bypass flow in TETrans. As pointed out in the "Theory" section of this chapter, there are believed to be two phases of soil water during water flow: a mobile and an immobile water phase. The immobile water phase is bypassed by the mobile phase as water moves through the soil profile. Several mechanistic models have been developed around the existence of mobile-immobile water phases and they will not be discussed here. For overviews of this modeling approach the user is referred to brief, but insightful review articles by Addiscott and Wagenet (1985), Nielsen et al. (1986) and van Genuchten and Jury (1987). Though hard to find, some immobile water fractions for different soil textures have been calculated and appear in the scientific literature. Stony soil and sandy soil have been found to have immobile water fractions of 0.15 and 0.33, respectively.

In TETrans, a value must only be entered either for percent water bypass or for the immobile water fraction, but not for both. Though the user is never aware of it, there are actually two solute transport models in TETrans: a hydraulic bypass transport model and a mobile-immobile transport model. The solute transport model that is chosen depends upon which hydraulic parameter, either percent water bypass or immobile water fraction, receives a value from the user. The mobile-immobile transport model is not only capable of handling the physical phenomenon of hydraulic bypass, but it also models diffusion-controlled physical nonequilibrium. The mobile-immobile model assumes that the sorption rate is controlled by the rate at which ions diffuse from relatively mobile (flowing) liquid regions to reaction sites in equilibrium with immobile (nonflowing or dead-end) water. Diffusion into and out of the immobile water areas is modeled as a first-order exchange process.

In the current version of TETrans (version 1.4), the immobile water fraction prompt appears grayed-out indicating that it is inactive.

This is because the mobile-immobile transport model is not present in version 1.4. At present the mobile-immobile transport model is involved in a verification study comparing various transport models. Once the results of this study are published, the mobile-immobile transport model will appear in TETrans, version 2.0, along with several other modifications. All users will receive the updated TETrans version after the publication.

Evapotranspiration Amount

The evapotranspiration (ET) amount may also be difficult to determine. TETrans specifically requests the amount of ET between irrigation or precipitation events. A precipitation/irrigation event is considered to be each day that irrigation water is applied or precipitation occurs. Therefore, the ET is calculated to be the total evapotranspiration from one irrigation/precipitation event to the next event.

A gross estimate of evapotranspiration can be made from consumptive water use data for a crop. Consumptive water use data is generally available for various locations from either irrigation management district offices, the state department of agriculture, the local agricultural extension office or the local soil and water conservation agency. If the crop consumptive water use is known then a simple sinusoidal wave can be applied to estimate daily evapotranspiration requirements. From this information, the total evapotranspiration between irrigations (or precipitation events) can be easily estimated.

Adsorption Coefficients

Adsorption coefficients are probable the next most difficult parameter to determine or locate in the literature. A good general reference for adsorption coefficient data and methods of estimating adsorption coefficients is W. J. Lyman's chapter titled "Adsorption Coefficient for Soils and Sediments" in Lyman, Reehl and Rosenblatt's Handbook of Chemical Property Estimation Methods. (Note: Estimated adsorption coefficients should only be used when there is no other source of information because estimation methods of adsorption coefficients are generally very inaccurate). With regards to the Keren boron adsorption coefficients, the user is referred to the publications by Keren and Mezuman (1981), Mezuman and Keren (1981), and Keren et al. (1981) which are listed in the reference appendix.

Plant Parameters

The plant parameters including maximum root penetration depth and root water uptake distribution for a crop or vegetative cover are usually available from the local agricultual extension personnel. As a rule-of-thumb, most plants are believed to have water uptake distributions of 40-30-20-10. In other words, 40% of the total transpired water is taken up from the top quarter of the root zone, 30% from the second quarter of the root zone, 20% from the third quarter and 10% from the bottom quarter. Unfortunately, maximum plant root depth can vary considerably from crop to crop and sometimes even from one location to the next. Check with local extension experts to get the best possible estimate.

Field Capacity

A parameter which may be unknown outside of the confines of the soil scieince field is that of field capacity. Field capacity represents the water content of a soil or soil layer after free drainage has stopped. This is generally considered to be the water content of the soil 2-3 days (sometimes longer depending on the soil texture) after an irrigation event. Field capacity can range from 0.08 cm^3/cm^3 for a sandy soil to 0.35 cm^3/cm^3 for a clay soil.

Minimum Water Content

The minimum water content is considered to be the lowest water content reached within a soil layer as a result of the extraction of water by plant roots. This water content is slightly above the water content at the wilting point of the plant. A range of minimum water contents would be 0.05 cm^3/cm^3 for sandy soil to 0.16 cm^3/cm^3 for a clay soil.

Soil pH

Soil pH is a parameter that needs no real discussion, except with regards to its use by TETrans. In TETrans, pH is only needed when using the Keren boron adsorption model. For this chemical model selection, pH information is needed as an initial condition and after each irrigation or precipitation event. Make sure that pHs are entered for both situations. The soil pH should be measured at field capacity just after an irrigation or a precipitation event. (See the **Soil pH** command of the **Parameters** menu in Chapter 4: Using TETrans for additional information).

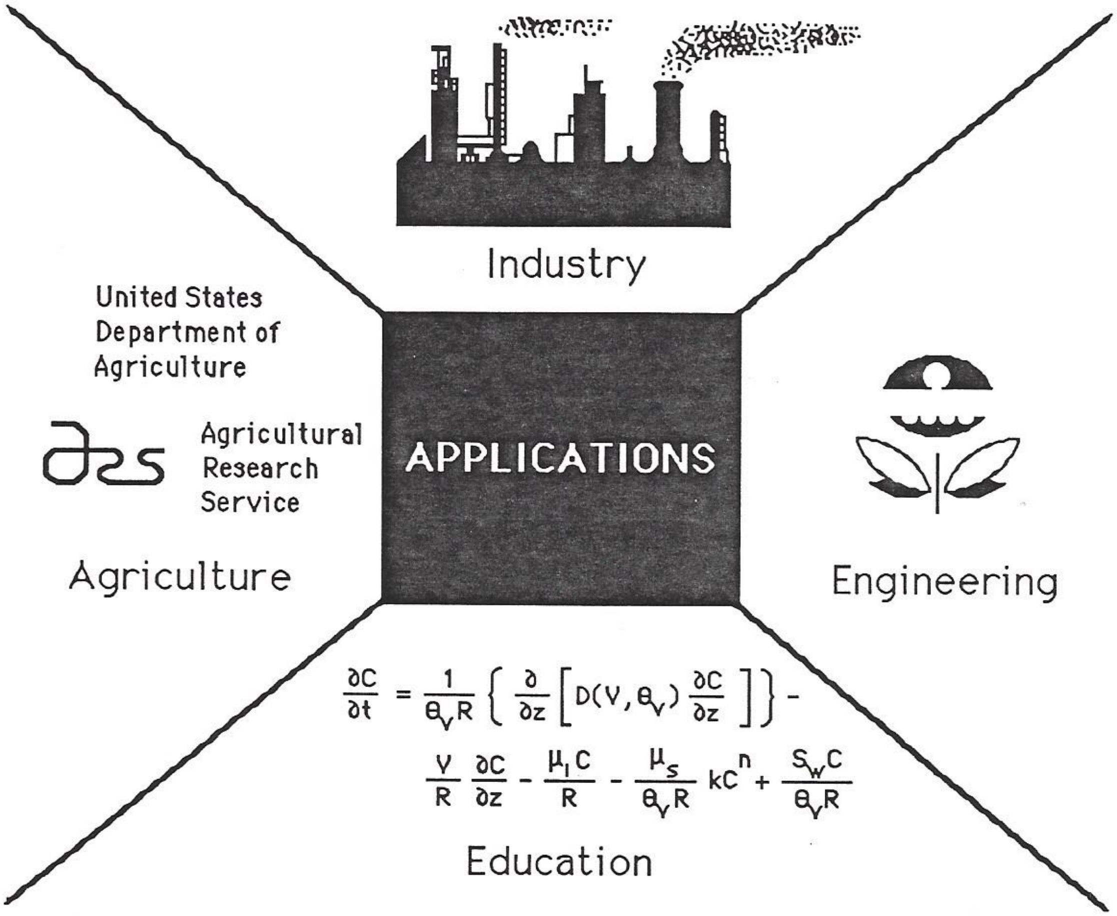
The extreme range of pH for most mineral soils is 3.5 to 10.0. However, the common range in pH for humid region mineral soils is 5 to 7, while arid region mineral soils commonly range from 7 to 9.

APPLICATIONS

TETrans was developed to meet two application needs. The first was for assessing the movement of nonvolatile solutes under transient-state field conditions. This would meet the needs of agricultural, industrial and engineering applications. The second was as an educational tool for studying the interrelationship of the physicochemical processes involved in the movement of a nonreactive or reactive solute through the vadose zone.

The need for a practical, management-oriented model for simulating solute movement by leaching is based upon two limitations of more theoretically rigorous models of transport. First, the soil data needed for sophisticated analytical and numerical models is typically well beyond the capacity of real-world users. Second, the spatial variability typical of field soils limits the accuracy of application of exact transport theory. Stochastic transport models do not provide a viable alternative for most real-world applications since again the data upon which they are derived is too labor and cost intensive to be practical.

As an educational tool, TETrans' user-friendly I/O interface reduces the complexities of input and output found in most transport models to interactive menu screens and dialog boxes. The user's time is spent more in the analysis of output than in learning how to operate the software. As a functional model, TETrans incorporates simplified treatments of solute and water flow with no claim to fundamentality; consequently, the effects of dispersion and preferential flow are lumped into a single term. In addition, no regard is given to the kinetics or hysteresis of adsorption-desorption reactions. Nonetheless, the basic interrelationships between water movement, evapotranspiration, and the equilibrium-chemistry processes of adsorption and exchange can be studied.


Industry
United States
Department of
Agriculture
ars
Agricultural
Research
Service
Agriculture
APPLICATIONS
Engineering
$\frac{\partial C}{\partial t} = \frac{1}{\theta_v R}\left\{\frac{\partial}{\partial z}\left[D(V,\theta_v)\frac{\partial C}{\partial z}\right]\right\} - \frac{V}{R}\frac{\partial C}{\partial z} - \frac{\mu_l C}{R} - \frac{\mu_s}{\theta_v R}kC^n + \frac{S_w C}{\theta_v R}$
Education

VERIFICATION TEST DATA

Verification test data for TETrans is documented in two Journal of Environmental Quality publications by Corwin et al. (1990a and 1990b).

STRENGTHS & LIMITATIONS

TETrans' strengths include:

- ◊ specifically designed for real-world transport applications
- ◊ designed to use input parameters which are readily available
- ◊ uses capacity input parameters which are less spatially variable than rate parameters
- ◊ has the computational speed of an analytical solution
- ◊ has the versatility of a numerical approach
- ◊ designed with a user-friendly I/O interface

A more detailed discussion of each strength is provided:

TETrans is a solute transport software package which is specifically designed for real-world transport applications. Because of cost constraints, real-world problems rarely permit the acquisition of difficult-to-measure parameters commonly needed for most transport models (i.e., mechanistic and stochastic models). For this reason, TETrans is designed for situations where there is a limited knowledge of sophisticated transport information. There is no need for dispersion - pore velocity relationships, water content - matric potential relationships and hydraulic conductivity - matric potential relationships.

Functional models, such as TETrans, make no claim to fundamentally describing the individual processes involved in transport. Rather, the mechanisms are treated in a simplified manner. As a functional model, TETrans requires capacity parameters which are known to be less spatially variable than rate parameters. In addition, capacity parameters are generally more readily available because they are easier to measure than rate parameters. Furthermore, because TETrans uses a mass-balance approach, it offers the speed of an anlytical solution, yet it has the versatility of a numerical approach, without a significant sacrifice in accuracy.

To add to TETrans' utililty, a user-friendly interface which uses Apple Macintosh standard pulldown menus and dialog boxes has been incorporated. The I/O interface enables the user to spend less time learning the operation of the software and more time analyzing data.

In contrast, the weaknesses of TETrans include:

- ◊ does not account for vapor phase flow
- ◊ does not account for lateral or upward water flow
- ◊ does not fundamentally describe dispersion or diffusion
- ◊ does not have the ability to account for spatially varying soil properties
- ◊ requires a precise knowledge of evapotranspiration
- ◊ outputs concentration distributions which are average concentration distributions

A discussion of each weakness is presented:

First and foremost, TETrans is designed to serve as a tool for use on real-world problems. To adhere to this philosophy of design, certain limitations on TETrans' complexity had to be made. Because the intended purpose of TETrans is to simulate the one-dimensional vertical transport of nonvolatile solutes, no consideration is given to vapor phase flow or lateral water flow. In the current version, the upward movement of water also is not taken into account.

To maintain its simplicity and practicality, TETrans is designed to be a management-oriented model, so its use as a research tool is limited because it is not mechanistically rigorous. In TETrans, there is no claim to a fundamentally rigorous description of the mechanisms involved in solute transport.

Even though TETrans uses capacity-type input parameters which are less spatially variable than rate parameters, it is still a deterministic model. As a deterministic model, a uniquely-definable outcome is presumed; therefore, TETrans does not account for spatially variable soil properties. However, stochastic models which do account for spatial variability are prohibitively expensive due to the intensive labor requirements necessary to establish the probability density functions for each property influencing the transport process.
Even though TETrans is designed for simplicity of use, some of the input parameters may still be difficult to obtain. Because TETrans uses a mass-balance approach, a precise knowledge of the amount of evapotransporation is essential, but evapotranspiration is often difficult to accurately measure. However, this is a weakness of all

transport models. Evapotranspiration is a crucial sink term in any modeling approach.

The solute concentration determined by TETrans is an average concentration for a depth increment rather than the single-valued concentration associated with a specific depth of mechanistic-deterministic models. For practical applications, average concentrations are usually an adequate first approximation to the description of solute movement.

CHAPTER 3: SAMPLE SIMULATION

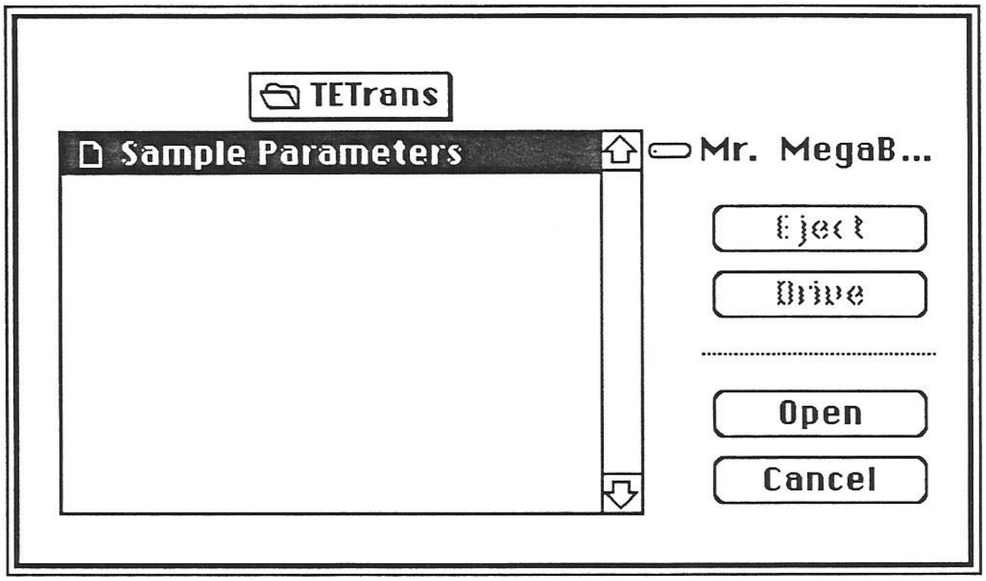
This chapter provides the user with a sample simulation. The sample simulation will walk you through the basic operation of TETrans' user-friendly I/O interface. Upon finishing the sample, more detailed information on any of TETrans' functions can be found in Chapter 4: Using TETrans.

Starting the Sample Simulation

The first step, if you haven't done so already, is to copy TETrans to your hard disk, if you have one, or to another floppy. Never work with the original TETrans disk. See Chapter 1: Installing TETrans.

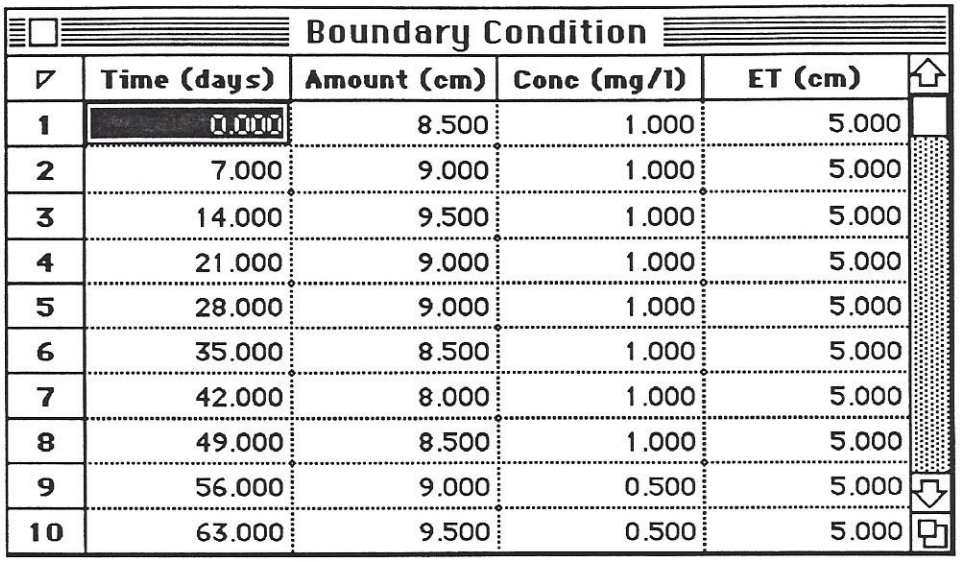
To begin the sample simulation, locate the TETrans folder (on your hard disk or floppy) and double-click on the folder to open it. Then start TETrans by double-clicking on the TETrans application icon.

◊ Choose **Open Parameters...** from the **File** menu. A dialog box appears:



◊ Select the sample data file of input parameters titled "Sample Parameters" by clicking on it once with the mouse. Then click on the "Open" button. This will dismiss the dialog box and load in the TETrans model parameters from the file.

◊ To view any of the model parameters, select any of the items from the **Parameters** menu. For example, to view the boundary conditions, choose **Boundary Condition...** from the **Parameters** menu. A spreadsheet window will appear:

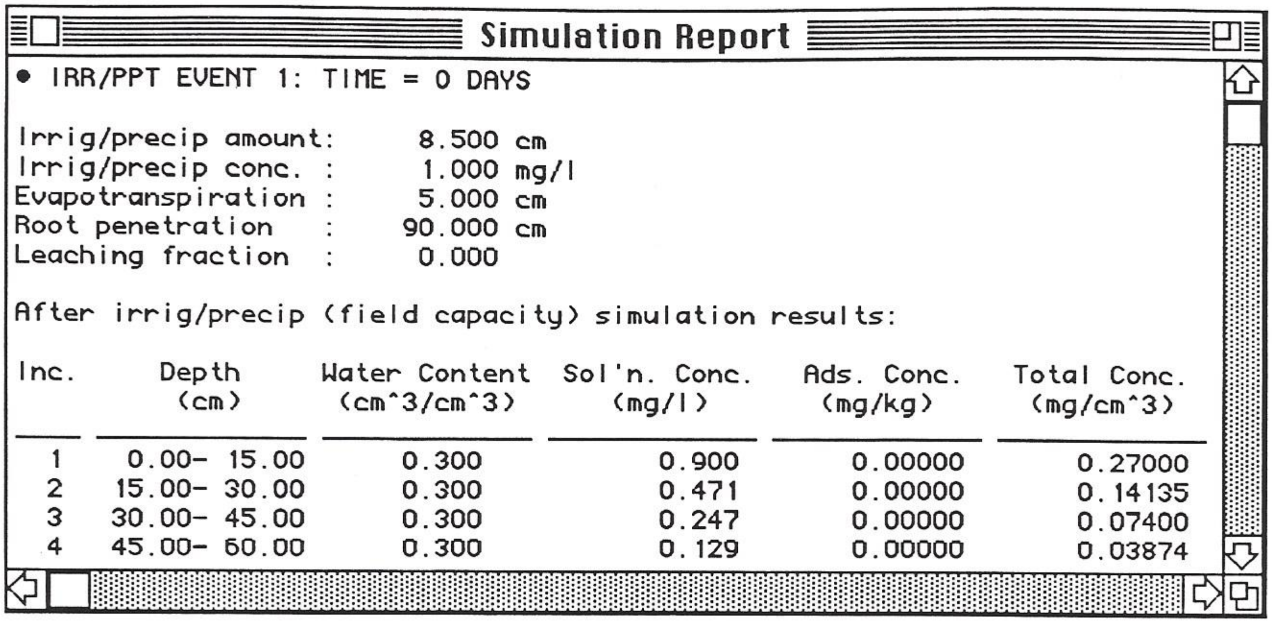


Boundary Condition

▽	Time (days)	Amount (cm)	Conc (mg/l)	ET (cm)
1	0.000	8.500	1.000	5.000
2	7.000	9.000	1.000	5.000
3	14.000	9.500	1.000	5.000
4	21.000	9.000	1.000	5.000
5	28.000	9.000	1.000	5.000
6	35.000	8.500	1.000	5.000
7	42.000	8.000	1.000	5.000
8	49.000	8.500	1.000	5.000
9	56.000	9.000	0.500	5.000
10	63.000	9.500	0.500	5.000

This spreadsheet contains information about the amounts of water and solute entering or leaving the surface of the soil. The **Time** column has the times of all of the irrigation or precipitation events. The **Amount** column has the amounts of irrigation water applied or precipitation that fell for each event. The **Conc** column has the irrigation water concentrations for the solute being modeled. The **ET** column has the evapotranspiration for each event. This is measured as the total evapotranspiration occuring from the time of one event until just prior to the next event. All of these values can be edited using standard Macintosh editing techniques. See the "Boundary Condition..." section in Chapter 4 for more details.

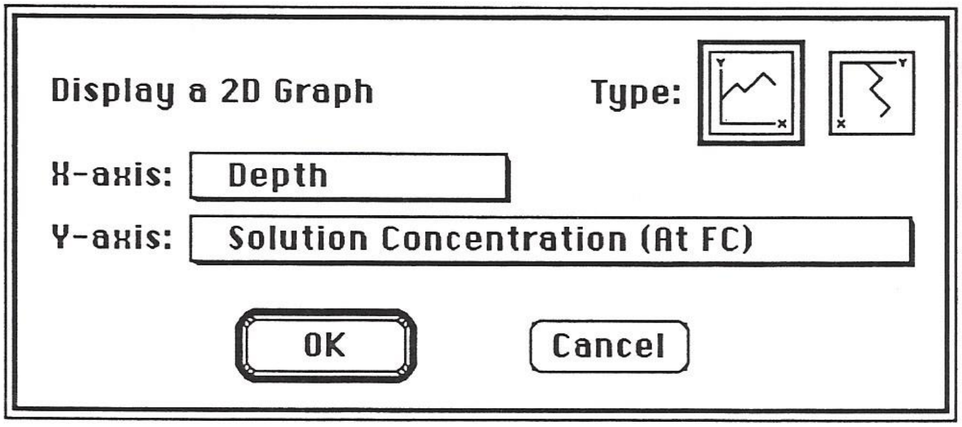
◊ Perform the simulation by choosing, appropriately enough, **Run** from the **Run** menu. The spinning tetrahedron cursor will appear. This will signal a coffee break for those who are owners of MacPluses running multi-year simulations. You may abort any simulation by holding down the command key (⌘) and typing a period (.). A window showing the results of the simulation will appear:



```
• IRR/PPT EVENT 1: TIME = 0 DAYS

Irrig/precip amount:      8.500 cm
Irrig/precip conc. :      1.000 mg/l
Evapotranspiration :      5.000 cm
Root penetration   :     90.000 cm
Leaching fraction  :      0.000

After irrig/precip (field capacity) simulation results:

Inc.      Depth       Water Content   Sol'n. Conc.    Ads. Conc.     Total Conc.
           (cm)       (cm^3/cm^3)       (mg/l)         (mg/kg)       (mg/cm^3)
____  ____________   _____________   ____________   ____________   ____________
 1     0.00- 15.00       0.300           0.900        0.00000        0.27000
 2    15.00- 30.00       0.300           0.471        0.00000        0.14135
 3    30.00- 45.00       0.300           0.247        0.00000        0.07400
 4    45.00- 60.00       0.300           0.129        0.00000        0.03874
```

The window may be resized, zoomed and scrolled to view its full extent. See the "Simulation Report" section in Chapter 4: Using TETrans for more details. The window shows simulated values for root depth, leaching fraction, soil water content; and solution, adsorbed and total solute concentration for each of the irrigation or precipitation events and soil profile depths.

