**User Data Input Tutorial**

The comma separated variable files in this ***ReadMeFirst!!*** folder are suggested starting points for the new user to study the file structures and modify them appropriately to build your own system. Make changes very gradually so that you will always have a working set of files to be able to return to! The most common mistake is to have sets of commas at the end of a data record, which causes the code to try to read null values. The end of every record should be a numerical value, not a comma.

These files parameterize a simple, 1-D soil column model. Compartments 1 and 12 are upper and lower boundary compartments, respectively. Compartments 2 – 11 are interior, “modeled” compartments where concentrations are simulated. Linear, equilibrium partitioning is assumed with uniform Kd values throughout as shown in ***LinearKdandTempCoef.csv***. Volatilization from compartment 2 into (boundary) compartment 1 is included in ***AreaSrcSnks.csv***. 1st order decay is included in all modeled compartments in ***VolumeSrcSnk.csv***. The system is driven by an initial concentration of 100 g/m3 in Compartment 2, as in ***Initial.csv***. Flows are uniform throughout the simulation and are given in ***Flows.csv***. Boundary values are given in ***Boundary.csv***.

There are a total of 13 ***\*.csv*** input files that must be present for the GEM to run. To a large extent, these files are self-explanatory from the 1st record, which consists of text describing each data field. These files are shown below for the recommended starting system (input ***\*.csv*** files in this ***Start Here!*** folder). Brief explanations of the formats and requirements for each file are noted below. Note that “-999” is used for entries that are not relevant, but must be included given the underlying data structures. More complete description is available in the CRC Press publication, “Environmental Fate and Transport Analysis with Compartment Modeling” (Little, 2012) as noted in the JEMS paper. GEM input units are meters (m), grams (g), and days. No unit corrections occur in the code. Output concentrations are g/m3.

***Input File 1: Control.csv***

This file provides macro data to the GEM regarding numbers of compartments, solution options, time step sizes, number of time steps, etc. Each data record is preceded by a text record denoting the subsequent data entries. The recommended starting example uses the EnvS (environmental system) option (record 2), is linear (record 4), involves 1 state variable, 10 “modeled” (nonboundary or nondummy) compartments, and 2 boundary compartments (record 10). Record 18 provides the temporal numerical solution option (back time or BT). Record 20 is the time step size in days (10) and record 22 is the number of 10-day time steps for the complete simulation (100). The remaining records pertain to nonlinear problems and/or problems where the GEM fires off (“shells”) an external, user-provided model as part of its functionality. These are advanced topics that the interested reader may investigate in the CRC Press document (Little, 2012).



***Input File 2: SVCompMap.csv***

This is another macro data file that specifies which state variables (SV) are relevant to which compartments. Following the field 1 descriptive record there is one record for every state variable. Our starting problem has a single state variable. The second record for the example gives the SV number in field 1 (1) followed by either 0 or 1 for each of the 12 compartments. In the example, SV 1 is relevant (is modeled) in compartments 2 – 11 and those field entries are “1”. SVs will never be relevant (modeled) in boundary or dummy compartments (1 and 12 in our example) and those entries are always “0”.



***Input File 3: Compartments.csv***

This file gives compartment-specific information. Following the 1st , descriptive record there is one record for each compartment. The first field is the compartment number. These must be sequential starting with “1”. The second field is the number of other compartments adjacent to the compartment. In our starting example, all modeled compartments (2-11) have 2 adjacent compartments while the boundary compartments (1 and 12) have only 1. The third field denotes the type of compartment. A “modeled” compartment is type “0”. A “dummy” compartment (not a boundary or modeled compartment, but needed otherwise to provide, e.g. flow inputs or receive outputs) is type “1”. A fixed boundary concentration-type compartment is “2”. Types 3 and 4 are zero gradient and linear gradient boundary compartments, respectively. Fields 4, 5, and 6 give spatial information to the GEM. The user will identify the centroid of some (any) compartment as a reference and that will have x,y,z coordinates of 0,0,0. The centroid coordinates of other compartments are then provided relative to this baseline. Negative values are fine. These coordinate data are used to calculate dispersion/diffusion mixing distances, when MixLengthOption = 1 in record 14 of ***Control.csv***, and to write output files. In all input files, “-999” is used to denote irrelevant data. For example, in the starting example dispersion is not modeled across the compartment 1,2 interface so “-999” is used for the coordinates of compartment 1. (It is certainly acceptable to have real coordinates for these compartments, but is not needed.) The following 4 fields provide compartment-specific volume (total volume), temperature, water content, and bulk density. Temperature is used to make temperature corrections for 1st order decay or transformation processes using a Van’t Hoff-type temperature correction. Entering 20 deg C negates any such corrections.