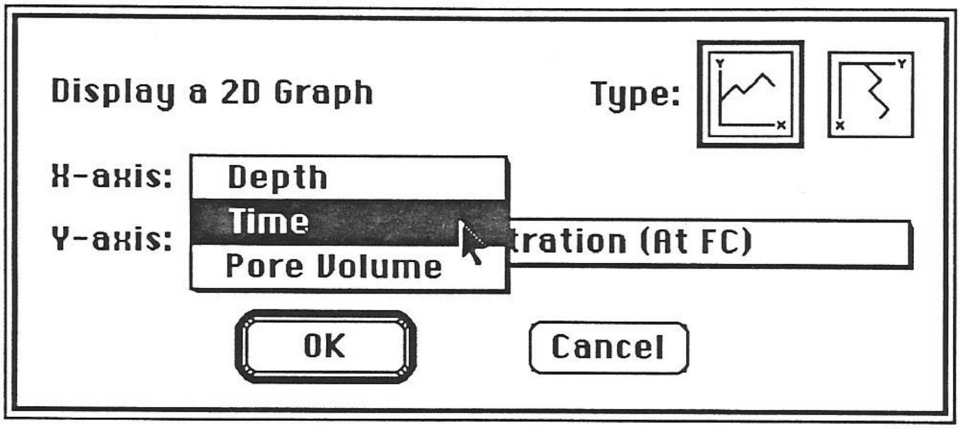
◊ A variety of graphs of the simulation results can be viewed with TETrans. To get a two-dimensional view of your simulation, choose **2D Graph** from the **Output** menu. This will display the dialog window:



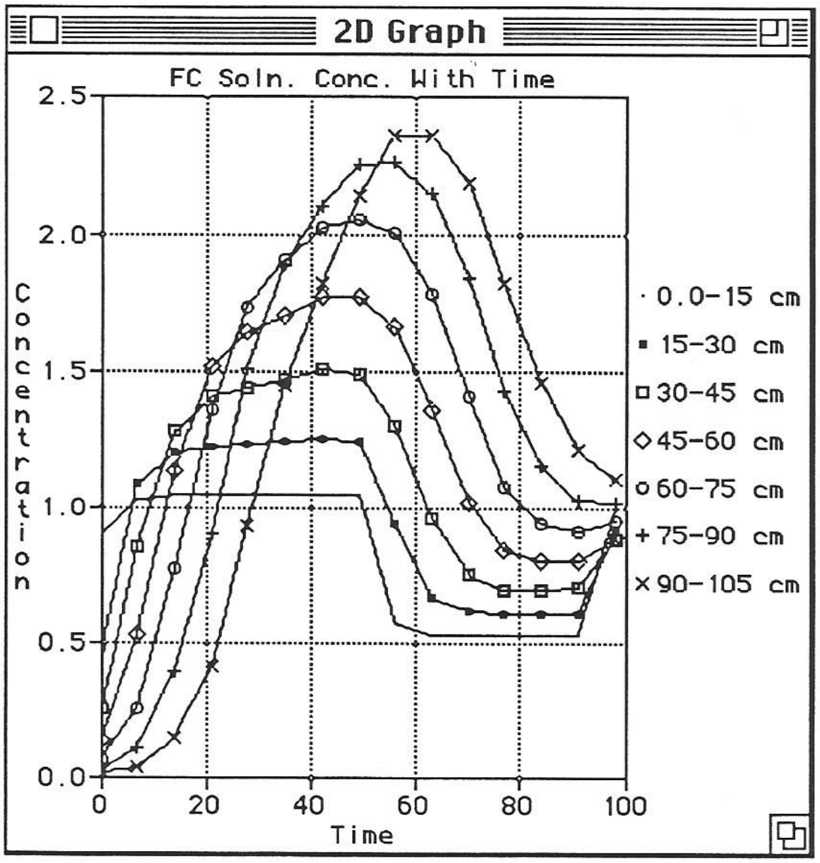
X-axis: Depth

Y-axis: Solution Concentration (At FC)

Pop-up menus allow you to select the x- and y-axis data to be displayed. Move the cursor over the "X-axis:" box and hold the mouse button down to view the x-axis pop-up menu. Now, with the mouse button still down, drag the cursor and select "Time" as shown below:

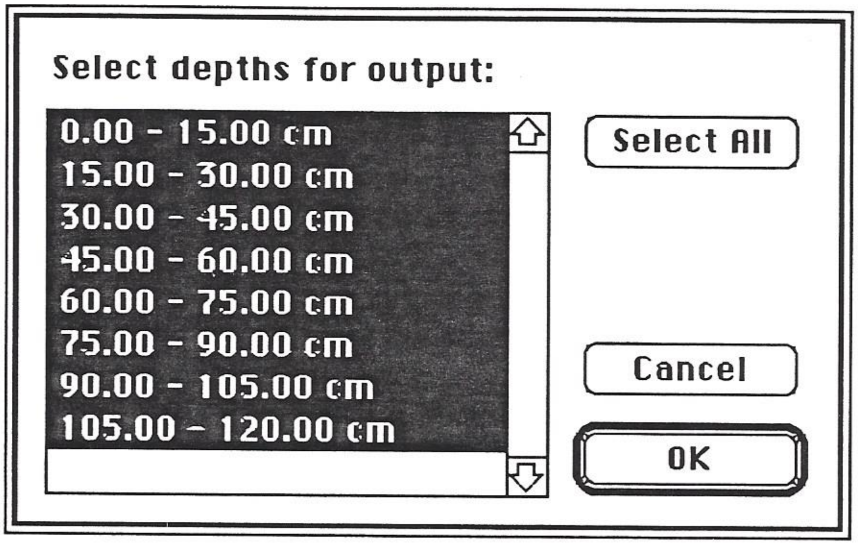


Click on the "OK" button. A graph of the simulated solution concentration values with time will appear:



NOTE: 2D graphs can display only up to seven curves. Close this window by choosing **Close** from the **File** menu. A dialog window will appear asking if you would like to save the graph to a file. Click the "No" button, which dismisses the dialog window and removes the graph.

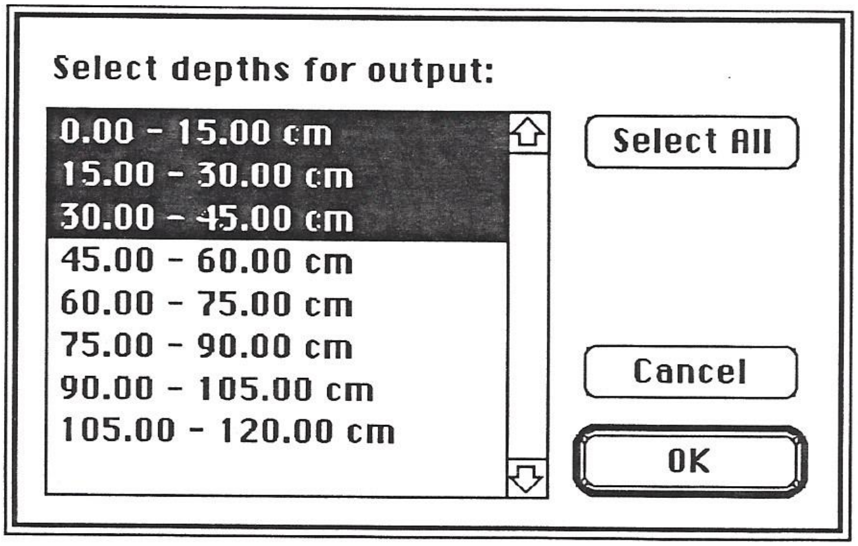
◊ To limit the number of curves displayed in a 2D graph, choose **Output Depths** from the **Output** menu. A dialog listing all of the available soil layer depths will appear:



Select depths for output:

0.00 - 15.00 cm
15.00 - 30.00 cm
30.00 - 45.00 cm
45.00 - 60.00 cm
60.00 - 75.00 cm
75.00 - 90.00 cm
90.00 - 105.00 cm
105.00 - 120.00 cm

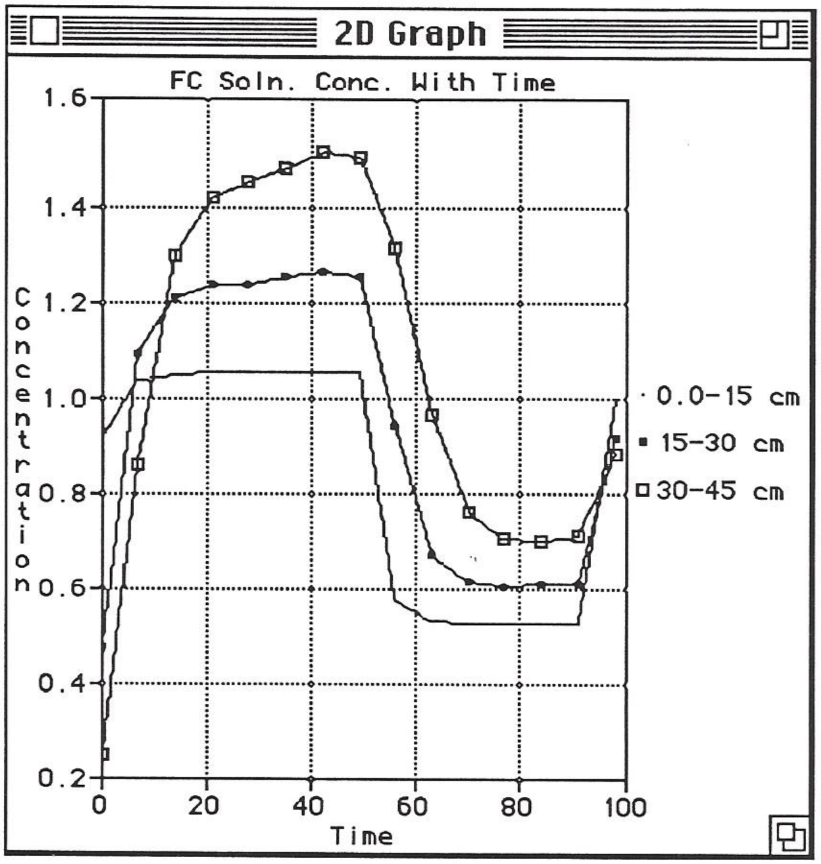
Using the mouse, click on the line that reads "45.00 - 60.00 cm" and drag downwards, holding the mouse button down as you go. This will deselect all but the first three soil layers. The dialog should now look like this:



Select depths for output:

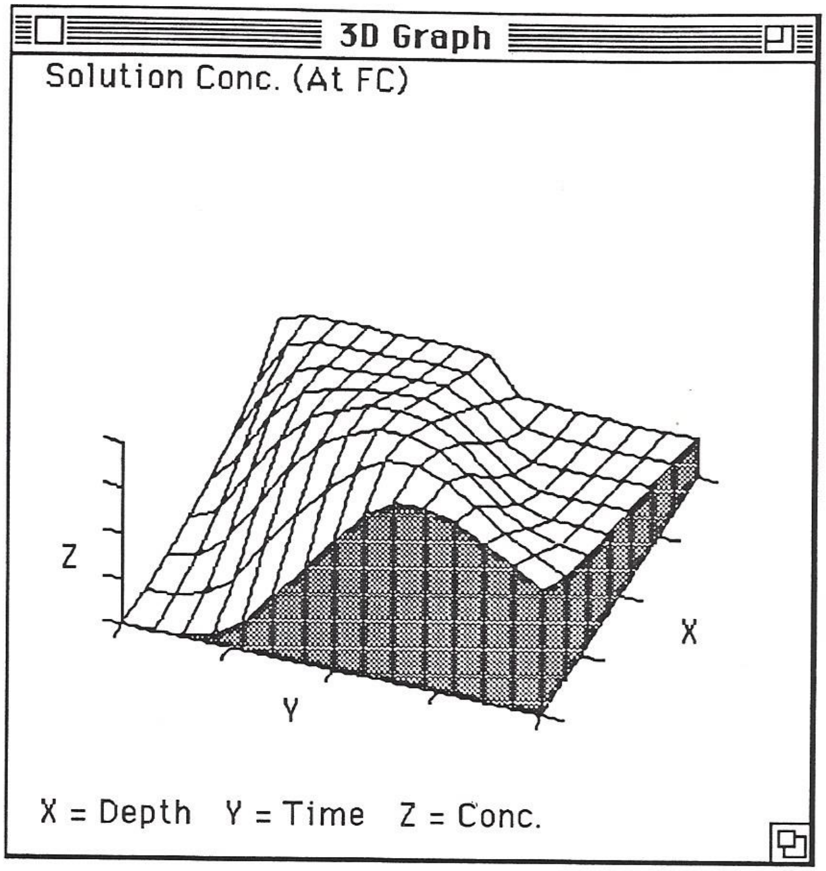
0.00 - 15.00 cm
15.00 - 30.00 cm
30.00 - 45.00 cm
45.00 - 60.00 cm
60.00 - 75.00 cm
75.00 - 90.00 cm
90.00 - 105.00 cm
105.00 - 120.00 cm

Click the "OK" button to dismiss the dialog. Now go back and select **2D Graph** from the **Output** menu again. Select "Time" from the "X-axis:" pop-up menu, as before, and click on the "OK" button. You will get the following graph:



which now displays simulated concentration values for only the first 3 soil layers: 0-15, 15-30 and 30-45 cm.

◊ For a rapid assessment of the overall simulation results, select **3D Graph** from the **Output** menu. As in the 2D Graph case, pop-up menus allow you to select the axis values for output. Click "OK", and a 3D graph of simulated solution concentration values with time and depth will be displayed:



Here the surface represents the solution concentration (z-axis) as a function of both depth and time (x-axis and y-axis, respectively). See the "3D Graph" section of Chapter 4: Using TETrans for more information.

◊ Quit the TETrans application by choosing **Quit** from the **File** menu. A dialog will ask you if you would like to save the 3D Graph window to a file. Choose "No". This dismisses the dialog, and TETrans will finish quitting without saving the graph. A "Save this window?" dialog will appear for each of the open windows as the application closes down. Select "No" in response to each.

◊ This sample simulation has briefly covered some of the features of TETrans, and it's approach to modeling solute movement. The next step is to input the parameters particular to your scenario. See Chapter 2: Introduction to TETrans for an overview of the input needs of TETrans. Chapter 4: Using TETrans gives detailed descriptions of entering input parameters, running the model, and displaying the results.

CHAPTER 4: USING TETrans

This chapter provides the user with a detailed description of each function of TETrans' user-friendly I/O interface. The I/O interface is the shell of TETrans through which the user enters input data, runs the TETrans model simulation and selects the form (i.e., text report or graphics display) of the simulation output.

STARTING TETrans

TETrans can be run from your hard disk or floppy disk. Follow the steps in Chapter 1: Installing TETrans to, obviously, install TETrans on your hard disk.

To start TETrans, open the TETrans folder by double-clicking on the icon. Open TETrans by double-clicking on the TETrans icon. The TETrans menu bar and an About TETrans box will appear. (Clicking the mouse within certain locations of the About TETrans box can sometimes have unusual effects). You are now ready to use TETrans. All I/O interactions with TETrans are conducted through the use of the menu bar.

Alternatively, double-clicking on any file created by TETrans (parameter file, report file, 2D graph file, etc.) will start TETrans and load in that file.

THE FILE MENU

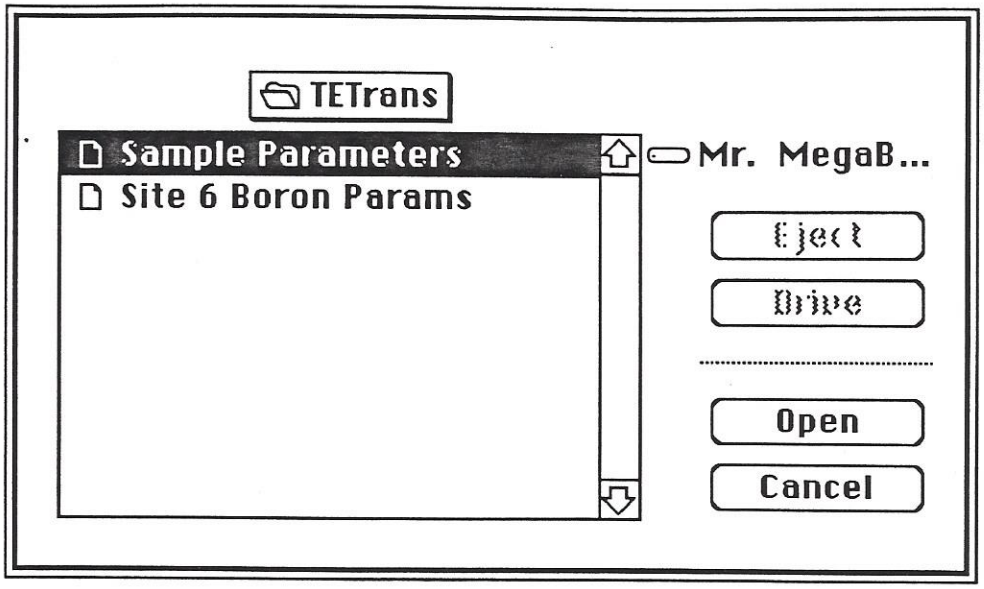
Use the **File** menu commands for starting a new TETrans input parameter file, opening an existing parameter file, saving the current parameters to a file, or viewing the name of the current input parameters file. In addition, use the **File** menu commands for saving a Simulation Report, Mass-Balance Report, 2D graph or 3D graph window, opening a previously saved window, printing a window, and quitting TETrans.

New Parameters

Choosing **New Parameters** sets all of TETrans' input parameter values to their initial or default values. A dialog will give you the chance to save the current parameters if they have been modified since they were last opened or saved, or if they have never been saved to disk. Any open Boundary Condition or Soil pH spreadsheets will be closed.

Open Parameters...

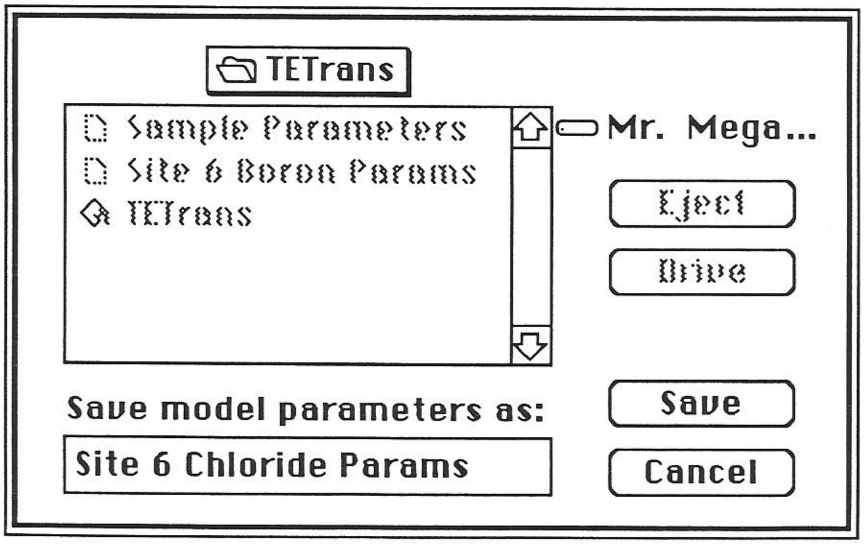
Use this command to open an existing TETrans input parameter file. A standard file opening dialog box will appear:



Only TETrans parameter files will appear in the list of files. Open the disk and folder that contains the parameter file you want, then double-click on the file name to open it.

Save Parameters...

Using this command allows you to save the current input parameter values to a disk file. A dialog box appears:



Save model parameters as: Site 6 Chloride Params

Select the disk and folder where you would like the file to be saved.

Then type your parameter file name in the "Save As" box. The name defaults to the last opened or saved parameter file, or "Untitled" if no parameter file has been opened or saved. When you are finished click the "Save" button.

Current Parameters

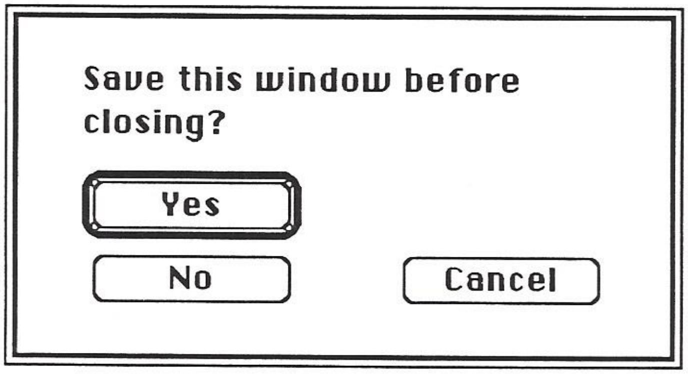
Choose **Current Parameters** when you want to see the name of the parameter file that was last opened (using the **Open Parameters...** menu command). A sub-menu will show the file's name, or "Untitled" if no parameter file was opened.

Open... (⌘O)

Choose this command when you want to open a previously saved Simulation Report, Mass-Balance Report, 2D graph or 3D graph. Select **Open** from the **File** menu, find the disk and folder of the file you wish to open, and double-click on the file's name to open it.

Close... (⌘W)

Use this command to close the active window. To close a window, choose **Close** from the **File** menu. If you are closing a window that has not been saved to disk, a dialog box will appear asking if you want to save the window:



Save this window before closing?

Click "Yes" to save the window. A **Save As** dialog box will appear. Type the name of your file in the "Save As" box and click "Save". See "**Save As**" below for more information.

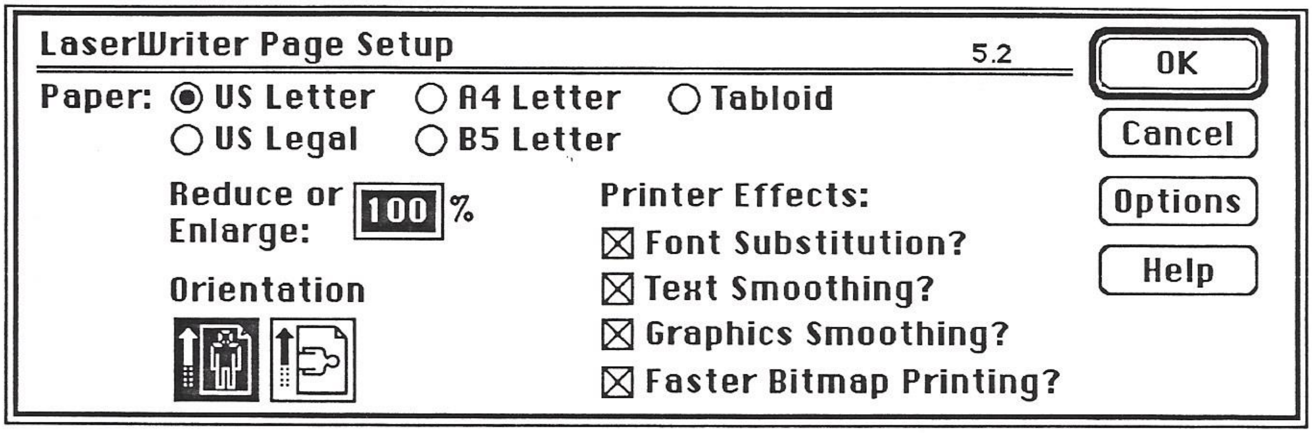
Click "No" to discard the window or click "Cancel" if you have changed your mind and don't want to close the window.

Save As... (⌘S)

The **Save As** command in the **File** menu lets you save the active window to disk. A dialog box will appear. Select the disk and folder where you would like the file to be saved. Type your file name in the "Save As" box and click on the "Save" button. If the file name you type already exists, an alert box will warn you and give you the chance to change your file name.

Page Setup...

Use this command to set the print characteristics of your connected printer. For example, choosing **Page Setup** with an Apple LaserWriter attached will display the following dialog:



LaserWriter Page Setup 5.2

Paper: ● US Letter ○ A4 Letter ○ Tabloid
○ US Legal ○ B5 Letter

Reduce or Enlarge: 100 %

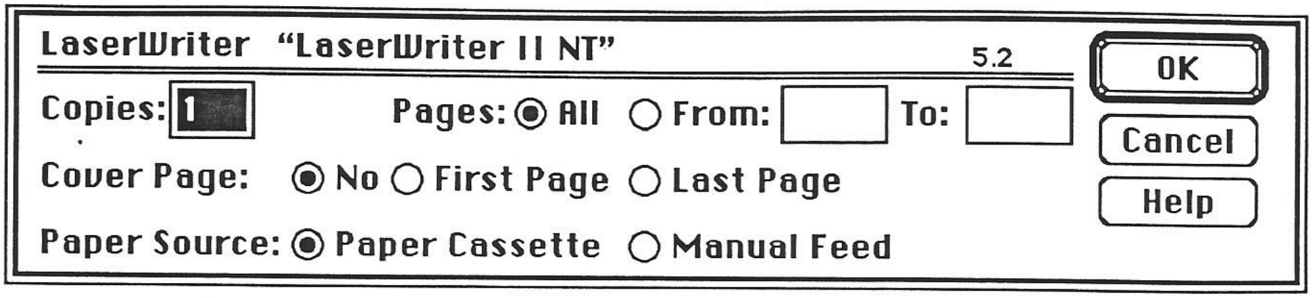
Orientation

Printer Effects:
☒ Font Substitution?
☒ Text Smoothing?
☒ Graphics Smoothing?
☒ Faster Bitmap Printing?

A variety of page size and print quality options can be set. Choosing "OK" accepts any changes you make. The actual dialog you see depends on the printer your are using. Consult your printer's documentation for more details.

Print... (⌘P)

Choose **Print** to print the active window. The following dialog appears if your printer is an Apple LaserWriter:



LaserWriter "LaserWriter II NT" 5.2

Copies: 1   Pages: ● All ○ From: □ To: □

Cover Page: ● No ○ First Page ○ Last Page

Paper Source: ● Paper Cassette ○ Manual Feed

Clicking the "OK" button starts the printing job. The actual dialog you see depends on the printer you are using. Consult your printer's documentation for more details.

Quit (⌘Q)

Choose Quit when you wish to quit TETrans and return to the Finder. If any unsaved windows are open, a dialog box will appear giving you a chance to save each window before quitting.

THE EDIT MENU

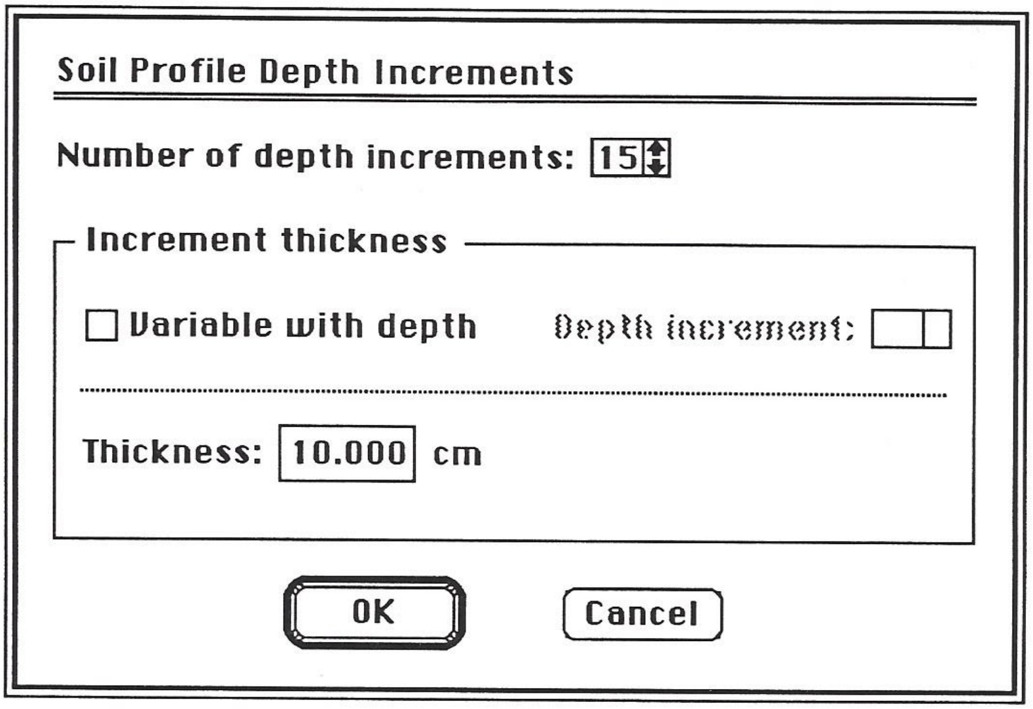
The **Edit** menu only supports desk accessory editing. In the future it will also support the editing of the spreadsheet windows, as well as the copying of reports and graphs created in TETrans to and from the Macintosh's clipboard. This will allow the copying of the simulation results and graphs to your word processor or drawing programs. A future **Table** item in the **Output** menu will allow the display of TETrans' results in tables, which could then be copied to CricketGraph™ for graphing.

THE PARAMETERS MENU

Use the **Parameters** menu to enter values for the TETrans input parameters, such as the soil bulk density, number of crops, etc. The values entered here may be saved at any time using the **Save Parameters As...** command in the **File** menu. See Chapter 2: Introduction to TETrans for an overview of the physical meaning of the less obvious input parameters.

Depth Increments...

The **Depth Increments** command allows you to set the number and thickness of each layer in your soil profile. When you choose this command, the following dialog box appears:



Soil Profile Depth Increments

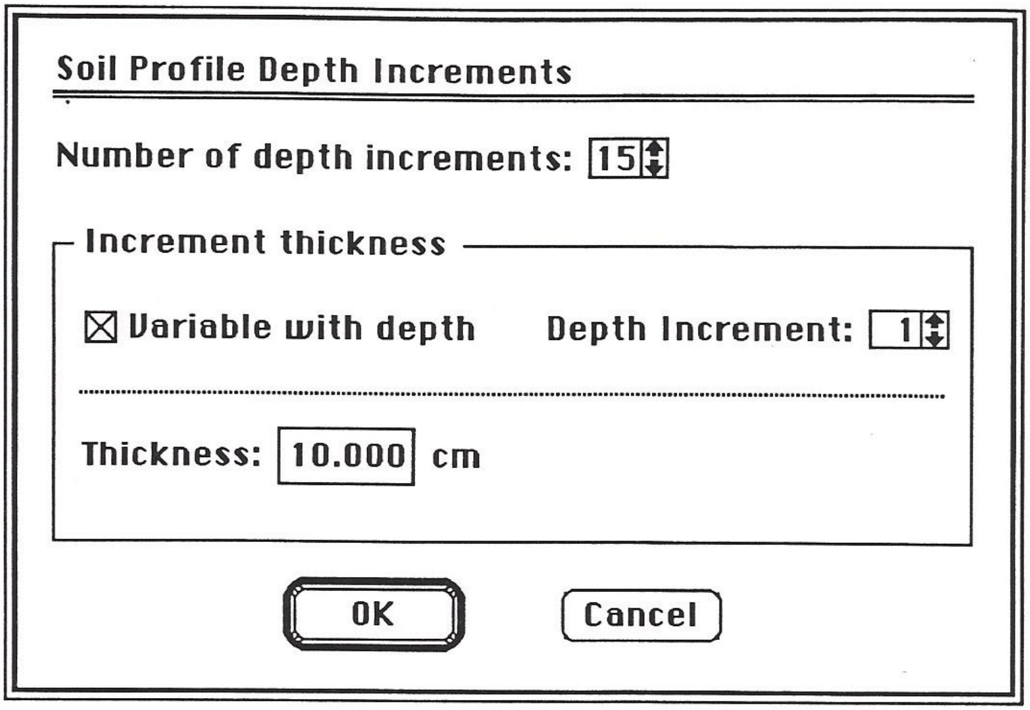
Number of depth increments: 15

Increment thickness

□ Variable with depth   Depth increment:

Thickness: 10.000 cm

Click on the up- or down-arrow buttons to change the number of soil profile depth increments or layers in your profile. Then type in the desired thickness in the "Thickness:" text box. Even if the soil is homogeneous with no change in soil properties, you will still want to enter more than one depth increment because the model simulation output is based on the number and thickness of depth increments entered in this dialog box. As a result, select how you want the solute concentration distribution profile output to appear by entering the number of depth increments and the constant thickness that each depth increment will have. If your profile is divided into layers of varying thickness, check the "Variable with depth" box. The dialog box will now look like this:



Soil Profile Depth Increments

Number of depth increments: 15

Increment thickness

☒ Variable with depth

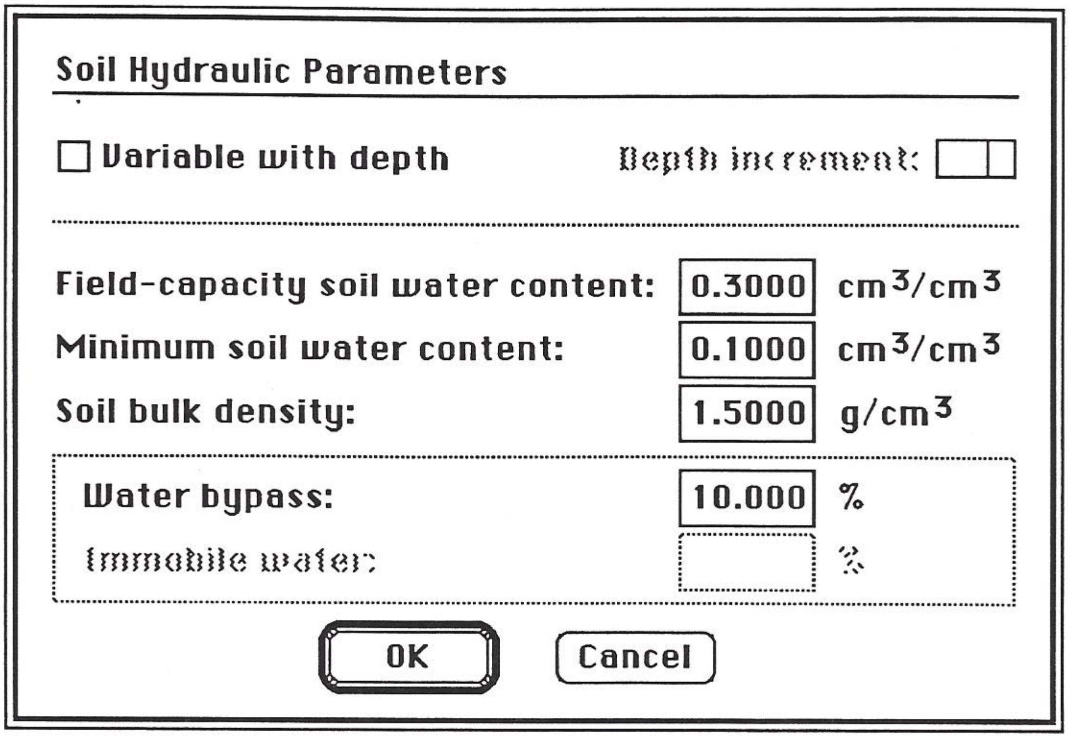
Depth Increment: 1

Thickness: 10.000 cm

The currently displayed soil profile depth increment is 1, it has a thickness of 10.0 cm and there are a total of 15 separate layers or depth increments. Fewer than 15 depth increments can be specified by using the "Number of Depth Increments:" down arrow button. Click on either the up or down "Depth Increment:" arrow button to change the currently displayed depth increment. Using the "Depth Increment:" and "Thickness:" text boxes, desired values for each of the depth increment's can be entered. Clicking on the "OK" button dismisses the dialog and accepts the values. Clicking "Cancel" dismisses the dialog without saving the changes.

Hydraulics...

Choose **Hydraulics** to set the soil hydraulic parameters for your simulation. A dialog box will appear:



Soil Hydraulic Parameters

□ Variable with depth   Depth increment:

Field-capacity soil water content: 0.3000 cm^3/cm^3

Minimum soil water content: 0.1000 cm^3/cm^3

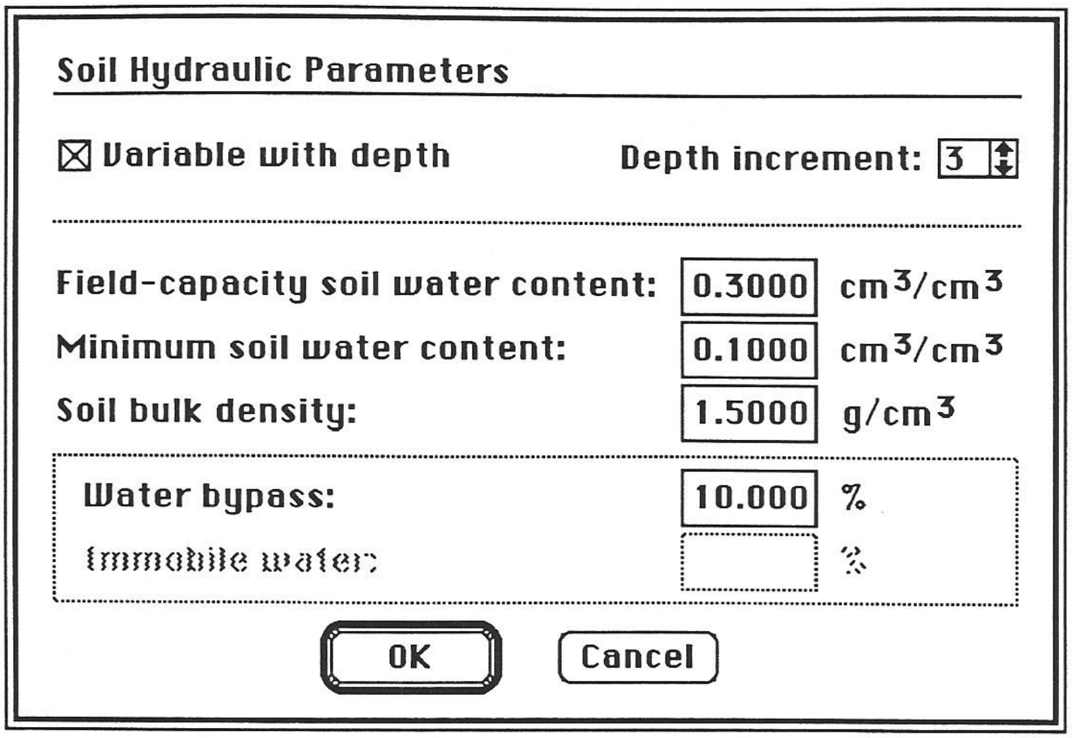
Soil bulk density: 1.5000 g/cm^3

Water bypass: 10.000 %

Immobile water: %

Text boxes allow you to type in the needed hydraulic parameters for each of the soil profile's depth layers or increments. These parameters are the soil increment's field-capacity water content, minimum water content, bulk density and percent of water bypass. See the "Input Parameters" section in Chapter 2: Introduction to TETrans for a discussion of these parameters.

By default, any parameter values you enter will apply to the entire soil profile, unless you check the "Variable with depth" box. Checking the "Variable with depth" box allows you to enter different values for each of the depth increments in the soil profile:



Soil Hydraulic Parameters

☒ Variable with depth

Depth increment: 3

Field-capacity soil water content: 0.3000 cm^3/cm^3

Minimum soil water content: 0.1000 cm^3/cm^3

Soil bulk density: 1.5000 g/cm^3

Water bypass: 10.000 %

Immobile water: %

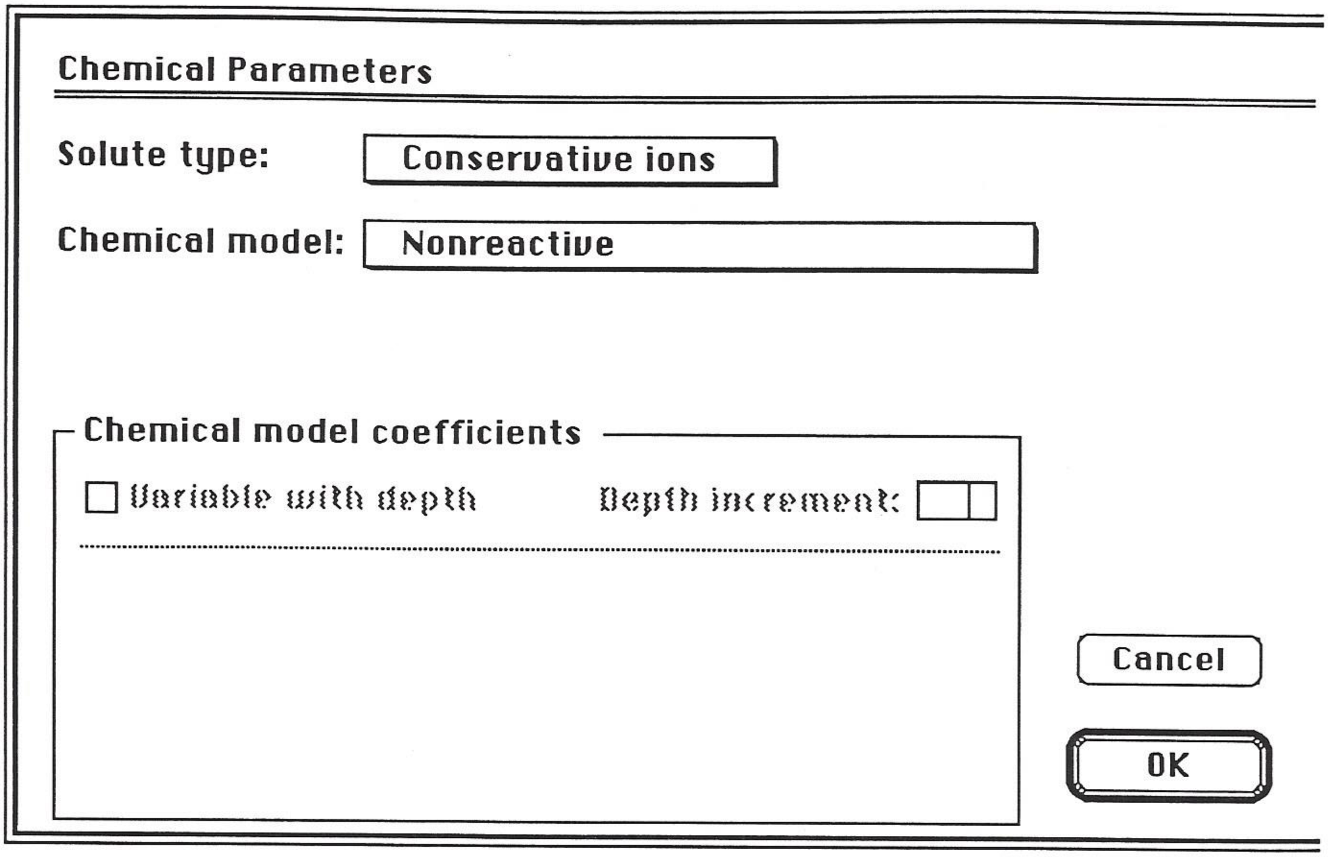
See the previous "Depth Increments" section to learn about how to set the number and thickness of the the depth increments in your soil profile. In the dialog shown above, the currently displayed values are for the third depth increment. Click the "Depth increment:" up or down arrows to select a particular depth increment, then type the hydraulic parameter values in the text boxes.

The "Immobile water" text box is inactive (i.e., it appears grayed out). This feature will be part of a future version of TETrans incorporating an optional mobile-immobile phase solute transport model (see "Input Parameters" section of Chapter 2: Introduction to TETrans).

Click "OK" to accept your entered values and to dismiss the dialog box.

Chemistry...

Use this command to select the type of solute you wish to model (e.g. conservative ions, organic, etc.), as well as which chemical model for TETrans to use in the simulation (e.g. linear adsorption, Langmuir adsorption, etc.). When you choose **Chemistry** from the **Parameters** menu you are shown the following dialog box:



Chemical Parameters

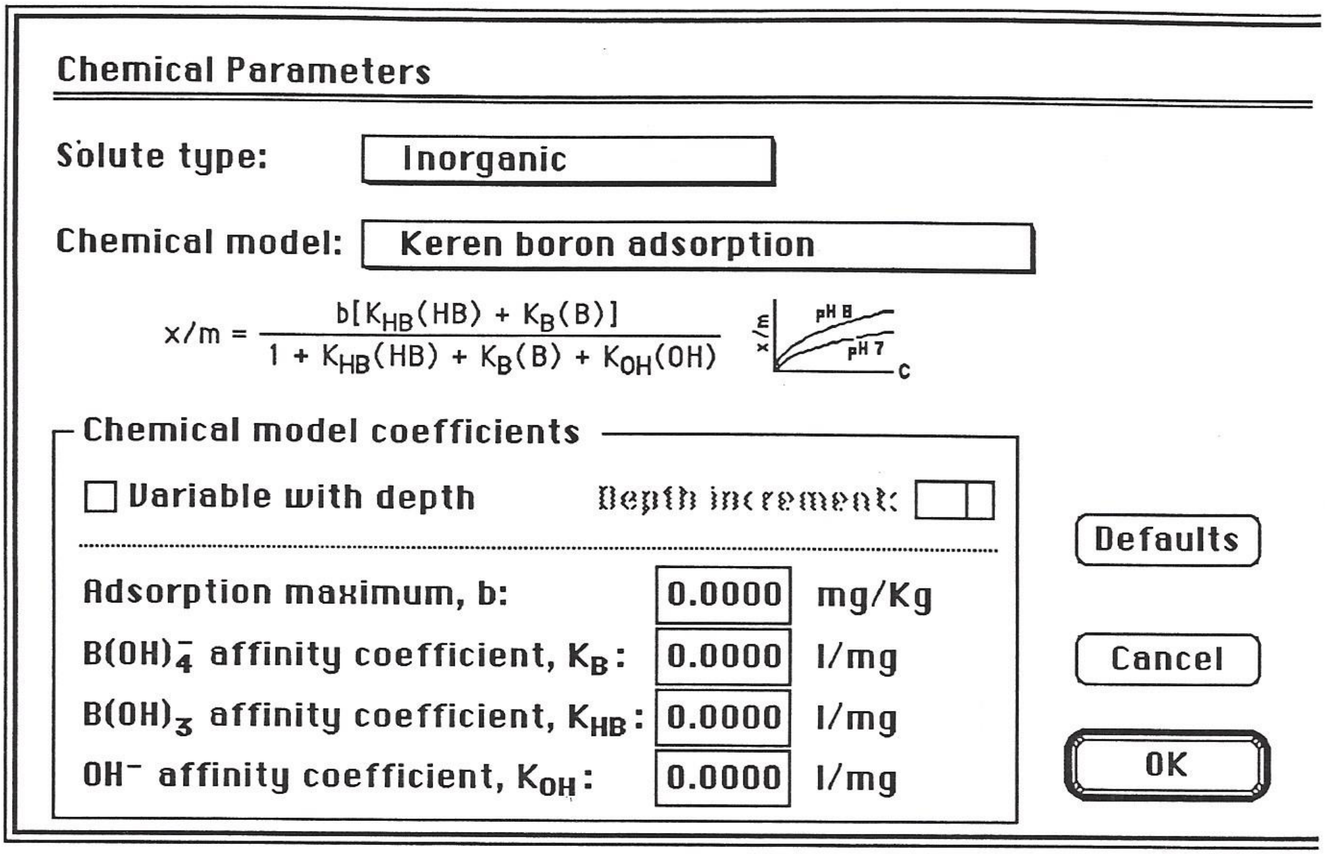
Solute type: Conservative ions

Chemical model: Nonreactive

Chemical model coefficients

□ Variable with depth   Depth increment: □

A pop-up menu for "Solute type:" allows you to select from conservative ions, major salt ions, organics or inorganics. Holding the mouse button down while the cursor is within the "Solute type:" box will reveal the pop-up menu of solute types available. To make your selection, drag the mouse with the button held down to the desired solute-type selection and then release the button. For each of these selections there are one or more possible chemical models that TETrans can use. Similarly, the pop-up menu for the various chemical models available is shown when the mouse button is held down while the cursor is within the "Chemical model:" box. Selection of the desired chemical model is done the same as for selecting the desired solute type, except only those chemical models that are highlighted are available for that solute type. For conservative ions there can be either the nonreactive model (e.g., chloride) or the anion exclusion model. For major salt ions there is the cation exchange model. For organics and inorganics there are the Freundlich (linear) and Langmuir adsorption models. In addition, the Keren adsorption model is included for the inorganic solute boron. For example, choosing **Inorganic** from the **Solute type** pop-up menu, and **Keren boron adsorption** from the **Chemical model** pop-up menu changes the dialog box to:



Chemical Parameters

Solute type: Inorganic

Chemical model: Keren boron adsorption

$$x/m = \frac{b[K_{HB}(HB) + K_B(B)]}{1 + K_{HB}(HB) + K_B(B) + K_{OH}(OH)}$$

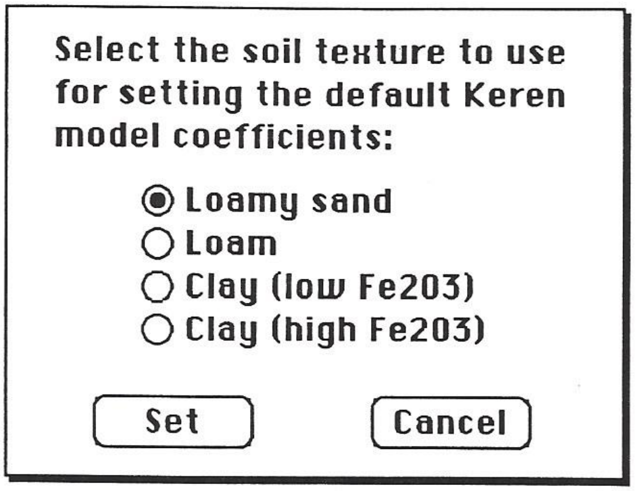
Chemical model coefficients

□ Variable with depth   Depth increment:

Adsorption maximum, b:	0.0000	mg/Kg
$B(OH)_4^-$ affinity coefficient, K_B:	0.0000	l/mg
$B(OH)_3$ affinity coefficient, K_{HB}:	0.0000	l/mg
OH^- affinity coefficient, K_{OH}:	0.0000	l/mg

The conventional description of the model's formula is displayed below the pop-up menus. The coefficients for the particular model are entered within the area labeled "Chemical model coefficients". The text boxes are where you enter the appropriate adsorption coefficients. In this particular case, you type in values for the four Keren boron adsorption coefficients: b, k_B, k_{HB} and k_{OH}. By default, any values you enter will apply to the entire soil profile. Checking the "Variable with depth" box allows you to enter different coefficient values for each of the depth increments in the soil profile. This is done in the same manner as for the hydraulic parameters, discussed above.

For the Keren boron adsorption model it is possible to assign default values to these coefficients. Click the "Defaults" button to display the following dialog box:



Select the soil texture to use for setting the default Keren model coefficients:

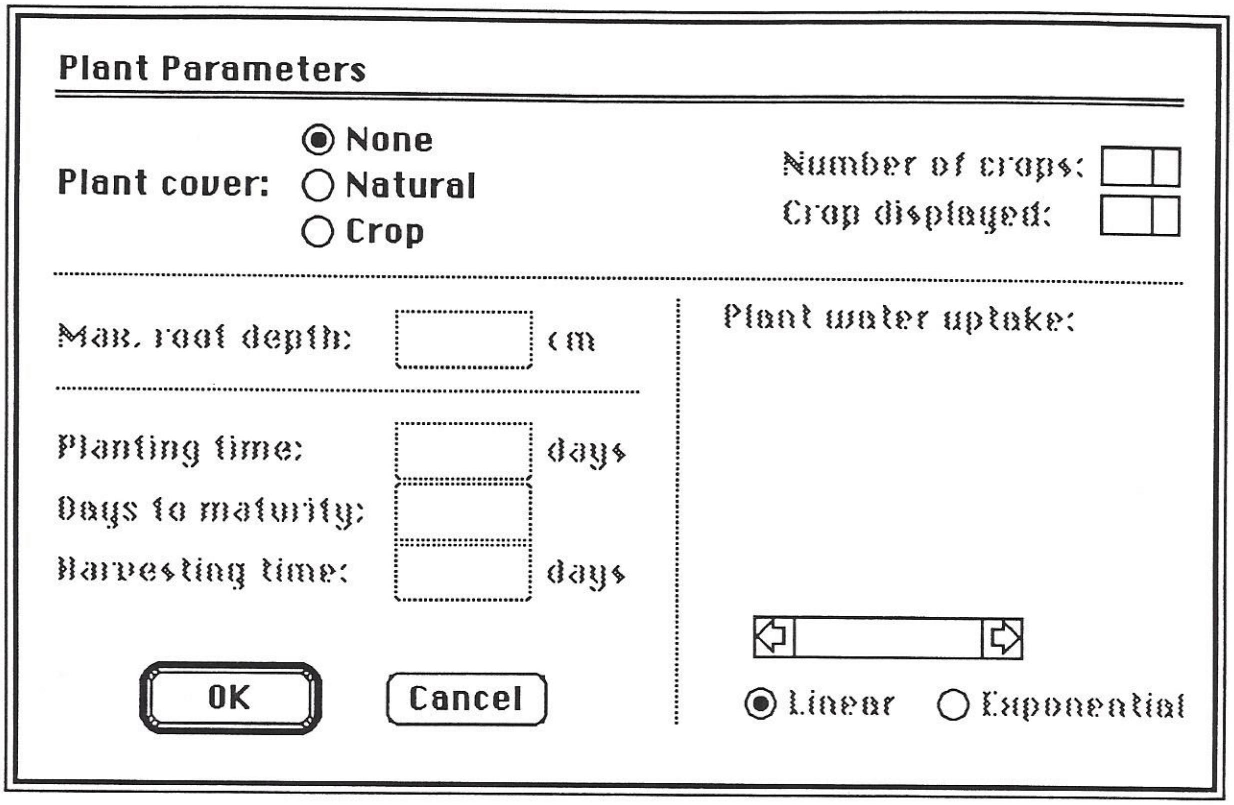
● Loamy sand
○ Loam
○ Clay (low Fe203)
○ Clay (high Fe203)

Here boron adsorption coefficient values are set according to the selected texture of the soil layer or entire soil profile. Click the appropriate radio button for the most suitable soil texture: loamy sand, loam, clay low in iron or clay high in iron. Then click the "Set" button to apply these values. If "Variable with depth" is checked in the chemical parameters dialog, these values will be applied only to the currently selected depth increment.

When you have set your desired solute type, chemical model and chemical model coefficients, click the "OK" button.

Plants...

Use this command to set the plant or crop characteristics for your simulation. Three plant conditions are possible: no plant cover, natural plant cover, or crops. No plant cover indicates fallow conditions. Natural plant cover applies to areas of natural vegetative cover, such as grasslands, chaparral, etc. Crops simply refers to man-planted crops. Choosing **Plants** gives the following dialog:



Plant Parameters

Plant cover: ● None ○ Natural ○ Crop

Number of crops:

Crop displayed:

Max. root depth: cm

Planting time: days

Days to maturity:

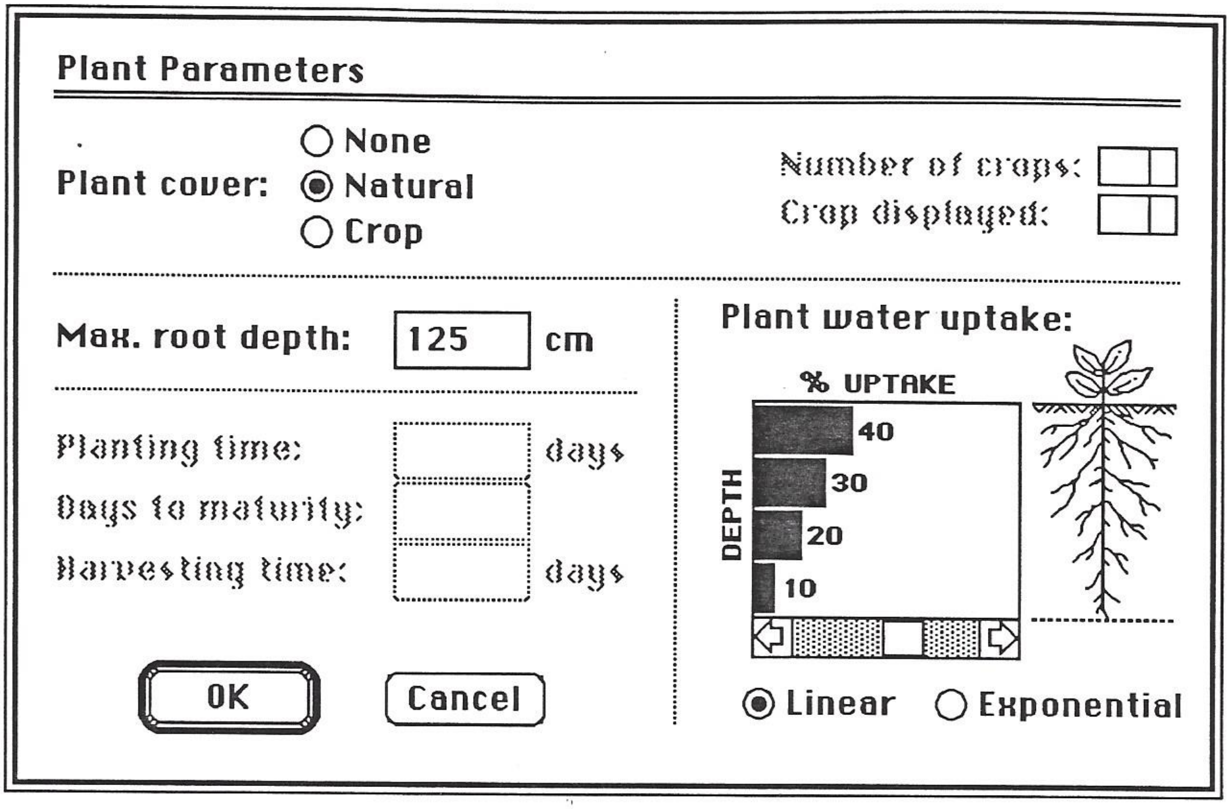
Harvesting time: days

Plant water uptake:

● Linear ○ Exponential

Here the plant cover is set to "None". For this case, any evapotranspiration entered in the **Boundary Condition** spreadsheet (see "Boundary Condition..." subsection of this section) will be taken as evaporation only from the surface of the soil.

Clicking the "Natural" plant cover button changes the dialog to:



Plant Parameters

Plant cover: ○ None ● Natural ○ Crop

Number of crops: □

Crop displayed: □

Max. root depth: 125 cm

Planting time: □ days

Days to maturity: □

Harvesting time: □ days

OK Cancel

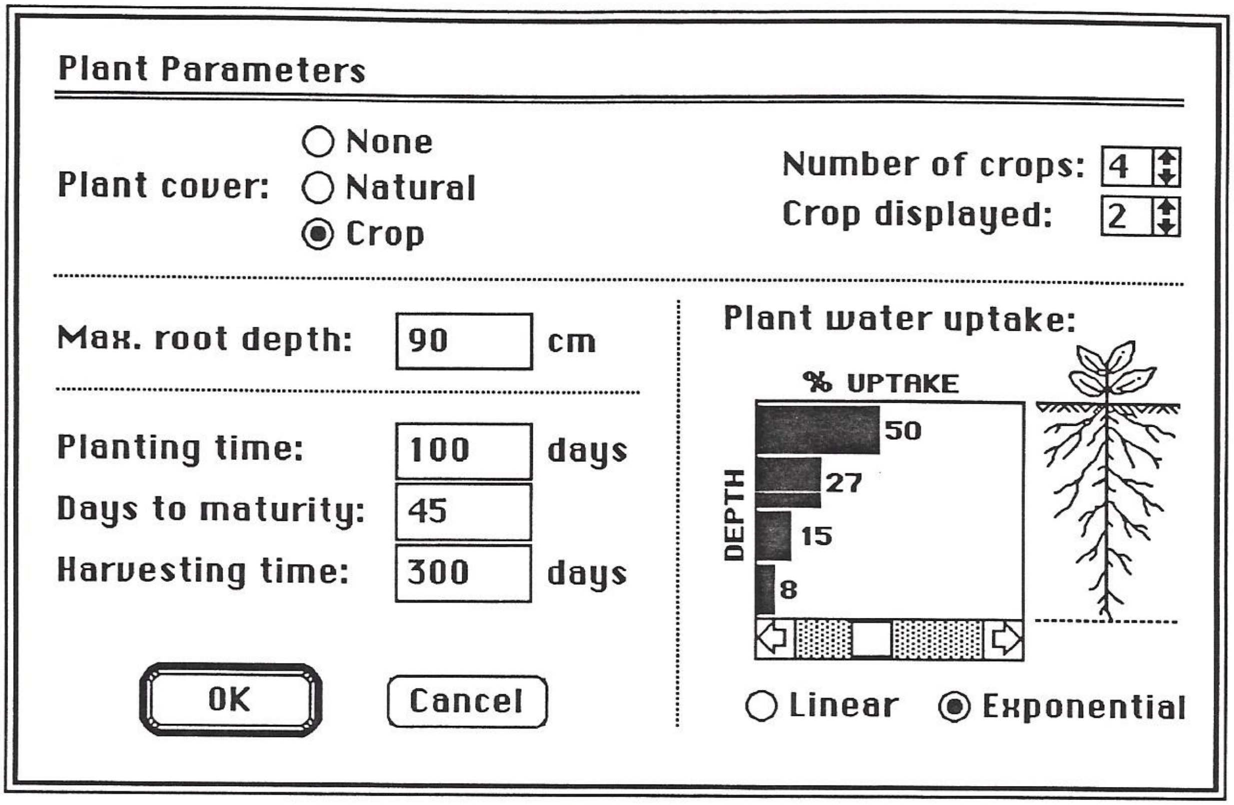
Plant water uptake:

● Linear ○ Exponential

The plant cover's maximum root depth is entered in the "Max. root depth:" text box. Entering zero for this value is the same as choosing no plant cover.

Use the "Plant water uptake:" scroll bar to set the plant's root water uptake profile. As you change the scroll bar's value, the bar graph will show the amount of water taken up over the root profile. For instance, the above graph shows a 40-30-20-10 uptake profile. This means that 40% of the plant's water uptake is in the upper one-fourth of it's root zone, 30% in it's second, and so on. Clicking the "Linear" or "Exponential" buttons changes the shape of the water uptake profile.

Clicking the "Crop" plant cover button changes the dialog to:



Plant Parameters

Plant cover: ○ None ○ Natural ● Crop

Number of crops: 4

Crop displayed: 2

Max. root depth: 90 cm

Planting time: 100 days

Days to maturity: 45

Harvesting time: 300 days

Plant water uptake:

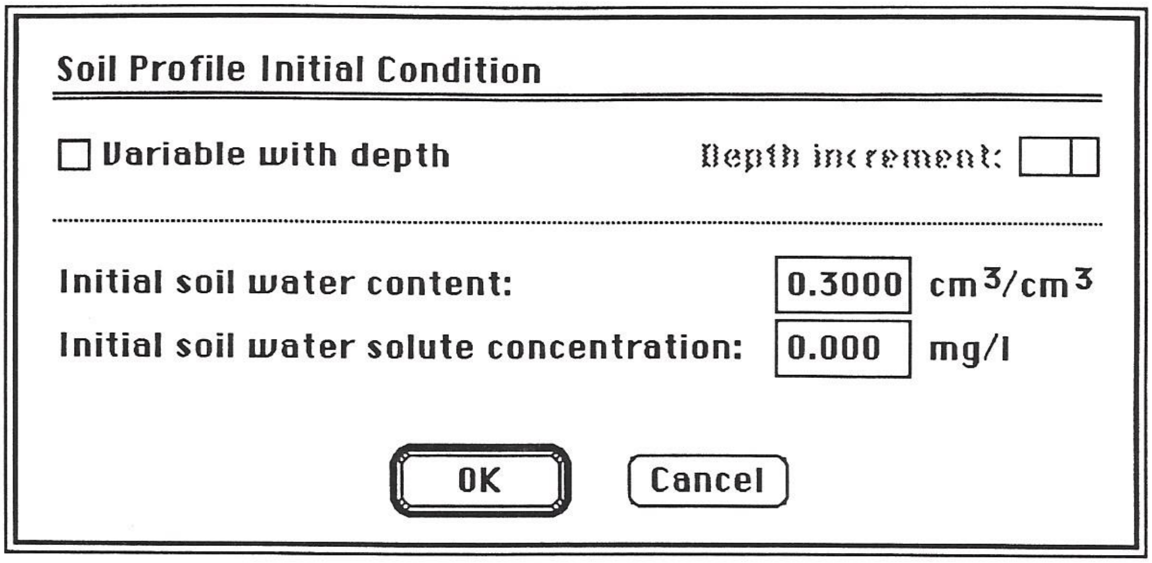
○ Linear ● Exponential

Use the "Number of crops:" up and down arrows to set the number of crops to be used in your simulation. The crop parameters displayed in the dialog are for the crop number indicated in the "Crop displayed:" box.

In addition to the maximum root depth and uptake profile, each crop must have its planting time, number of days to maturity and harvesting time entered in the appropriate text boxes. To enter the next crop's values, use the "Crop displayed:" up-arrow button to display the next crop, then enter its values. At any time you may change the displayed crop by clicking on the "Crop displayed:" up- or down-arrow buttons. Click "OK" when you are through.

Initial Condition...

Use this command to set the soil profile's initial water content and solute concentration. Choosing **Initial Condition** brings up the following dialog:



Soil Profile Initial Condition

□ Variable with depth

Depth increment:

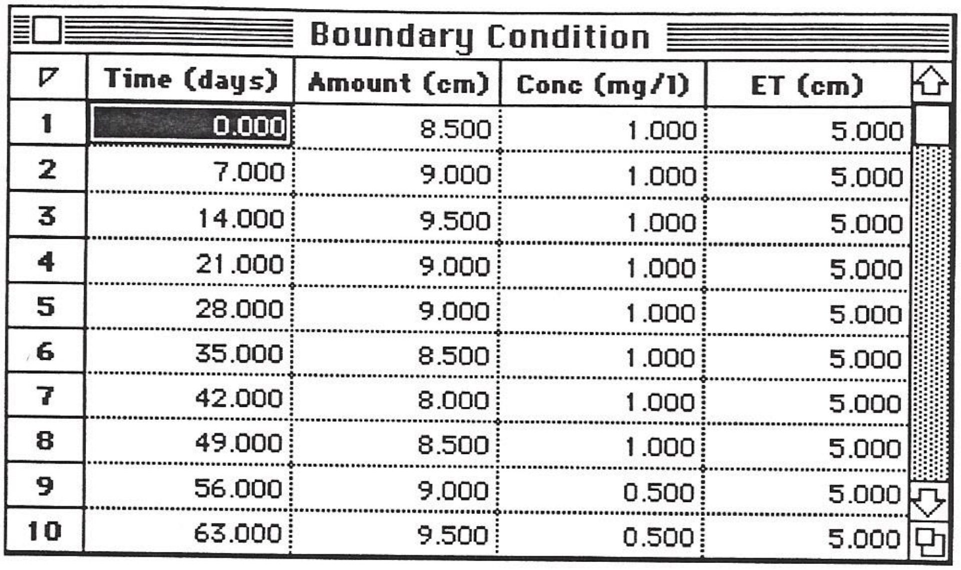
Initial soil water content: 0.3000 cm^3/cm^3

Initial soil water solute concentration: 0.000 mg/l

Enter the initial soil water content, and the associated concentration of the solute that you are simulating, in the appropriate text boxes. Checking the "Variable with depth" box allows you to enter different initial values for each of the depth increments in the soil profile. In this way you can establish the initial soil water content and the initial solute concentration profiles needed as the starting point for TETrans. This is done in the same manner as for the hydraulic parameters, discussed above. Click on the "OK" button when finished.

Boundary Condition...

Choose the **Boundary Condition** command to enter the soil surface boundary conditions. These parameters describe the amount of water and solute entering and leaving the surface of the soil over the course of the simulation, and they are absolutely essential. Choosing this command brings up a spreadsheet window:



Boundary Condition

▽	Time (days)	Amount (cm)	Conc (mg/l)	ET (cm)
1	0.000	8.500	1.000	5.000
2	7.000	9.000	1.000	5.000
3	14.000	9.500	1.000	5.000
4	21.000	9.000	1.000	5.000
5	28.000	9.000	1.000	5.000
6	35.000	8.500	1.000	5.000
7	42.000	8.000	1.000	5.000
8	49.000	8.500	1.000	5.000
9	56.000	9.000	0.500	5.000
10	63.000	9.500	0.500	5.000

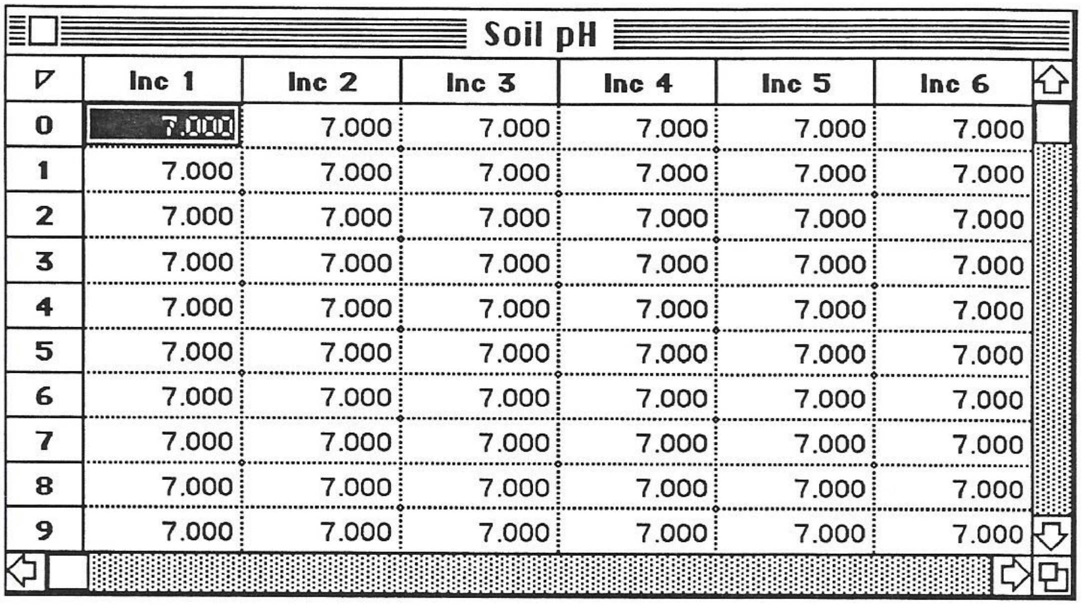
The first column is the number of the irrigation or precipitation event. The time of an irrigation or precipitation event is entered in the **Time** column. The amount of irrigation water applied or precipitation that fell at this time is entered in the **Amount** column. The solute concentration (of the solute being modeled only) of the irrigation water is entered in the **Conc** column. Of course, for precipitation a concentration of zero would be the most likely value. The evapotranspiration is entered in the **ET** column. This is taken to be the total amount of evapotranspiration occurring from the time of the irrigation or precipitation event until just prior to the next event. For example, the ET for event 3 would be the total amount of evapotranspiration that occurred from the time of event 3 to the time of event 4, a period of seven days in this case.

Use the arrow keys or mouse to select different cells or to move about in the spreadsheet. The spreadsheet may be grown vertically to the full extent of your screen.

NOTE: Make sure that the entered events are sorted in order of increasing time. This version of TETrans will not sort them for you.

Soil pH...

This command is only available when the Keren model of boron adsorption is chosen, and is used to enter the soil pH values for each soil profile depth increment of each irrigation event. Refer to the "Chemistry..." subsection above to see how to choose the Keren chemical model. Choosing the **pH** command brings up the spreadsheet:



Soil pH

	Inc 1	Inc 2	Inc 3	Inc 4	Inc 5	Inc 6
0	7.000	7.000	7.000	7.000	7.000	7.000
1	7.000	7.000	7.000	7.000	7.000	7.000
2	7.000	7.000	7.000	7.000	7.000	7.000
3	7.000	7.000	7.000	7.000	7.000	7.000
4	7.000	7.000	7.000	7.000	7.000	7.000
5	7.000	7.000	7.000	7.000	7.000	7.000
6	7.000	7.000	7.000	7.000	7.000	7.000
7	7.000	7.000	7.000	7.000	7.000	7.000
8	7.000	7.000	7.000	7.000	7.000	7.000
9	7.000	7.000	7.000	7.000	7.000	7.000

Each column, titled **Inc 1**, **Inc 2**, etc., corresponds to a soil profile depth increment, as set in the "Depth increments..." section above. NOTE: The first row, **0**, is the initial pH distribution before any irrigations or precipitations have occurred. This is an esssential inital condition for the Keren boron adsorption model. The subsequent rows are the pH distributions at field capacity following each irrigation or precipitation event. Therefore, row **1** corresponds to irrigation or precipitation event 1, row **2** corresponds to event 2, and so on. (The values for irrigation or precipitation amounts should have already been entered using the **Boundary Condition** command from the **Parameters** menu). pH values are set to a default value of 7, if you choose not to enter all of the values.

THE RUN MENU

Run (⌘R)

Run starts the TETrans simulation, using the current input parameters.

A text window will appear with the simulation results for all of the irrigation/precipitation events entered in the **Boundary Condition** spreadsheet, and for all of the depths entered in the **Depth Increments** dialog.

NOTE: Currently, the text windows showing the simulation results are limited to 32K of text. If you find that the results for some of your events are not shown in the Simulation Report window, use the **Output Depths** and/or **Output Times** dialogs in the **Output** menu to limit the amount of results displayed. Then choose **Simulation Report** from the **Output** menu to make a new window with less text. This flaw will be corrected in a future version.

To check the name of the current parameter file choose **Current Parameters** from the **File** menu. A sub-menu will pop out with the name of the current parameters (or "Untitled", if the current parameters have not been saved).

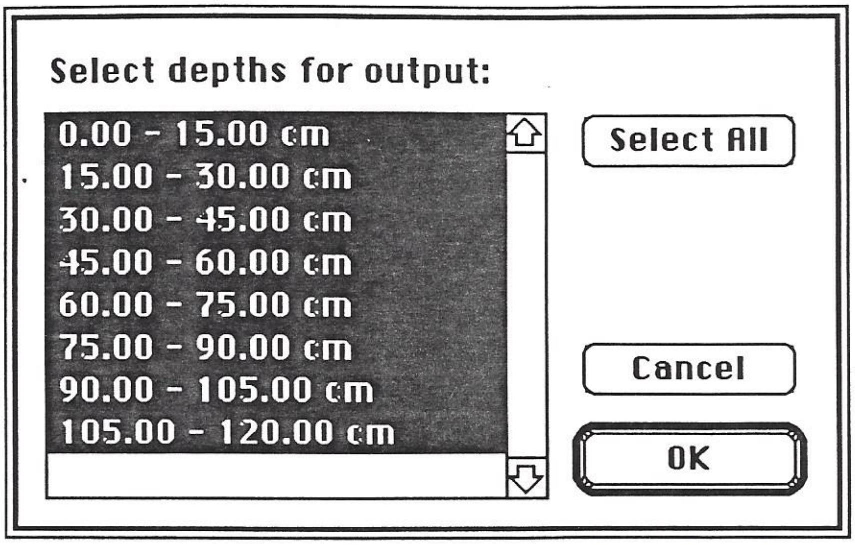
You may abort any simulation before it is done by holding down the command key (⌘) and typing a period (.).

THE OUTPUT MENU

Use the **Output** menu to create text reports showing simulation results and mass-balance information, as well as to create two- and three-dimensional graphs of simulation results and input values.

Output Depths...

Choose this option to select or limit the output depths for simulation results displayed in the simulation report, mass-balance report and 2D graph windows. Choosing **Output Depths** displays this dialog box:



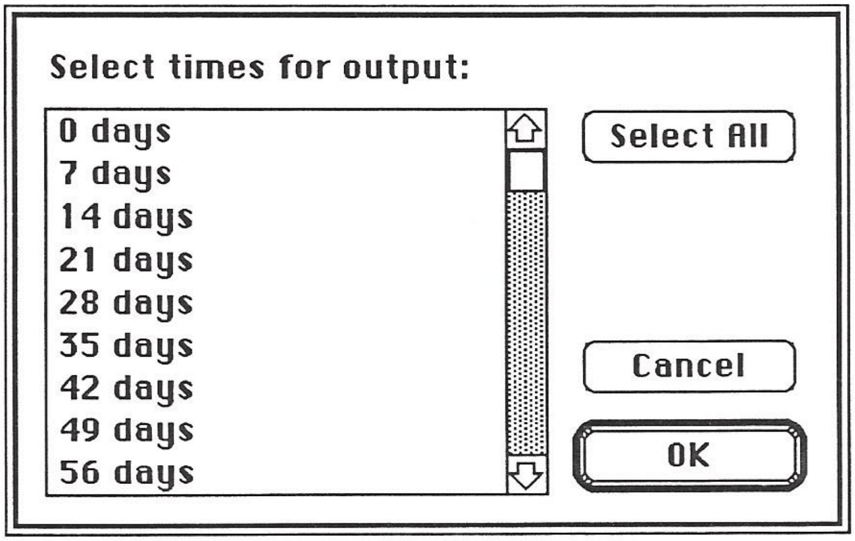
Select depths for output:

0.00 - 15.00 cm
15.00 - 30.00 cm
30.00 - 45.00 cm
45.00 - 60.00 cm
60.00 - 75.00 cm
75.00 - 90.00 cm
90.00 - 105.00 cm
105.00 - 120.00 cm

The depth increments, set from the **Depth Increments** command in the **Parameters** menu, are shown in a scrollable list. Clicking on any given depth will select it for output. Clicking on it again will deselect it. By clicking the mouse and dragging, multiple depths can be selected or deselected. Clicking the "Select All" button selects all of the depths for output. Clicking "OK" saves your selections and dismisses the dialog box.

Output Times...

Choose this option to select or limit the output times for simulation results displayed in the simulation report, mass-balance report and 2D graph windows. Choosing **Output Times** displays this dialog box:



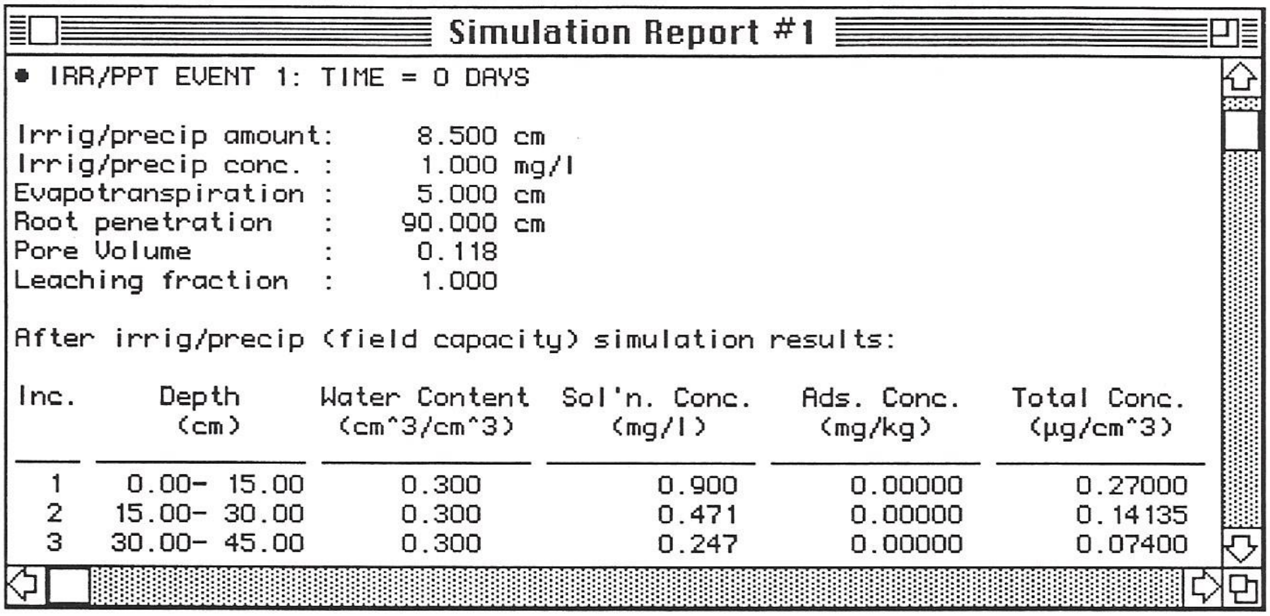
Select times for output:

0 days
7 days
14 days
21 days
28 days
35 days
42 days
49 days
56 days

The event times, from the events entered in the **Boundary Condition** spreadsheet of the **Parameters** menu, are shown in a scrollable list. Clicking on any given time will select that time for output. Clicking on it again will deselect it. By clicking the mouse and dragging, multiple times can be selected or deselected rapidly. Clicking the "Select All" button selects all of the times for output. Clicking "OK" saves your selections and dismisses the dialog box.

Simulation Report

Choose this option to get a listing of the simulated solute and water content values from the most recently run simulation. Choosing Simulation Report will display a window like this:



```
● IRR/PPT EVENT 1: TIME = 0 DAYS

Irrig/precip amount:      8.500 cm
Irrig/precip conc. :      1.000 mg/l
Evapotranspiration :      5.000 cm
Root penetration   :     90.000 cm
Pore Volume        :      0.118
Leaching fraction  :      1.000

After irrig/precip (field capacity) simulation results:

Inc.      Depth       Water Content   Sol'n. Conc.     Ads. Conc.     Total Conc.
           (cm)       (cm^3/cm^3)       (mg/l)          (mg/kg)       (µg/cm^3)
____   _____________  _____________  _____________  _____________  _____________
  1     0.00- 15.00       0.300           0.900         0.00000        0.27000
  2    15.00- 30.00       0.300           0.471         0.00000        0.14135
  3    30.00- 45.00       0.300           0.247         0.00000        0.07400
```

TETrans will create this window for you automatically after any simulation. The results shown will be for the output times and depths selected in the **Output Times** and **Output Depths** dialogs (see the subsections above on "Output Times" and "Output Depths").

The following information is displayed for each irrigation (or precipitation) event:

- Amount of irrigation water or precipitation
- Irrigation water concentration for the solute being modeled
- ET

- Depth of root penetration
- Leaching fraction
- Soil water content, solution concentration, adsorbed concentration and total concentration following the irrigation or precipitation. This is at the soil's field capacity water content and the results are displayed for each depth increment selected for output.
- Root water uptake, soil water content, solution concentration and adsorbed concentration immediately before the next irrigation or precipitation event. This is at the soil's most dry state, after all of the ET occurs. The results are displayed for each depth increment selected for output.

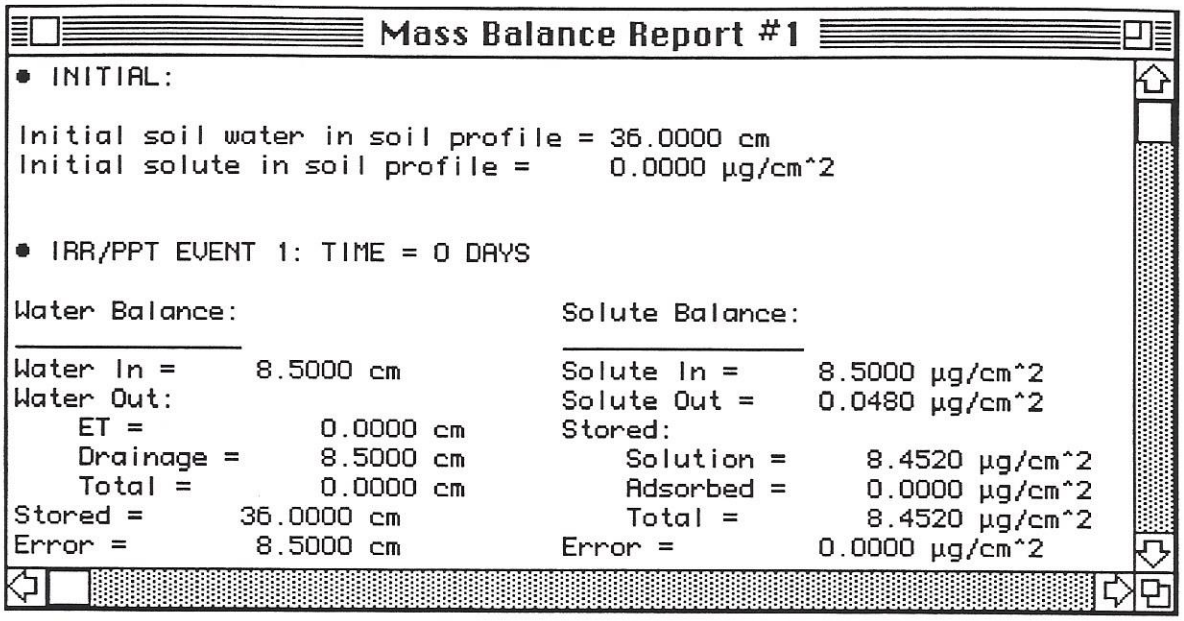
Simulation reports can be created at any time following a simulation. By using the **Output Times** and **Output Depths** dialogs, simulation results for any combination of event times and soil profile depths can be displayed.

Simulation reports are saved as ordinary text files. See the "Save As" subsection of the "File Menu" section above.

Currently, the window is limited to 32K of text. If you find that the results for some of your events are not shown in the Simulation Report window, use the **Output Depths** and/or **Output Times** dialogs in the **Output** menu to limit the amount of results displayed. Then choose **Simulation Report** from the **Output** menu to make a new window with less text. This flaw will be corrected in a future version.

Mass-Balance Report

Choose this to obtain diagnostic information about the most recently run simulation. Choosing **Mass-Balance Report** will display a window like the one below:



```
• INITIAL:

Initial soil water in soil profile = 36.0000 cm
Initial solute in soil profile =      0.0000 µg/cm^2


• IRR/PPT EVENT 1: TIME = 0 DAYS

Water Balance:                         Solute Balance:
_______________                        _______________
Water In =      8.5000 cm              Solute In =      8.5000 µg/cm^2
Water Out:                             Solute Out =     0.0480 µg/cm^2
    ET =            0.0000 cm          Stored:
    Drainage =      8.5000 cm              Solution =       8.4520 µg/cm^2
    Total =         0.0000 cm              Adsorbed =       0.0000 µg/cm^2
Stored =       36.0000 cm                  Total =          8.4520 µg/cm^2
Error =         8.5000 cm              Error =          0.0000 µg/cm^2
```

The water and solute mass-balance information for each irrigation (or precipitation) event is displayed in a scrollable, resizeable window. The water-balance information consists of the amount of water entering the soil profile surface, the amount of water leaving the soil profile (through drainage or ET) and the amount of water stored in the soil profile. The solute-balance information consists of the amount of solute entering the soil surface in the irrigation water, the amount leaving through drainage, and the amount stored in the soil profile (in both the solution and the adsorbed phases). Any appreciable error in either the water or solute balance indicates a problem with the model parameters. See Chapter 5: Troubleshooting TETrans for more information.

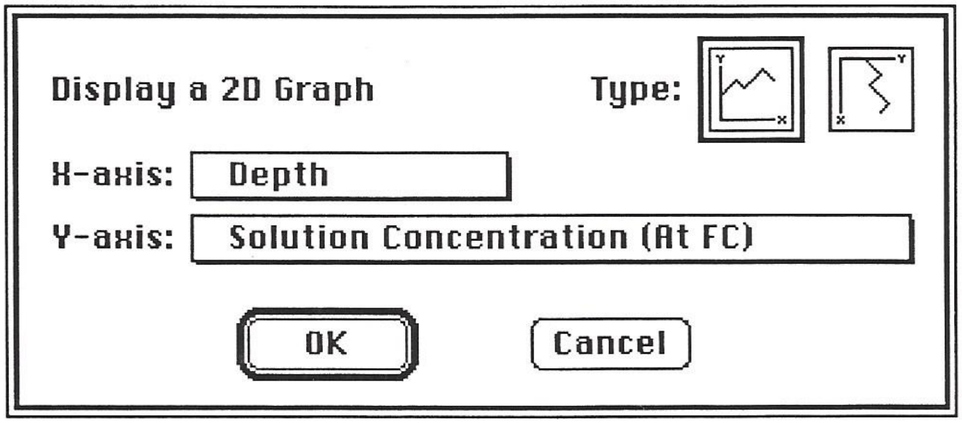
Mass-balance reports can be created at any time following a simulation. The irrigation events for which the mass-balance information are listed can be set using the **Output Times** command of the **Output** menu. See the subsection above on "Output Times" to see how to do this.

Mass-Balance reports are saved as ordinary text files. See the "Save As" subsection of the "File Menu" section above.

Currently, the window is limited to 32K of text. If you find that the results for some of your events are not shown in the Mass-Balance Report window, use the **Output Times** dialogs in the **Output** menu to limit the amount of results displayed. Then choose **Mass-Balance Report** from the **Output** menu to make a new window with less text. This flaw will be corrected in a future version.

2D Graph...

Use this command to create a two-dimensional graph of the simulation results, input parameters or mass-balance results. When you choose 2D Graph, the following dialog box appears:



Display a 2D Graph

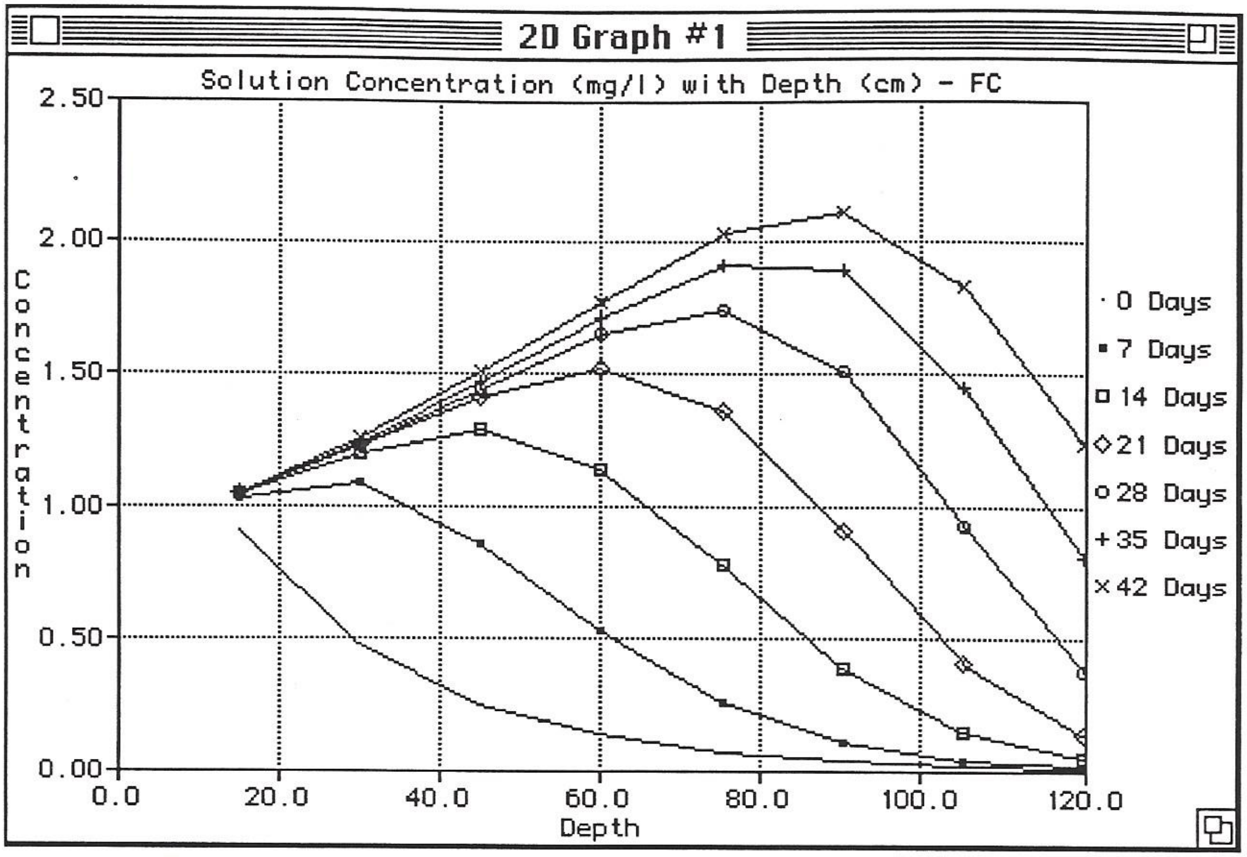
Type:

X-axis: Depth

Y-axis: Solution Concentration (At FC)

Icons representing a normal or a 90° rotated x-y graph appear in the upper-right corner. Click on the icon of the desired type to select it. The 90° rotated type is nice for graphing results that are a function of depth.

The "X-axis" pop-up menu allows you to select from "Depth", "Time" or "Pore Volume" for the x-axis variable. Holding the mouse button down while the cursor is within the "X-axis:" box will allow you to view the pop-up menu. To make your selection, drag the mouse with the button down to the desired x-axis selection and then release the button. The "Y-axis" pop-up menu allows you to select from a variety of simulation variables, such as solution concentration, adsorbed concentration, total ET, etc. Follow the same steps to view and select the y-axis as you did for the x-axis. For example, choosing "Depth" as the the x-axis variable and "Solution Concentration (At Field Capacity)" as the y-axis variable, then clicking the "OK" button, creates a graph like the one below:



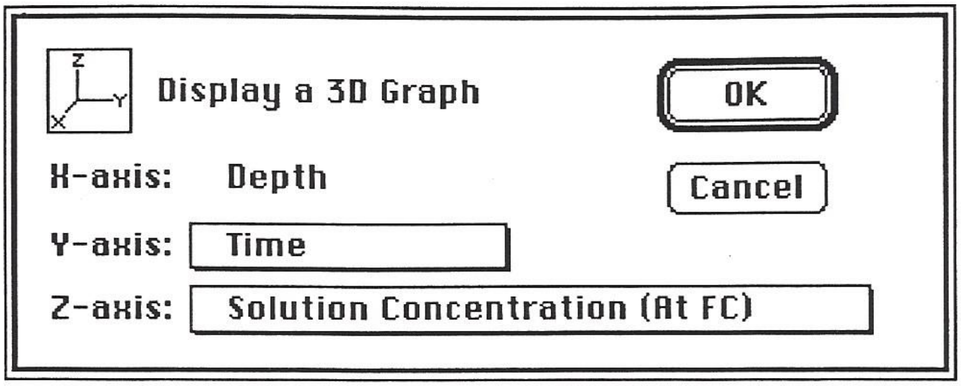
The actual values depend on your simulation parameters, of course. The number of curves displayed in a graph depends on the y-axis variable and the output times and depths chosen in the **Output Times** or **Output Depths** dialogs (see the "Output Times" or "Output Depths" sections above).

The window can be resized by clicking on the grow icon in the upper-right corner of the window or by using the mouse to drag the resize icon in the lower righthand corner.

2D graphs are saved in a unique format that only TETrans can read. See the "Save As" subsection of the "File Menu" section above. The next version of TETrans will allow saving graphs as PICT files, as well as copying graphs to the clipboard for export to other applications.

3D Graph...

Use this command to create a 3D graph of the simulation results. A 3D graph can give the you a quick assessment of the movement and distribution of the solute through time. When you choose 3D Graph, the following dialog box appears:

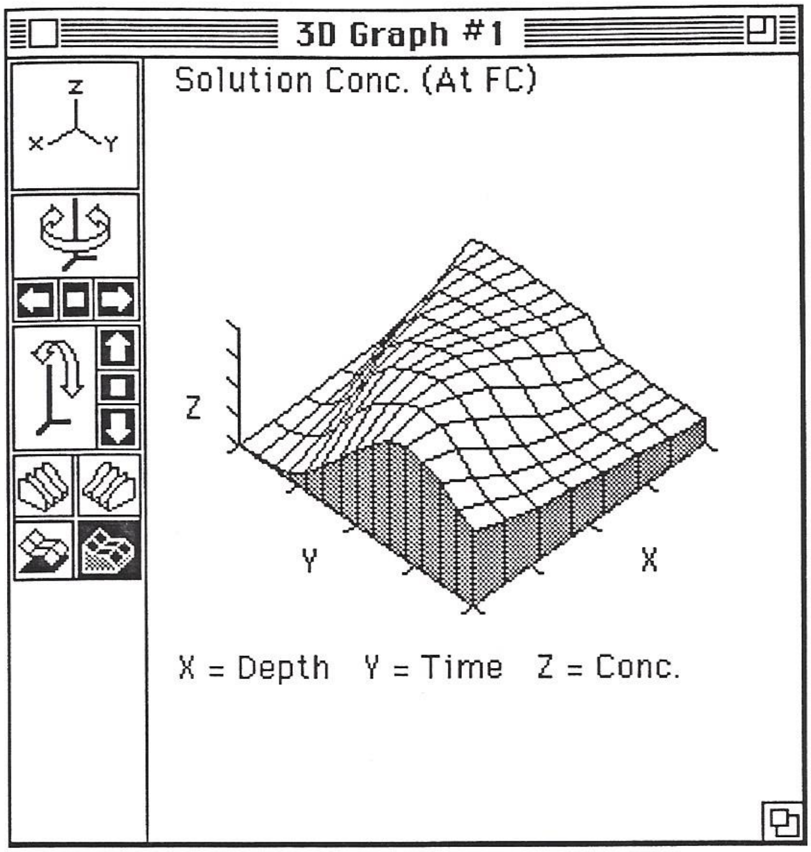


X-axis: Depth

Y-axis: Time

Z-axis: Solution Concentration (At FC)

This dialog is used to set which variables are used for each of the 3D graph's axes. The "X-axis" variable is the soil depth. The "Y-axis" pop-up menu allows you to choose from "Time" or "Pore Volume" for the y-axis variable. Holding the mouse button down while the cursor is within the "Y-axis:" box will allow you to view the y-axis pop-up menu. To make your selection, drag the mouse with the button down to the desired y-axis selection and then release the button. The "Z-axis" pop-up menu allows you to choose from a variety of simulation variables for the z-axis variable, such as solution or adsorbed concentrations, water uptake, etc. View and select the desired "Z-axis" just as you did for the y-axis. Click the "OK" button to dismiss the dialog box and create the graph. For example, the graph below was created choosing "Time" for the y-axis and "Solution Concentration (at Field-Capacity)" for the z-axis:



The view point is positioned 45° above the x-y plane. The 3D-graph can be rotated by using the controls on the left-hand side of the graph window. In addition, slices of the graph along the x- or the y-axis and freely-suspended or enclosed 3D graphs can be selected. The window can be resized by clicking on the grow icon in the upper righthand corner of the window or by using the mouse to drag the resize icon in the lower righthand corner.

3D graphs are saved in a unique format that only TETrans can read. See the "Save As" part of the "File Menu" section above. The next version of TETrans will allow saving graphs as PICT files, as well as copying graphs to the clipboard for export to other applications.

CHAPTER 5: TROUBLESHOOTING TETrans

This chapter provides some "preventative medicine" for reducing the chance of problems occuring with TETrans and an explanation of the internal error messages which are specific to the operation of TETrans.

GENERAL SUGGESTIONS

Within TETrans there are built-in error messages which are generated when physically impossible results are calculated (e.g., negative adsorption). Most error messages are sufficiently self-explanatory to be understood by the user. However, some messages are cryptic and need further discussion (see the "Error Messages" section of this chapter).

A few "preventative medicine" suggestions which can either prevent problems from occurring or help in the detection of problems are provided:

1. In some instances, the estimated plant input parameters may be incompatible with the entered evapotranspiration amounts. This will result in the loss of water mass balance. To check that this is not occurring, a mass-balance text report should be generated for each simulation. If a loss of water mass balance does occur, then the root water uptake distribution, minimum water content, field capacity and/or maximum root penetration depth are all suspect. To correct the mass-balance problem, adjust these parameters until mass balance is maintained. The maximum root penetration depth and the root water uptake distribution are the most likely parameters to be in error.

2. Because TETrans does not model upward water flow, situations where evapotranspiration significantly exceeds irrigation or precipitation can present problems. Under irrigated agriculture, evapotranspiration will seldom exceed the amount of irrigation. However, under dry land farming conditions or natural landscapes where precipitation is the primary source of water, evapotranspiration can significantly exceed precipitation. This will generate error messages by the TETrans' code and result in program termination. If it is believed that upward water flow is not significant, then the user can short circuit this problem by summing the evapotranspiration and precititation data over several precipitation and evapotranspiration events until the difference between the two no is longer substantial.

(NOTE: A precipitation event is considered to be each day that precipitation occurs, while an evapotranspiration event is considered to be the evaporation that occurs between successive precipitation events).

ERROR MESSAGES

ERROR: Divergence in Newton-Raphson Method.
Corrective action: Check the Freundlich adsorption coefficients for their correct values. If the Freundlich values are correct, then the error tolerance in the source code may be set too low. Call technical support for advice.

ERROR: Improbable scenario.
No adsorption & water content depleted to 0.
Corrective action: This error indicates that the user has selected parameters such that no adsorption is permitted, yet the water lost from an evapotranspiration event has depleted one of the soil layers to 0 which, of course, would force adsorption into occurring. Obviously, there is a dilemma. To correct this problem, change the minimum water content to greater than 0. A value of 0.05 to 0.16 cm^3/cm^3 is most probable. If the minimum water content is already greater than 0 and less than the field capacity water content, then call technical support for advice.

ERROR: Insufficient water in column to meet ppt. event ET requirements.
Corrective action: The evapotranspiration is causing water to be removed below the minimum water content. Either lower the evapotranspiration amount for this event or lower the minimum water content. The latter is most preferred.

ERROR: Invalid entry.
Corrective action: An entry is not valid, so check the input parameters.

**ERROR: Invalid ET model.
Check rootflg variable.**

Corrective action: Call technical support and provide them with the evapotranspiration model which was selected.

**ERROR: Invalid piston flow/mixing entry.
Entry defaulted to 1.**

Corrective action: This error refers to the percent water bypass. A value between 0-100% should have been entered. The entry has defaulted to complete piston-type flow which is a percent water bypass of 0%.

**ERROR: Invalid printout flag.
Check prntflg variable.**

Corrective action: Call technical support for advice.

ERROR: Lack of convergence in Newton-Raphson method.

Corrective action: The error tolerance is probably set too low. Call technical support for advice.

**ERROR: Negative water uptake.
Inappropriate water uptake model selection.**

Corrective action: Select an alternate plant water uptake model (choice of linear or exponential). If this fails, then adjust the maximum root penetration depth or the root water uptake distribution. Check the evapotranspiration values to see that they are not significantly exceeding the irrigation/precipitation amounts. If there is a significant difference between the irrigation/precipitation amounts and evapotranspiration amounts, then follow the suggestion outlined as corrective action for the error: "Precipitation event has insufficient H2O to bring column to field capacity." If all else fails, call technical support for advice.

ERROR: Newton-Raphson approximation is less than 0.

Corrective action: The error tolerance is probably set too low or the bisection method is not functioning correctly. Call technical support for advice.

ERROR: Precipitation event has insufficient H2O to bring column to field capacity.

Corrective action: This error may occur when a low amount of precipitation or irrigation is associated with an evapotranspiration event which is high relative to the irrigation/precipitation amount. A way of dealing with this is to add successive precipitation/irrigation and evapotranspiration events:

For example:

Irrigation Event #	Irrigation Amount (cm)	ET Amount (cm)
50	3.50	2.75
51	0.01	0.50
52	4.00	2.00
53	2.75	1.0

CHANGE TO

Irrigation Event #	Irrigation Amount (cm)	ET Amount (cm)
50	3.50	2.75
51	4.01	2.50
52	2.75	1.0

NOTE: Technical support phone numbers are Dennis Corwin (714) 369-4819 and Bart Waggoner (714) 369-4875.

APPENDIX A - Notation

α_1 = linear plant root disrtibution coefficient ($-1 \leq \alpha_1 \leq 1$)

α_2 = exponential plant root distribution coefficient ($\alpha_2 > 0$)

γ = mobility coefficient, or more specifically, the fraction of V_{BI} which is subject to piston-flow (where, $0 \leq \gamma \leq 1$, $\gamma = 0$ represents total bypass, $\gamma = 1$ represents complete piston-type flow)

$1.0 - \gamma$ = fraction of V_{BI} which is bypassed

ρ_b = soil bulk density (kg/m^3)

ψ = soil-water suction head (m)

θ_{BI} = volumetric water content immediately before an irrigation (cm^3/cm^3)

θ_{fc} = volumetric water content at field capacity (cm^3/cm^3)

a $= \alpha_2 / L$

b = Langmuir adsorption maximum (mg/kg)

C_{ad} = adsorbed solute concentration (kg/m^3)

C_{AI} = concentration of solute in the soil water after an irrigation (kg/m^3)

C_{BI} = concentration of solute in the soil water immediately before an irrigation (kg/m^3)

C_{fc} = concentration of solute in the soil water at field apacity (kg/m^3)

C_{in} = solute concentration of the entering water (kg/m^3)

C_{out} = solute concentration of the exiting water (kg/m^3)

$g(z)$ = root distribution function

k = Langmuir affinity adsorption coefficient (L/mg)

L = plant root penetration depth (m)

$r(\psi)$ = reducing factor

$S(\psi,z)$ = volumetric root extraction function

T_a = integral of the volumetric extraction function from the soil surface ($z = 0$) to the depth of root penetration $(z=L)$

T_{ad} = total amount of adsorbed solute in V_t (kg)

T_{AI} = total amount of solute in a volume, V_t, of soil after an irrigation (kg)

T_{BI} = total amount of solute in a volume, V_t, of soil immediately before an irrigation (kg)

T_{in} = total amount of solute entering V_t (kg)

T_{out} = total amount of solute leaving V_t (kg)

T_p = potential volumetric transpiration (m^3)

T_{sw} = total amount of solute in the soil water of V_t (kg)

$U_l(z)$ = relative water uptake over the soil depth interval z_1 to z_2 for a linear root distribution

$U_e(z)$ = relative water uptake over the soil depth interval z_1 to z_2 for an exponential root distribution

V_{AI} = volume of soil water in V_t after an irrigation (m^3)

V_{BI} = volume of soil water in V_t immediately before an irrigation (m^3) = $\theta_{BI} V_t$

V_{fc} = volume of water in V_t at field capacity (m^3) = $\theta_{fc} V_t$

V_{in} = volume of water entering V_t (m^3)

V_{min} = minimum volume of soil water in V_t (m^3)

V_{out} = volume of water leaving V_t (m^3)

V_t = a unit volume of soil within the depth interval z_1 to z_2 (m^3)

x/m = adsorbed concentration of the Langmuir adsorption isotherm (mg/kg)

z = soil depth with zero at the soil surface and positive downward (m)

APPENDIX B - References

Addiscott, T. M. 1977. A simple computer model for leaching in . structured soils. J. of Soil Sci. 28:554-563.

Addiscott, T. M. and R. J. Wagenet. 1985. Concepts of solute leaching in soils: a review of modelling approaches. J. of Soil Sci. 36:411-424.

Bond, W. J. and D. E. Smiles. 1988. Predicting the average movement of reactive solutes in soils. Soil Use and Mgmt. 4(4):115-120.

Bressler, E. 1967. A model for tracing salt distribution in the soil profile and estimating the efficient combination of water quality and quantity under varying field conditions. Soil Sci. 104:227-233.

Burns, I. G. 1975. An equation to predict the leaching of surface-applied nitrate. J. Agric. Sci. 85:443-454.

Burns, I. G. 1976. Equations to predict the leaching of nitrate uniformly incorporated to a known depth or uniformly distributed throuhgout a soil profile. J. Agric. Sci. 86:305-313.

Coats, K. H. and B. D. Smith. 1964. Dead end pore volume and dispersion in porous media. Soc. Petr. Eng. J. 4:74-84.

Corwin, D. L., B. L. Waggoner and J. D. Rhoades. 1990a. A functional transport model which accounts for bypass. J. Environ. Qual. (submitted).

Corwin, D. L., B. L. Waggoner and J. D. Rhoades. 1990b. Simulating the movement of a reactive solute through a soil lysimeter column using a functional model of solute transport. J. Environ. Qual. (submitted).

Deans, H. H. 1963. A mathematical model for dispersion in the direction of flow in porous media. Trans. A.I.M.E. 228

Engesgaard, P. and Th. H. Christensen. 1988. A review of chemical solute transport models. Nordic Hydrol. 19:183-216.

Feddes, R. A., P. J. Kowalik and H. Zaradny. 1978. Simulation of field water use and crop yield. Pudoc, Wageningen, Simulation Monogr., 189 pp.

Feddes, R. A., P. Kabat, P. J. T. van Bakel, J. J. B. Bronswijk and J. Halbertsma. 1988. Modelling soil water dynamics in the unsaturated zone - state of the art. J. of Hydrol. 100:69-111.

Hoogland, J. C., R. A. Feddes and C. Belmans. 1981. Root water uptake model depending on soil water pressure head and maximum extraction rate. Acta Hort. 119:123-136.

Keren, R., R. G. Gast and B. Bar-Yosef. 1981. pH-Dependent boron adsorption by Na-montmorillonite. Soil Sci. Soc. Amer. Proc. 45:45-48.

Keren, R. and U. Mezuman. 1981. Boron adsorption by clay minerals using a phenomenological equation. Clays and Clay Minerals. 29:198-204.

Lyman, W. J. 1982. Adsorption coefficient for soils and sediments. (W. J. Lyman, W. F. Reehl and D. H. Rosenblatt, editors) *In:* Handbook of Chemical Property Estimation Methods. McGraw-Hill Book Co. New York. pp. 4-1 to 4-32.

Mezuman, U. and R. Keren. 1981. Boron adsorption by soils using a phenomenological adsorption equation. Soil Sci. Soc. Amer. Proc. 45:722-726.

Molz, F. J. and I. Remson. 1970. Extraction term models of soil moisture use by transpiring plants. Water Resour. Res. 6(5):1346-1356.

Nielsen, D. R., M. Th. van Genuchten, and J. W. Biggar. 1986. Water flow and solute transport processes in the unsaturated zone. Water Resour. Res. 22(9):89S-108S.

Perrochet, P. 1987. Water uptake by plant roots - A simulation model, I. Conceptual model. J. Hydrol. 95:55-61.

Prasad, R. 1988. A linear root water uptake model. J. of Hydrol. 99:297-306.

Ritchie, J. T. 1984. A user-oriented model of the soil water balance in wheat. USDA-ARS, Grassland, Soil and Water Res. Lab., Temple, TX. 21 pp.

Ritchie, J. T. and S. Otter. 1984. Description and performance of CERES-wheat: A user-oriented wheat yield model. USDA-ARS, Grassland, Soil and Water Res. Lab., Temple, TX, 37 pp.

Rose, C. W. , W. L. Hogarth and P. W. A. Dayananda. 1982. Movement of peak solute concentration position by leaching in a non-sorbing soil. Aust. J. Soil Res. 20:23-36.

Tanji, K. K., L. D. Doneen, G. V. Ferry and R. S. Ayers. 1972. Computer simulation analysis on reclamation of salt-affected soils in the San Joaquin Valley. Soil Sci. Soc. Amer. Proc. 36:127-133.

Turner, G. A. 1958. The flow-structure in packed beds. Chem. Eng. Sci. 7:156-165.

van Genuchten, M. Th. and W. A. Jury. 1987. Progress in saturated flow and transport modeling. Rev. of Geophys. 25(2):135-140.

Waggoner, B. L., D. L. Corwin, J. D. Rhoades and U. Shani. 1990. Simulation of boron transport in lysimeters under various irrigation management schemes. Soil Sci. (submitted).

APPENDIX C - MPW Fortran Source Code

Only the transport model source code is provided. The I/O interface code is excluded since it is not regarded as essential in helping the user understand the model.

```
!!M Inlines.f

      subroutine TET1SUB(err)

      common irrdate, irramt, et, ciw, ph, sph, zz, fc, h2omin, k, q,
     1       rho, sh2o, csw, cad, ctotal, khb, kb, koh, qph, kf, nf,
     2       pdate, mdate, hdate, alpha, rpmax, gamma, watup,
     3       total, h2oin, h2oout, cin, cout, starth2o, startsol,
     4       excess, watcon, solcon, rp, uptake, cbi, aa, beta, a,
     5       temp, z1, z2, reluptake, h, oh, b, hb, kchb, kcb,
     6       kcoh, kcq, root1, root2, root3, mg1, mg2, mg3, phi,
     7       collen, initsol, inith2o, totet, totdrain, toth2oin,
     8       toth2oout, lf, storeh2o, totsolin, totsolout, totsolst,
     9       totad, totsoln, avefc, avesat, fcpv, satpv, errh2o,
     1       errsol, jjj, jjjj, out, printin, nirr,
     2       site, treat, ncomp, ncrop, irr, crop, i, j, unitno,
     3       prnt, diskno, loop, chemmdl, phflg, soil,
     4       kacflg, rootflg, border, fileinput, filename, ht,
     5       printout, jread, mbflg, htitle

      implicit real*8 (a-h)
      implicit real*8 (p-z)
      implicit integer*4 (i-o)

      real*8 irrdate(100), irramt(100), et(100), ciw(100), phi(50),
     1       ph(25,100), sph(25), zz(25), fc(25), h2omin(25), k(25),
     2       q(25), rho(25), sh2o(25), csw(25), cad(25), ctotal(25),
     3       khb(25), kb(25), koh(25), qph(25), kf(25), nf(25),
     4       pdate(25), mdate(25), hdate(25), alpha(25), rpmax(25),
     5       gamma(25), watup(25), total(25), h2oin(25),
     6       h2oout(25), cin(25), cout(25), starth2o(25), startsol(25),
     7       excess, watcon, solcon, rp, uptake, cbi, aa, beta, a, temp,
     8       z1, z2, reluptake, h, oh, b, hb, kchb, kcb, kcoh,
     9       kcq, root1, root2, root3, mg1, mg2, mg3, collen, initsol,
     1       inith2o, totet, totdrain, toth2oin, toth2oout, lf,
     2       storeh2o, totsolin, totsolout, totsolst, totad, totsoln,
     3       avefc, avesat, fcpv, satpv, errh2o, errsol

      integer*4  jjj, jjjj, out, printin, nirr,  site,
     1       treat, ncomp, ncrop, irr, crop, i, j, unitno, prnt,
     2       diskno, loop, chemmdl, phflg, soil, kacflg,
     3       rootflg(25), printout, jread, mbflg, errnum

      character htitle*77, border*56, fileinput*20, filename*20, ht

C *** Following 2 lines added for Mac version:
      integer*4 err
      cexternal AnimCursor

C     ***** Variable Definitions ***************************************
C           irrdate = irrigation date        irramt = irrigation amt.
C           et = evapotranspiration          ciw = conc. of irr. h2o
C           ph = pH of each compartment      sph = starting pH
C           zz = depth increment             fc = field capacity
C           h2omin = min. h2o content        k = Langmuir K
C           q = Langmuir max. ads.           rho = bulk density
C           sh2o = h2o con. before each ppt. csw = conc. in soil h2o
C           cad = adsorbed conc.             ctotal = cmpt. total conc.
C           qph = Keren pH model max. ads.   khb = B(OH)3 affinity coef.
C           kb = B(OH)4 affinity coef.       koh = (OH-) affinity coef.
C           pdate = planting date            mdate = maturing date
C           hdate = harvest date             alpha = root descriptor
C           rpmax = max. root penetration    gamma = piston/mixing term
```

```
C           watup = % plant H2O uptake
C           total = total solute/cm^3        h2oin = H2O entering cmpt.
C           h2oout = H2O leaving             cin = solute conc. entering
C           cout = solute conc leaving       excess = excess cmpt. H2O
C           watcon = sh2o / zz               solcon = csw
C           rp = root penetration            uptake = ET in a compartment
C           cbi = conc before irrig.         z1 & z2 = depth indicators
C           reluptake = rel. uptake          h = [H+]
C           oh = [OH-]                       b = [B(OH)4]
C           hb = [B(OH)3]                    root1 = alpha / (rpmax)^2
C           root2 = alpha / rpmax             root3 = 1 / rpmax
C           mg1,mg2,mg3 = convert units      collen = soil column length
C           initsol = init. total solute conc. in soil column
C           inith2o = initial total H2O in soil column
C           totet = cumulative ET            totdrain = total drainage
C           toth2oin = total H2O applied     toth2oout = totet+totdrain
C           lf = runnign leaching fraction   storeh2o = H2O stored in col.
C           totsolin = total solute applied  totsolout = tot. sol. drain.
C           totsolst = total solute stored   totad = total solute adsorb.
C           totsoln = total solute in soln.  avefc = ave. col. field cap.
C           avesat = ave. col. saturation    fcpv = pore vol. (field cap)
C           satpv = pore vol. (saturation)   errh2o = mass bal. H2O error
C           errsol = mass bal. solute error  starth2o = start. h2o content
C           startsol = starting sol. conc.   mbflg = mass balance flag
C           cmpt = cmpts. of output data     jjj = counter
C           jjjj = counter                   out = output device
C           printin = screen printout flag   nirr = no. of irrigations
C           endirr = ending irrigation       ncrop = no. of crops
C           ncomp = no. of compartments      irr = irr. event counter
C           crop = crop counter              i & j = counters
C           unitno = terminal no.            prnt = printer device no.
C           diskno = disk unit no.           endmess = error mess. flag
C           loop = main loop counter         chemmdl = chem. model flag
C           phflg = ph flag                  soil = soil type flag
C           kacflg = Keren affin. coef. flag rootflg = root model flag
C           fileinput = input file           filename = output file name
C           ht = horizontal tab              printout = printout flag
C           phi = Newton-Raphson estimate    jread = flag to read input
C           prntflg = printout flag          htitle = title heading
C     *****************************************************************


C ********* Put up the cursor
            call AnimCursor()

C     ***** Variable Assignment and Initialization
      j = 0
      out = 3
      nirr = 0
      site = 0
      treat = 0
      ncomp = 0
      ncrop = 0
      irr = 0
      crop = 0
      chemmdl = 0
      phflg = 0
      soil = 1
      kacflg = 1
      printin = 0
      printout = 1
      unitno = 6
      prnt = 9
      diskno = 7
      jread = 2
      errnum = 0
      mbflg = 0
      mat irrdate = 0.0
      mat irramt = 0.0
      mat et = 0.0
      mat ciw = 0.0
      mat ph = 7.0
      mat sph = 7.0
      mat phi = 0.0
      mat zz = 0.0
      mat fc = 0.0
      mat h2omin = 0.0
      mat k = 0.0
      mat q = 0.0
      mat rho = 0.0
      mat sh2o = 0.0
      mat csw = 0.0
      mat cad = 0.0
      mat ctotal = 0.0
      mat khb = 0.0
      mat kb = 0.0
      mat koh = 0.0
      mat qph = 0.0
      mat kf = 0.0
      mat nf = 0.0
      mat pdate = 0.0
      mat mdate = 0.0
      mat hdate = 0.0
      mat alpha = 0.0
      mat rpmax = 0.0
      mat rootflg = 2
      mat gamma = 1.0
      mat watup = 0.0
      mat total = 0.0
      mat h2oin = 0.0
      mat h2oout = 0.0
      mat cin = 0.0
      mat cout = 0.0
      mat starth2o = 0.0
      mat startsol = 0.0
      excess = 0.0
      watcon = 0.0
      solcon = 0.0
      rp = 0.0
      uptake = 0.0
      cbi = 0.0
      z1 = 0.0
      z2 = 0.0
      reluptake = 0.0
      h = 0.0
      oh = 0.0
      b = 0.0
      hb = 0.0
      kchb = 0.0
      kcb = 0.0
      kcoh = 0.0
      kcq = 0.0
      collen = 0.0
      initsol = 0.0
      inith2o = 0.0
      totet = 0.0
      totdrain = 0.0
      toth2oin = 0.0
      toth2oout = 0.0
      lf = 0.0
      storeh2o = 0.0
      totsolin = 0.0
      totsolout = 0.0
      totsolst = 0.0
      totad = 0.0
      totsoln = 0.0
      avefc = 0.0
      avesat = 0.0
      fcpv = 0.0
      satpv = 0.0
      errh2o = 0.0
      errsol = 0.0
      mbflg = 1
      crop = 1
      err = 0

C     Unit Conversion Factors for Keren Model (ml/mg to l/moles)
      mg1 = 1.0 / 61.835
      mg2 = 1.0 / 78.843
      mg3 = 1.0 / 17.008

C *** Call input subroutine - reads previously created input file into memory
      call inroutine(err)

C     Determine the starting adsorbed conc. (cad) in mg/kg from
C     the starting soln. conc. (csw) in mg/l
      do 770 i=1,ncomp

C          ## No adsorption
           if (chemmdl.eq.1) then
                cad(i) = 0.0
C          ## Langmuir chemical adsorption model
           else if (chemmdl.eq.2) then
                if (csw(i).eq.0.0) then
                     cad(i) = 0.0
                else
                     cad(i) = (k(i)*q(i)*csw(i)) / (1.0+k(i)*csw(i))
                end if
C          ## Keren boron adsorption model
           else if (chemmdl.eq.3) then
                if (csw(i).eq.0.0) then
                     cad(i) = 0.0
                else
                     alpha1 = (10.0**9.15) / (10.0**sph(i))
                     beta1 = (koh(i) * (10.0**sph(i))) /
     1                       (10.0**14.0)
                     b = (csw(i)/10811.0) / (1.0+alpha1)
                     hb = b * alpha1
                     cad(i) = (qph(i)*((khb(i)*(1.0/mg1)*hb)+
     1                        (kb(i)*(1.0/mg2)*b))) / (1.0+
     2                        khb(i)*(1.0/mg1)*hb+kb(i)*
     3                        (1.0/mg2)*b+(1.0/mg3)*beta1)
                end if
           else if (chemmdl.eq.4) then
                if (csw(i).eq.0.0) then
                     cad(i) = 0.0
                else
                     cad(i) = kf(i) * (csw(i)**nf(i))
                end if
           end if
  770 continue

C     Convert inputted water contents to volumes, calculate total solute
C     in column at start, calculate total soil column length, and determine
C     initial H2O and solute in soil column
      do 780, i=1,ncomp
           starth2o(i) = sh2o(i)
           startsol(i) = csw(i)
           sh2o(i) = sh2o(i) * zz(i)
             fc(i) = fc(i) * zz(i)
             h2omin(i) = h2omin(i) * zz(i)
             ctotal(i) = sh2o(i) * csw(i) + rho(i) * zz(i) * cad(i)
           collen = collen + zz(i)
           inith2o = inith2o + sh2o(i)
           initsol = initsol + ctotal(i)
  780 continue
      initsol = initsol / collen

C *** Create and open the output file:
      open (99, file='TETRANS1.OUT', status='new')

C     TOP OF MAIN LOOP
      do 1000 irr = 1, nirr

C ********* Rotate the cursor
            call AnimCursor()

C ********* Check for command-. or command-q presses
            call CheckAbort(err)
            if ( err .ne. 0 ) leave

C           Advance Crop Counter, If Appropriate
            if (crop.lt.ncrop) then
                  if (irrdate(irr).ge.pdate(crop+1)) crop = crop + 1
            end if

C           Call Root Growth Subroutine
            call rootgrow(err)
            if ( err .ne. 0 ) leave

C           Call Irrigation Subroutine
            call irrigation(err)
            if ( err .ne. 0 ) leave

C ********* Save after irrigation results to file:
            call savedat1()

C ********* Rotate the cursor again
            call AnimCursor()

C           Call Root Water Uptake Subroutine
            call evapotran(err)
            if ( err .ne. 0 ) leave

C ********* Save after et results to file:
            call savedat2()

 1000 continue

C *** Restore the arrow cursor
      call InitCursor
```

```
C *** Close output file:
      close(99)

C *** Was there an error?
      if ( err .ne. 0 ) call errmess(err,irr)
      return
      end


C     ***** Root Growth Subroutine

      subroutine rootgrow(err)

      common irrdate, irramt, et, ciw, ph, sph, zz, fc, h2omin, k, q,
     1       rho, sh2o, csw, cad, ctotal, khb, kb, koh, qph, kf, nf,
     2       pdate, mdate, hdate, alpha, rpmax, gamma, watup,
     3       total, h2oin, h2oout, cin, cout, starth2o, startsol,
     4       excess, watcon, solcon, rp, uptake, cbi, aa, beta, a,
     5       temp, z1, z2, reluptake, h, oh, b, hb, kchb, kcb,
     6       kcoh, kcq, root1, root2, root3, mg1, mg2, mg3, phi,
     7       collen, initsol, inith2o, totet, totdrain, toth2oin,
     8       toth2oout, lf, storeh2o, totsolin, totsolout, totsolst,
     9       totad, totsoln, avefc, avesat, fcpv, satpv, errh2o,
     1       errsol, jjj, jjjj, out, printin, nirr,
     2       site, treat, ncomp, ncrop, irr, crop, i, j, unitno,
     3       prnt, diskno, loop, chemmdl, phflg, soil,
     4       kacflg, rootflg, border, fileinput, filename, ht,
     5       printout, jread, mbflg, htitle

      implicit real*8 (a-h)
      implicit real*8 (p-z)
      implicit integer*4 (i-o)

      real*8 irrdate(100), irramt(100), et(100), ciw(100), phi(50),
     1       ph(25,100), sph(25), zz(25), fc(25), h2omin(25), k(25),
     2       q(25), rho(25), sh2o(25), csw(25), cad(25), ctotal(25),
     3       khb(25), kb(25), koh(25), qph(25), kf(25), nf(25),
     4       pdate(25), mdate(25), hdate(25), alpha(25), rpmax(25),
     5       gamma(25), watup(25), total(25), h2oin(25),
     6       h2oout(25), cin(25), cout(25), starth2o(25), startsol(25),
     7       excess, watcon, solcon, rp, uptake, cbi, aa, beta, a, temp,
     8       z1, z2, reluptake, h, oh, b, hb, kchb, kcb, kcoh,
     9       kcq, root1, root2, root3, mg1, mg2, mg3, collen, initsol,
     1       inith2o, totet, totdrain, toth2oin, toth2oout, lf,
     2       storeh2o, totsolin, totsolout, totsolst, totad, totsoln,
     3       avefc, avesat, fcpv, satpv, errh2o, errsol

      integer*4 err

      integer*4  jjj, jjjj, out, printin, nirr,  site,
     1       treat, ncomp, ncrop, irr, crop, i, j, unitno, prnt,
     2       diskno, loop, chemmdl, phflg, soil, kacflg,
     3       rootflg(25), printout, jread, mbflg

      character htitle*77, border*56, fileinput*20, filename*20, ht

C     ## Root penetration determination
      if (irrdate(irr).ge.hdate(crop)) then
            rp = 0.0
      else if (irrdate(irr).ge.mdate(crop)) then
            rp = rpmax(crop)
      else
            rp = rpmax(crop)*(irrdate(irr)-pdate(crop))/(mdate(crop)
     1                         - pdate(crop))
      end if

      return
      end


C     ***** Irrigation Subroutine

      subroutine irrigation(err)

      common irrdate, irramt, et, ciw, ph, sph, zz, fc, h2omin, k, q,
     1       rho, sh2o, csw, cad, ctotal, khb, kb, koh, qph, kf, nf,
     2       pdate, mdate, hdate, alpha, rpmax, gamma, watup,
     3       total, h2oin, h2oout, cin, cout, starth2o, startsol,
     4       excess, watcon, solcon, rp, uptake, cbi, aa, beta, a,
     5       temp, z1, z2, reluptake, h, oh, b, hb, kchb, kcb,
     6       kcoh, kcq, root1, root2, root3, mg1, mg2, mg3, phi,
     7       collen, initsol, inith2o, totet, totdrain, toth2oin,
     8       toth2oout, lf, storeh2o, totsolin, totsolout, totsolst,
     9       totad, totsoln, avefc, avesat, fcpv, satpv, errh2o,
     1       errsol, jjj, jjjj, out, printin, nirr,
     2       site, treat, ncomp, ncrop, irr, crop, i, j, unitno,
     3       prnt, diskno, loop, chemmdl, phflg, soil,
     4       kacflg, rootflg, border, fileinput, filename, ht,
     5       printout, jread, mbflg, htitle

      implicit real*8 (a-h)
      implicit real*8 (p-z)
      implicit integer*4 (i-o)

      real*8 irrdate(100), irramt(100), et(100), ciw(100), phi(50),
     1       ph(25,100), sph(25), zz(25), fc(25), h2omin(25), k(25),
     2       q(25), rho(25), sh2o(25), csw(25), cad(25), ctotal(25),
     3       khb(25), kb(25), koh(25), qph(25), kf(25), nf(25),
     4       pdate(25), mdate(25), hdate(25), alpha(25), rpmax(25),
     5       gamma(25), watup(25), total(25), h2oin(25),
     6       h2oout(25), cin(25), cout(25), starth2o(25), startsol(25),
     7       excess, watcon, solcon, rp, uptake, cbi, aa, beta, a, temp,
     8       z1, z2, reluptake, h, oh, b, hb, kchb, kcb, kcoh,
     9       kcq, root1, root2, root3, mg1, mg2, mg3, collen, initsol,
     1       inith2o, totet, totdrain, toth2oin, toth2oout, lf,
     2       storeh2o, totsolin, totsolout, totsolst, totad, totsoln,
     3       avefc, avesat, fcpv, satpv, errh2o, errsol

      integer*4 err

      integer*4  jjj, jjjj, out, printin, nirr,  site,
     1       treat, ncomp, ncrop, irr, crop, i, j, unitno, prnt,
     2       diskno, loop, chemmdl, phflg, soil, kacflg,
     3       rootflg(25), printout, jread, mbflg

      character htitle*77, border*56, fileinput*20, filename*20, ht

C     ## Add to total applied ppt (cm):
      toth2oin = toth2oin + irramt(irr)

C     ## Add to total applied solute (ug/cm^2):
      totsolin = totsolin + ciw(irr) * irramt(irr)

C     ## Initial conditions
      h2oin(1) = irramt(irr)
      cin(1) = ciw(irr)
```

```fortran
      storeh2o = 0.0
      totsoln = 0.0
      totad = 0.0
      totsolst = 0.0

C     ## Loop through each increment and mix:
      do 2000 i=1,ncomp

C        ## Error check:
         if (h2oin(i) .lt. 0.0) then
            err = 1
            return
         end if

C        ## Calc total solute in increment before the irrigation:
         cbi = sh2o(i)*csw(i)+zz(i)*rho(i)*cad(i)

C        ## After irrig mixing. If complete bypass (gamma = 0), do following:
         if (gamma(i) .eq. 0.0) then

C           ## Insufficient ppt to reach field capacity:
            if (h2oin(i) .le. (fc(i) - sh2o(i))) then
               h2oout(i) = 0.0
               cout(i) = 0.0
               sh2o(i) = h2oin(i) + sh2o(i)

C           ## Sufficient water:
            else
               h2oout(i) = h2oin(i) - fc(i) + sh2o(i)
               cout(i) = ((gamma(i)*csw(i)*sh2o(i))-(cin(i)*fc(i))+
     1                   (cin(i)*h2oin(i))+((1.0-gamma(i))*cin(i)*
     2                   sh2o(i)))/h2oout(i)
               sh2o(i) = fc(i)
            end if

C         ## Otherwise, for gamma between 0 and 1 do the following:
          else if (gamma(i) .gt. 0.0 .and. gamma(i) .le. 1.0) then

C           ## Insufficient ppt to reach field capacity:
            if (h2oin(i) .le. (fc(i) - sh2o(i))) then
               h2oout(i) = 0.0
               cout(i) = 0.0
               sh2o(i) = h2oin(i)+sh2o(i)
            else if ((h2oin(i) .gt. (fc(i)-sh2o(i))) .and.
     1              (h2oin(i) .le. (fc(i)-((1.0-gamma(i))*sh2o(i))))) then
               h2oout(i) = h2oin(i) - fc(i) + sh2o(i)
               cout(i) = csw(i)
               sh2o(i) = fc(i)
            else
               h2oout(i) = h2oin(i) - fc(i) + sh2o(i)
               cout(i) = ((gamma(i)*csw(i)*sh2o(i))-(cin(i)*fc(i))+
     1                   (cin(i)*h2oin(i))+((1.0-gamma(i))*cin(i)*
     2                   sh2o(i)))/h2oout(i)
               sh2o(i) = fc(i)
            end if
         end if

C        ## Total solute in increment after irrigation (=stored+inputs-outputs)
         cai = cbi + h2oin(i)*cin(i) - h2oout(i)*cout(i)

C        # Round-off error correction:
         if ((cin(i) .eq. 0.0) .and. (cai .lt. 0.0) .and. (chemmdl .eq. 1)) then
            cai = 0.0
         end if
         ctotal(i) = cai

C        ## Solute & H2O drained:
         if  (i.eq.ncomp) then

C           # Total H2O drainage (cm) from bottom of soil column:
            totdrain = totdrain + h2oout(ncomp)

C           # Total solute drained (ug/cm^2) from bottom of soil column:
            totsolout = totsolout + cout(ncomp) * h2oout(ncomp)
         else

C           # H2O drained from one compartment and entering next:
            h2oin(i+1) = h2oout(i)

C           # Solute drained from one compartment and entering next:
            cin(i+1) = cout(i)
         end if

C        ## Stored H2O in soil column:
         storeh2o = storeh2o + sh2o(i)

C        ## Partitioning of solute (cai) into ads & sol'n phases after irrig:
         call adsorption(err)
         if (err .ne. 0) return

C        ## Solute in soil column:
C        # Total solute in solution (ug/cm^3)
         totsoln = totsoln + csw(i)*sh2o(i)
C        # Total solute adsorbed (ug/cm^3)
         totad = totad + cad(i) * rho(i) * zz(i)
C        # Total stored solute (ug/cm^3)
         totsolst = totsolst + ctotal(i)

 2000 continue
      return
      end


C     ***** Root Water Uptake Subroutine -- Evapotranspiration

      subroutine evapotran(err)

      common irrdate, irramt, et, ciw, ph, sph, zz, fc, h2omin, k, q,
     1       rho, sh2o, csw, cad, ctotal, khb, kb, koh, qph, kf, nf,
     2       pdate, mdate, hdate, alpha, rpmax, gamma, watup,
     3       total, h2oin, h2oout, cin, cout, starth2o, startsol,
     4       excess, watcon, solcon, rp, uptake, cbi, aa, beta, a,
     5       temp, z1, z2, reluptake, h, oh, b, hb, kchb, kcb,
     6       kcoh, kcq, root1, root2, root3, mg1, mg2, mg3, phi,
     7       collen, initsol, inith2o, totet, totdrain, toth2oin,
     8       toth2oout, lf, storeh2o, totsolin, totsolout, totsolst,
     9       totad, totsoln, avefc, avesat, fcpv, satpv, errh2o,
     1       errsol, jjj, jjjj, out, printin, nirr,
     2       site, treat, ncomp, ncrop, irr, crop, i, j, unitno,
     3       prnt, diskno, loop, chemmdl, phflg, soil,
     4       kacflg, rootflg, border, fileinput, filename, ht,
     5       printout, jread, mbflg, htitle

      implicit real*8 (a-h)
      implicit real*8 (p-z)
      implicit integer*4 (i-o)
      real*8 irrdate(100), irramt(100), et(100), ciw(100), phi(50),
     1       ph(25,100), sph(25), zz(25), fc(25), h2omin(25), k(25),
     2       q(25), rho(25), sh2o(25), csw(25), cad(25), ctotal(25),
     3       khb(25), kb(25), koh(25), qph(25), kf(25), nf(25),
     4       pdate(25), mdate(25), hdate(25), alpha(25), rpmax(25),
     5       gamma(25), watup(25), total(25), h2oin(25),
     6       h2oout(25), cin(25), cout(25), starth2o(25), startsol(25),
     7       excess, watcon, solcon, rp, uptake, cbi, aa, beta, a, temp,
     8       z1, z2, reluptake, h, oh, b, hb, kchb, kcb, kcoh,
     9       kcq, root1, root2, root3, mg1, mg2, mg3, collen, initsol,
     1       inith2o, totet, totdrain, toth2oin, toth2oout, lf,
     2       storeh2o, totsolin, totsolout, totsolst, totad, totsoln,
     3       avefc, avesat, fcpv, satpv, errh2o, errsol

        integer*4 err

      integer*4  jjj, jjjj, out, printin, nirr,  site,
     1       treat, ncomp, ncrop, irr, crop, i, j, unitno, prnt,
     2       diskno, loop, chemmdl, phflg, soil, kacflg,
     3       rootflg(25), printout, jread, mbflg

      character htitle*77, border*56, fileinput*20, filename*20, ht

C     Initialize:
      mat watup = 0.0

C     Compute total ET (for mass balance):
      totet = totet + et(irr)

C     Calculation of exponential plant water uptake parameters:
      if ( rootflg(crop) .eq. 2 ) then
         if ( rp .eq. 0.0 ) then
            excess = et(irr)
         else
            excess = 0.0
            a = alpha(crop)/rp
            temp = 1.0 - dexp(-a*rp)
         end if
      end if

C     Calculation of Perrochet/Ritchie linear plant water uptake model coefs:
      if ( rootflg(crop) .eq. 3 ) then
         if ( rp .eq. 0.0 ) then
            excess = et(irr)
            root1 = 0.0
            root2 = 0.0
            root3 = 0.0
         else
            excess = 0.0
            root1 = alpha(crop)/(rp**2)
            root2 = alpha(crop)/rp
            root3 = 1.0/rp
         end if
      end if

C     Compute water uptake from each increment, where z1 is the top and
C     z2 is the bottom depth of the increment:
      z1 = 0.0
      do 3000 i=1,ncomp
         z2 = z1 + zz(i)

C        Exponential plant water uptake model
```

```
         if ( rootflg(crop) .eq. 2 ) then
            if ( rp .le. z1 ) then
               uptake = 0.0
            else if ( rp .gt. z1 .and. rp .lt. z2 ) then
               reluptake = (dexp(-a*z1) - dexp(-a*rp))/temp
               uptake = reluptake*et(irr)
            else
               reluptake = (dexp(-a*z1) - dexp(-a*z2))/temp
               uptake = reluptake*et(irr)
            end if
         end if

C        Perrochet/Ritchie plant water uptake model
         if ( rootflg(crop) .eq.3 ) then
            if ( rp .le. z1) then
               uptake = 0.0
            else if ( rp .gt. z1 .and. rp .lt. z2 ) then
               reluptake = root1*((rp**2) - (z1**2))
     &                     - (root2 - root3)*(rp - z1)
               uptake = reluptake*et(irr)
            else
               reluptake = root1*((z2**2) - (z1**2))
     &                     - (root2 - root3)*(z2 - z1)
               uptake = reluptake*et(irr)
            end if
         end if

C        Remove ET from soil water profile
         sh2o(i) = sh2o(i) - uptake - excess
         watup(i) = uptake + excess
         if ( sh2o(i) .ge. h2omin(i) ) then
            excess = 0.0
         else
            if ( i .eq. ncomp ) then
               err = 2
               return
            else
               watup(i) = fc(i) - h2omin(i)
               excess = h2omin(i) - sh2o(i)
               sh2o(i) = h2omin(i)
            end if
         end if
         z1 = z2

C        Partitioning of solute into adsorbed & solution phases after ET
         call adsorption(err)
         if ( err .ne. 0 ) return
 3000 continue
      return
      end


C     ***** Adsorption Subroutine

      subroutine adsorption(err)

      common irrdate, irramt, et, ciw, ph, sph, zz, fc, h2omin, k, q,
     1       rho, sh2o, csw, cad, ctotal, khb, kb, koh, qph, kf, nf,
     2       pdate, mdate, hdate, alpha, rpmax, gamma, watup,
     3       total, h2oin, h2oout, cin, cout, starth2o, startsol,
     4       excess, watcon, solcon, rp, uptake, cbi, aa, beta, a,
     5       temp, z1, z2, reluptake, h, oh, b, hb, kchb, kcb,
     6       kcoh, kcq, root1, root2, root3, mg1, mg2, mg3, phi,
     7       collen, initsol, inith2o, totet, totdrain, toth2oin,
     8       toth2oout, lf, storeh2o, totsolin, totsolout, totsolst,
     9       totad, totsoln, avefc, avesat, fcpv, satpv, errh2o,
     1       errsol, jjj, jjjj, out, printin, nirr,
     2       site, treat, ncomp, ncrop, irr, crop, i, j, unitno,
     3       prnt, diskno, loop, chemmdl, phflg, soil,
     4       kacflg, rootflg, border, fileinput, filename, ht,
     5       printout, jread, mbflg, htitle

      implicit real*8 (a-h)
      implicit real*8 (p-z)
      implicit integer*4 (i-o)

      real*8 irrdate(100), irramt(100), et(100), ciw(100), phi(50),
     1       ph(25,100), sph(25), zz(25), fc(25), h2omin(25), k(25),
     2       q(25), rho(25), sh2o(25), csw(25), cad(25), ctotal(25),
     3       khb(25), kb(25), koh(25), qph(25), kf(25), nf(25),
     4       pdate(25), mdate(25), hdate(25), alpha(25), rpmax(25),
     5       gamma(25), watup(25), total(25), h2oin(25),
     6       h2oout(25), cin(25), cout(25), starth2o(25), startsol(25),
     7       excess, watcon, solcon, rp, uptake, cbi, aa, beta, a, temp,
     8       z1, z2, reluptake, h, oh, b, hb, kchb, kcb, kcoh,
     9       kcq, root1, root2, root3, mg1, mg2, mg3, collen, initsol,
     1       inith2o, totet, totdrain, toth2oin, toth2oout, lf,
     2       storeh2o, totsolin, totsolout, totsolst, totad, totsoln,
     3       avefc, avesat, fcpv, satpv, errh2o, errsol

      integer*4 err

      integer*4  jjj, jjjj, out, printin, nirr,  site,
     1       treat, ncomp, ncrop, irr, crop, i, j, unitno, prnt,
     2       diskno, loop, chemmdl, phflg, soil, kacflg,
     3       rootflg(25), printout, jread, mbflg

      character htitle*77, border*56, fileinput*20, filename*20, ht

C     ## No adsorption
      if (chemmdl.eq.1) then
           if (sh2o(i).eq.0.0) then
                    err = 4
                        return
           else
                csw(i) = ctotal(i) / sh2o(i)
                cad(i) = 0.0
           end if
C     ## Langmuir adsorption model routine
      else if (chemmdl.eq.2) then
           if ((k(i).eq.0.0).or.(q(i).eq.0.0)) then
                if (sh2o(i).eq.0.0) then
                             err = 4
                               return
                else
                   csw(i) = ctotal(i) / sh2o(i)
                 cad(i) = 0.0
                end if
         else
                if (sh2o(i).eq.0.0) then
                     csw(i) = 0.0
                     cad(i) = ctotal(i) / (rho(i)*zz(i))
                else if (ctotal(i).eq.0.0) then
                     csw(i) = 0.0
                     cad(i) = 0.0
                else
```

```
                   aa = sh2o(i)+zz(i)*rho(i)*k(i)*q(i)-ctotal(i)*k(i)
                     csw(i) = (-1.0*aa+dsqrt(aa**2.0+(4.0*sh2o(i)*k(i)*
     1                         ctotal(i)))) / (2.0*sh2o(i)*k(i))
                     cad(i) = (ctotal(i)-(sh2o(i)*csw(i))) / (rho(i)*
     1                         zz(i))
                end if
         end if
C      ## Keren boron adsorption model routine
      else if (chemmdl.eq.3) then
         if ((qph(i).eq.0.0).or.((khb(i).eq.0.0).and.(kb(i).
     1          eq.0.0))) then
                if (sh2o(i).eq.0.0) then
                             err = 4
                             return
                else
                     csw(i) = ctotal(i) / sh2o(i)
                     cad(i) = 0.0
                end if
           else
                if (sh2o(i).eq.0.0) then
                     csw(i) = 0.0
                     cad(i) = ctotal(i) / (rho(i)*zz(i))
                else if (ctotal(i).eq.0.0) then
                     csw(i) = 0.0
                     cad(i) = 0.0
                else
                     alpha1 = (10.0**9.15) / (10.0**ph(i,irr))
                     beta1 = (koh(i)*(10.0**ph(i,irr)))/(10.0**14.0)
                     a1 = 10811.0 *sh2o(i)*(alpha1+1.0)*(((1.0/
     1                    mg1)*khb(i)*alpha1)+((1.0/mg2)*kb(i)))
                     b1 = (10811.0*sh2o(i)*(alpha1+1.0)*((1.0/mg3)
     1                     *beta1+1.0)) + (rho(i) *zz(i)*qph(i)-
     2                    ctotal(i))*(((1.0/mg1)*khb(i)*alpha1)+
     3                    ((1.0/mg2)*kb(i)))
                     c1 = -ctotal(i)*((1.0/mg3)*beta1+1.0)
                     d1 = ((dabs(b1))**2.0) - (4.0*a1*c1)
                     b = (-b1 + dsqrt(d1)) / (2.0*a1)
                     hb = b * alpha1
                     csw(i) = 10811.0 * (b + hb)
                     cad(i) = (ctotal(i)-(csw(i)*sh2o(i))) / (rho(i)*
     1                         zz(i))
                end if
           end if
C      ## Freundlich/linear adsorption model
      else if (chemmdl.eq.4) then
           if (kf(i).eq.0.0) then
                if (sh2o(i).eq.0.0) then
                             err = 4
                               return
                else
                     csw(i) = ctotal(i) / sh2o(i)
                     cad(i) = 0.0
                end if
           else if (nf(i).eq.1.0) then
                if (sh2o(i).eq.0.0) then
                     csw(i) = 0.0
                     cad(i) = ctotal(i) / (rho(i)*zz(i))
                else
                     csw(i) = ctotal(i) / (sh2o(i) + zz(i)*rho(i)*
     1                        kf(i))
                     cad(i) = (ctotal(i)-(csw(i)*sh2o(i))) / (rho(i)*
     1                        zz(i))
                end if
           else
                if (sh2o(i).eq.0.0) then
                     csw(i) = 0.0
                     cad(i) = ctotal(i) / (rho(i)*zz(i))
                else if (ctotal(i).eq.0.0) then
                     csw(i) = 0.0
                     cad(i) = 0.0
                else
                     call modnewt(err)
                               if ( err .ne. 0 ) return
                     cad(i) = (ctotal(i)-(csw(i)*sh2o(i))) / (rho(i)*
     1                        zz(i))
                end if
           end if
      end if

C     ## Additional error checks
      if ((csw(i).lt.0.0).or.(cad(i).lt.0.0).or.(ctotal(i).lt.0.0)) then
            err = 10
            return
      end if

      return
      end


C     ***** Modified Newton-Raphson Method Subroutine

      subroutine modnewt(err)

      common irrdate, irramt, et, ciw, ph, sph, zz, fc, h2omin, k, q,
     1       rho, sh2o, csw, cad, ctotal, khb, kb, koh, qph, kf, nf,
     2       pdate, mdate, hdate, alpha, rpmax, gamma, watup,
     3       total, h2oin, h2oout, cin, cout, starth2o, startsol,
     4       excess, watcon, solcon, rp, uptake, cbi, aa, beta, a,
     5       temp, z1, z2, reluptake, h, oh, b, hb, kchb, kcb,
     6       kcoh, kcq, root1, root2, root3, mg1, mg2, mg3, phi,
     7       collen, initsol, inith2o, totet, totdrain, toth2oin,
     8       toth2oout, lf, storeh2o, totsolin, totsolout, totsolst,
     9       totad, totsoln, avefc, avesat, fcpv, satpv, errh2o,
     1       errsol, jjj, jjjj, out, printin, nirr,
     2       site, treat, ncomp, ncrop, irr, crop, i, j, unitno,
     3       prnt, diskno, loop, chemmdl, phflg, soil,
     4       kacflg, rootflg, border, fileinput, filename, ht,
     5       printout, jread, mbflg,  htitle

      implicit real*8 (a-h)
      implicit real*8 (p-z)
      implicit integer*4 (i-o)

      real*8 irrdate(100), irramt(100), et(100), ciw(100), phi(50),
     1       ph(25,100), sph(25), zz(25), fc(25), h2omin(25), k(25),
     2       q(25), rho(25), sh2o(25), csw(25), cad(25), ctotal(25),
     3       khb(25), kb(25), koh(25), qph(25), kf(25), nf(25),
     4       pdate(25), mdate(25), hdate(25), alpha(25), rpmax(25),
     5       gamma(25), watup(25), total(25), h2oin(25),
     6       h2oout(25), cin(25), cout(25), starth2o(25), startsol(25),
     7       excess, watcon, solcon, rp, uptake, cbi, aa, beta, a, temp,
     8       z1, z2, reluptake, h, oh, b, hb, kchb, kcb, kcoh,
     9       kcq, root1, root2, root3, mg1, mg2, mg3, collen, initsol,
     1       inith2o, totet, totdrain, toth2oin, toth2oout, lf,
     2       storeh2o, totsolin, totsolout, totsolst, totad, totsoln,
     3       avefc, avesat, fcpv, satpv, errh2o, errsol
```

```
        integer*4 err

      integer*4  jjj, jjjj, out, printin, nirr,  site,
     1       treat, ncomp, ncrop, irr, crop, i, j, unitno, prnt,
     2       diskno, loop, chemmdl, phflg, soil, kacflg,
     3       rootflg(25), printout, jread, mbflg

      character htitle*77, border*56, fileinput*20, filename*20, ht,

C     ## Determines the root (i.e., csw) of the Freundlich based
C        function f(x)=0 using the modified Newton-Raphson method
C        combined with the bisection method to prevent the root
C        from being a negative number

C                   phi = difference between current iteration's zero
C                         value and the previous iteration's value
C                   terror = allowable error set within source code
C
C                   xinit = initial value of root (csw)
C                   x = current value of root (csw)
C                   fx = function f(x)
C                   fpx = f'(x)
C                   fdpx = f''(x)
C                   ux = u(x) = f(x) / f'(x)
C                   upx = u'(x) = 1.0 - f(x) * f''(x) / f'(x)**2

C     ## Set tolerance error
      terror = 0.00000001

C     ## Set inital guess for x
      xinit = ctotal(i) / (sh2o(i) + zz(i)*rho(i)*kf(i))
      x = xinit

C     ## Determine coefs. for f(x)
      a1 = rho(i) * zz(i) * kf(i)
      b1 = sh2o(i)
      c1 = ctotal(i)

C     ## Solve for csw
      jjj = 1
      jjjj = 0
      phi(jjj) = 1.0

C     ## Modified Newton-Raphson method with the bisection method used
C        whenever Newton-Raphson would take the solution below zero
      do while ((dabs(phi(jjj)).gt.terror).and.(jjj.le.50))
           jjj = jjj + 1
           fx = (a1*(x**nf(i))) + b1*x - c1
           fpx = (a1*nf(i)*(x**(nf(i)-1.0))) + b1
           fdpx = a1*nf(i)*(nf(i)-1.0)*(x**(nf(i)-2.0))
           ux = fx / fpx
           upx = 1.0 - ((fx * fdpx) / (fpx**2.0))
           phi(jjj) = -ux / upx
           x1 = x + phi(jjj)
C          # Bisection routine
           if ((x1.lt.0.0).and.(dabs(phi(jjj)).gt.terror)) then
                do while (x1.lt.0.0)
                     x1 = (x1+x) / 2.0
                end do
           end if
C          # Reset approximation to new approximation
           x = x1
```

```
C          # Check for divergence
           if ((dabs(phi(jjj)).gt.dabs(phi(jjj-1)))) then
                jjjj = jjjj + 1
                if (jjjj.ge.10) then
                              err = 5
                                return
                end if
           end if
      end do

C     ## Check for a negative root (Note: concentrations cannot be
C        negative.)
      if (x.lt.0.0) then
               err = 6
               return
      end if

C     ## Check for lack of convergence
      if (jjj.ge.50) then
               err = 3
               return
      end if

C     ## Set equilibrium solution conc. equal to x
      csw(i) = x

      return
      end


C     ***** Read Input File Subroutine

      subroutine inroutine(err)

      common irrdate, irramt, et, ciw, ph, sph, zz, fc, h2omin, k, q,
     1       rho, sh2o, csw, cad, ctotal, khb, kb, koh, qph, kf, nf,
     2       pdate, mdate, hdate, alpha, rpmax, gamma, watup,
     3       total, h2oin, h2oout, cin, cout, starth2o, startsol,
     4       excess, watcon, solcon, rp, uptake, cbi, aa, beta, a,
     5       temp, z1, z2, reluptake, h, oh, b, hb, kchb, kcb,
     6       kcoh, kcq, root1, root2, root3, mg1, mg2, mg3, phi,
     7       collen, initsol, inith2o, totet, totdrain, toth2oin,
     8       toth2oout, lf, storeh2o, totsolin, totsolout, totsolst,
     9       totad, totsoln, avefc, avesat, fcpv, satpv, errh2o,
     1       errsol, jjj, jjjj, out, printin, nirr,
     2       site, treat, ncomp, ncrop, irr, crop, i, j, unitno,
     3       prnt, diskno, loop, chemmdl, phflg, soil,
     4       kacflg, rootflg, border, fileinput, filename, ht,
     5       printout, jread, mbflg,  htitle

      implicit real*8 (a-h)
      implicit real*8 (p-z)
      implicit integer*4 (i-o)

      real*8 irrdate(100), irramt(100), et(100), ciw(100), phi(50),
     1       ph(25,100), sph(25), zz(25), fc(25), h2omin(25), k(25),
     2       q(25), rho(25), sh2o(25), csw(25), cad(25), ctotal(25),
     3       khb(25), kb(25), koh(25), qph(25), kf(25), nf(25),
     4       pdate(25), mdate(25), hdate(25), alpha(25), rpmax(25),
     5       gamma(25), watup(25), total(25), h2oin(25),
     6       h2oout(25), cin(25), cout(25), starth2o(25), startsol(25),
     7       excess, watcon, solcon, rp, uptake, cbi, aa, beta, a, temp,
     8       z1, z2, reluptake, h, oh, b, hb, kchb, kcb, kcoh,
     9       kcq, root1, root2, root3, mg1, mg2, mg3, collen, initsol,
     1       inith2o, totet, totdrain, toth2oin, toth2oout, lf,
     2       storeh2o, totsolin, totsolout, totsolst, totad, totsoln,
     3       avefc, avesat, fcpv, satpv, errh2o, errsol

        integer*4 err

      integer*4  jjj, jjjj, out, printin, nirr,  site,
     1       treat, ncomp, ncrop, irr, crop, i, j, unitno, prnt,
     2       diskno, loop, chemmdl, phflg, soil, kacflg,
     3       rootflg(25), printout, jread, mbflg

      character htitle*77, border*56, fileinput*20, filename*20, ht

      open (diskno, file='TETRANS1.IN', status='old', readonly)

C     Read the number of irrigation/precipitation events:
      read (diskno,*) nirr

C     Read the irrigation/ppt. time (in days from time 0), amount of
C     irrigation/ppt. water (cm), evapotranspiration (cm) and
C     irrigation/ppt. water solute concentration (mg/l):
      do 9050 i=1,nirr
         read (diskno,*) irrdate(i), irramt(i), et(i), ciw(i)
 9050 continue

C     Read the number of increments comprising the soil profile:
      read (diskno,*) ncomp

C     Read the pHs for each soil increment, by event:
      do 9120 i=1,nirr
         do 9110 j=1,ncomp
            read (diskno,*) ph(j,i)
 9110    continue
 9120 continue

C     Read in the type of chemical model:
      read(diskno,*) chemmdl

C     For each increment read the increment thickness (cm), field-capacity
C     water content (cm/cm), minimum water content allowed (cm/cm),
C     Langmuir affinity constant (l/mg), Langmuir adsorption maximum (mg/kg),
C     Keren KHB (ml/mg), Keren KB (ml/mg), Keren KOH (ml/mg), Keren adsorption
C     maximum (mg/kg), Freundlich K (ppm) & n (dimensionless), soil bulk
C     density (g/cm^3), initial soil water content (cm/cm), initial soil water
C     solute conc. (mg/l), initial soil pH and piston-mixing factor:
      do 9175 i=1,ncomp
         read (diskno,*) zz(i),fc(i),h2omin(i),k(i),q(i),khb(i),kb(i),koh(i),
     &                   qph(i),kf(i),nf(i),rho(i),sh2o(i),csw(i),sph(i),gamma(i)
 9175 continue

C     Read the number of crops:
      read (diskno,*) ncrop

C     For each crop, read the planting, maturing and harvesting times (in days
C     from time 0), the maximum plant root penetration (cm), the root uptake
C     model (2 = exponential uptake, 3 = linear uptake) and the mature plant's
C     root uptake profile coefficient for that model:
      do 9290, i=1,ncrop
         read (diskno,*) pdate(i),mdate(i),hdate(i),rpmax(i),rootflg(i),alpha(i)
 9290 continue

      close (diskno)
      return
      end


C**** Write the raw after irrigation data to the output file:
      subroutine savedat1()

      common irrdate, irramt, et, ciw, ph, sph, zz, fc, h2omin, k, q,
     1       rho, sh2o, csw, cad, ctotal, khb, kb, koh, qph, kf, nf,
     2       pdate, mdate, hdate, alpha, rpmax, gamma, watup,
     3       total, h2oin, h2oout, cin, cout, starth2o, startsol,
     4       excess, watcon, solcon, rp, uptake, cbi, aa, beta, a,
     5       temp, z1, z2, reluptake, h, oh, b, hb, kchb, kcb,
     6       kcoh, kcq, root1, root2, root3, mg1, mg2, mg3, phi,
     7       collen, initsol, inith2o, totet, totdrain, toth2oin,
     8       toth2oout, lf, storeh2o, totsolin, totsolout, totsolst,
     9       totad, totsoln, avefc, avesat, fcpv, satpv, errh2o,
     1       errsol, jjj, jjjj, out, printin, nirr,
     2       site, treat, ncomp, ncrop, irr, crop, i, j, unitno,
     3       prnt, diskno, loop, chemmdl, phflg, soil,
     4       kacflg, rootflg, border, fileinput, filename, ht,
     5       printout, jread, mbflg, htitle

      implicit real*8 (a-h)
      implicit real*8 (p-z)
      implicit integer*4 (i-o)

      real*8 irrdate(100), irramt(100), et(100), ciw(100), phi(50),
     1       ph(25,100), sph(25), zz(25), fc(25), h2omin(25), k(25),
     2       q(25), rho(25), sh2o(25), csw(25), cad(25), ctotal(25),
     3       khb(25), kb(25), koh(25), qph(25), kf(25), nf(25),
     4       pdate(25), mdate(25), hdate(25), alpha(25), rpmax(25),
     5       gamma(25), watup(25), total(25), h2oin(25),
     6       h2oout(25), cin(25), cout(25), starth2o(25), startsol(25),
     7       excess, watcon, solcon, rp, uptake, cbi, aa, beta, a, temp,
     8       z1, z2, reluptake, h, oh, b, hb, kchb, kcb, kcoh,
     9       kcq, root1, root2, root3, mg1, mg2, mg3, collen, initsol,
     1       inith2o, totet, totdrain, toth2oin, toth2oout, lf,
     2       storeh2o, totsolin, totsolout, totsolst, totad, totsoln,
     3       avefc, avesat, fcpv, satpv, errh2o, errsol

      integer*4  jjj, jjjj, out, printin, nirr,  site,
     1       treat, ncomp, ncrop, irr, crop, i, j, unitno, prnt,
     2       diskno, loop, chemmdl, phflg, soil, kacflg,
     3       rootflg, printout, jread, mbflg

      character htitle*77, border*56, fileinput*20, filename*20, ht

C**** Write simulation data to file:
      write(99, '(f15.7)') rp
      do 10 i = 1, ncomp
         write(99, '(3f15.7)') csw(i), cad(i), ctotal(i)/zz(i)
   10 continue

C**** Write leaching fraction and pore volume data to file:
      write(99, '(2f15.7)')  lf, satpv

C**** Write mass balance data to file:
      write(99, '(6f15.7)') toth2oin,totet,totdrain,toth2oout,storeh2o,errh2o
      write(99, '(6f15.7)') totsolin,totsolout,totsoln,totad,totsolst,errsol

      return
      end
```

```
C**** Write the raw after et data to the output file:
      subroutine savedat2()

      common irrdate, irramt, et, ciw, ph, sph, zz, fc, h2omin, k, q,
     1       rho, sh2o, csw, cad, ctotal, khb, kb, koh, qph, kf, nf,
     2       pdate, mdate, hdate, alpha, rpmax, gamma, watup,
     3       total, h2oin, h2oout, cin, cout, starth2o, startsol,
     4       excess, watcon, solcon, rp, uptake, cbi, aa, beta, a,
     5       temp, z1, z2, reluptake, h, oh, b, hb, kchb, kcb,
     6       kcoh, kcq, root1, root2, root3, mg1, mg2, mg3, phi,
     7       collen, initsol, inith2o, totet, totdrain, toth2oin,
     8       toth2oout, lf, storeh2o, totsolin, totsolout, totsolst,
     9       totad, totsoln, avefc, avesat, fcpv, satpv, errh2o,
     1       errsol, jjj, jjjj, out, printin, nirr,
     2       site, treat, ncomp, ncrop, irr, crop, i, j, unitno,
     3       prnt, diskno, loop, chemmdl, phflg, soil,
     4       kacflg, rootflg, border, fileinput, filename, ht,
     5       printout, jread, mbflg, htitle

      implicit real*8 (a-h)
      implicit real*8 (p-z)
      implicit integer*4 (i-o)

      real*8 irrdate(100), irramt(100), et(100), ciw(100), phi(50),
     1       ph(25,100), sph(25), zz(25), fc(25), h2omin(25), k(25),
     2       q(25), rho(25), sh2o(25), csw(25), cad(25), ctotal(25),
     3       khb(25), kb(25), koh(25), qph(25), kf(25), nf(25),
     4       pdate(25), mdate(25), hdate(25), alpha(25), rpmax(25),
     5       gamma(25), watup(25), total(25), h2oin(25),
     6       h2oout(25), cin(25), cout(25), starth2o(25), startsol(25),
     7       excess, watcon, solcon, rp, uptake, cbi, aa, beta, a, temp,
     8       z1, z2, reluptake, h, oh, b, hb, kchb, kcb, kcoh,
     9       kcq, root1, root2, root3, mg1, mg2, mg3, collen, initsol,
     1       inith2o, totet, totdrain, toth2oin, toth2oout, lf,
     2       storeh2o, totsolin, totsolout, totsolst, totad, totsoln,
     3       avefc, avesat, fcpv, satpv, errh2o, errsol

      integer*4  jjj, jjjj, out, printin, nirr,  site,
     1       treat, ncomp, ncrop, irr, crop, i, j, unitno, prnt,
     2       diskno, loop, chemmdl, phflg, soil, kacflg,
     3       rootflg, printout, jread, mbflg

      character htitle*77, border*56, fileinput*20, filename*20, ht

C**** Write data to file:
      do 10 i=1,ncomp
         if ( et(irr) .eq. 0.0 ) then
            h2oup = 0.0
         else
            h2oup = (watup(i)/et(irr))*100.0
         end if
         write(99, '(4f15.7)') h2oup, sh2o(i)/zz(i), csw(i), cad(i)
   10 continue

      return
      end


C     Error Message Subroutine

      function errmess(err, irrnum)
      integer*4 err
      integer*4 irrnum
      string*200 mess
      character*200 msg

      if (err.eq.1) then
            write(msg,'(a,i3)') 'MODEL ERROR: Irrigation event has insufficient wa
            mess = msg
            call ErrorDlog(mess)
            return

      else if (err.eq.2) then
            write(msg,'(a,i3)') 'MODEL ERROR: Insufficient water in column to meet
            mess = msg
            call ErrorDlog(mess)
            return

      else if (err.eq.3) then
            write(msg,'(a,i3)') 'ERROR: Lack of convergence in Newton-Raphson meth
            mess = msg
            call ErrorDlog(mess)
            return

      else if (err.eq.4) then
            write(msg,'(a,i3)') 'MODEL ERROR: Improbable scenario - No adsorption
            mess = msg
            call ErrorDlog(mess)
            return

      else if (err.eq.5) then
            write(msg,'(a,i3)') 'MODEL ERROR: Divergence in Newton-Raphson method.
            mess = msg
            call ErrorDlog(mess)
            return

      else if (err.eq.6) then
            write(msg,'(a,i3)') 'MODEL ERROR: Newton-Raphson approximation less th
            mess = msg
            call ErrorDlog(mess)
            return

      else if (err.eq.10) then
            write(msg,'(a,i3)') 'MODEL ERROR: Adsorbed, solution or total concentr
            mess = msg
            call ErrorDlog(mess)
            return
      end if
      return
      end


C********************** Routines which access the Mac toolbox follow **************

C     CheckAbort
C     Check for command-. or command-q key press in event queue.

      subroutine CheckAbort(err)

!!SETC USINGINCLUDES = FALSE
      include 'events.f'

      record /EventRecord/ the_event
      integer*4   err
      logical*1   an_event
      integer*2   chcode, event_mask
      character*1 ch

      event_mask = $0008      ! Check for key-down event only
      an_event = getnextevent(%val(event_mask),%ref(the_event))
      if (an_event) then
         if (the_event.what = 3) then     ! key press
            chcode = jiand(the_event.message,charCodeMask)
            ch = char(chcode)
            chcode = jiand(the_event.modifiers,cmdKey)
            if ((chcode .NE. 0).and.((ch = '.').or.(ch = 'Q').or.(ch = 'q'))) err
         end if
      end if
      return
      end


!     ErrorDlog
!     Post an error alert

      subroutine ErrorDlog(str0)
      string*200 str0,str1,str2,str3
      integer*2  alertID
      integer*4  nil,ret

      nil = 0
      alertID = 5000
      str1 = ' '
      str2 = ' '
      str3 = ' '
      call InitCursor
      call ParamText(%val(str0),%val(str1),%val(str2),%val(str3))
      ret = StopAlert(%Val(alertID), %Val(nil))
      return
      end
```