The next 3 files, ***Interfaces.csv, Flows.csv, and Ecoefficints.csv***, have a similar structure and should be built in concert. Because each of these has a “ragged, right-edge” (records may involve different numbers of fields), a common user input error is to save the file with Excel in csv format where Excel will then by default make each record have the same number of fields as the maximum in the file. Fields not relevant to a given record may then involve data such as “,,,,”. These false fields are then read by the GEM as actual input data and it will crash. (Hopefully with an error statement.) After you build these ragged, right-edge files, edit them to delete any trailing commas. Each record ends with a data value, not a comma. In addition, these three files need to have the same structure with respect to compartment number followed by adjoining compartment numbers. For example, our compartment 2 has 2 adjacent compartments, 1 and 3. If the record for compartment 2 has the first adjacent compartment as 1 and the second as 3, then that pattern needs to be maintained for all three files. Each record is specific to a compartment “i” and involves data related to all compartments “j” adjacent to i. The length of the records is user-determined and varies according to the number of adjacent compartments.

***Input File 4: Interfaces.csv***

This file provides data specific to compartment interfaces. After the 1st descriptive record, compartment numbers are entered sequentially beginning with “1”. For each compartment’s record, fields 2 – 6, provide the adjacent compartment number, interfacial area between compartment “i” and “j”, the distance from compartment i’s centroid to the i,j interface (used in computing dispersion mixing length for MixLengthOption = 1), Mixing Length, and “alpha”. If MixLengthOption = 1 (***Control.csv***) then the mixing length is irrelevant and “-999” is applicable. If the user desires to specify the mixing length (MixLengthOption = 2), then the actual mixing length is entered here. Alpha = 1 is a back space (BS) method while alpha = 0.5 is a center space (CS) method. These 5 fields are then repeated for each adjacent compartment “j” as needed.



***Input File 5: Flows.csv***

Static flows across compartment interfaces are input via this file. (For dynamic flows, an advanced feature of the GEM using the shell functionality – see records 55 and 56 in Control.csv -- to dynamically update Flows.csv using an external program is available.) After the description record, there is one record per compartment. Different flows for different SVs are possible. Field 1 is the SV number. Fields 2 and 3 give the “i,j” compartment numbers. With respect to compartment “i”, if a flow is leaving “i” to “j” it has a negative value. If entering “I” from “j”, it has a positive value. The i,j data are entered on the same record for all adjacent compartments j. It should be noted that an internal flow balance is performed around all nonboundary/nondummy compartments (modeled compartments) using a flow balance tolerance as specified in ***Control.csv***. If this tolerance is not satisfied, the GEM will error-out with an error statement.



***Input File 6: ECoefficients.csv***

This file is analogous to the Flows.csv file and specifies dispersion/diffusion coefficients across the compartment interfaces. In general, if the dispersion coefficient across, e.g. the compartment 2/compartment 3 interface, is specified as, e.g. 1.00E-04 m2/day, in record 3, then the same value would be specified as the compartment 3/compartment 2 parameter in record 4. To the extent that these values are the same, the file has significant redundancy. However, in some cases, it is conceivable that the dispersion value from i to j might be different than from j to i and this design would be needed.



***Input file 7: FlowandEMultipliers.csv***

This file exists to allow the user to multiply the flow values (in ***Flows.csv***) and/or dispersion values (in ***Ecoefficients.csv***) by arbitrary scaling factors. In our example, and in the general case, such multipliers are not relevant. However, the file itself must exist. The first record is, again, descriptive text. The GEM reads the first record. It then reads subsequent records until it reads “-999” in the first field after which it closes. If there were SVs and compartments for which multipliers are desired, these would be entered between the initial, description record and the terminating “-999” record. The number of these intermediate records is up to the user.



***Input file 8: VolumeSrcSnks.csv***

This is another file with a user-determined number of records, ending with a “-999” value. The 1st order rate coefficients (1/day) entered here are used in volume-based source sink processes, e.g. 1st order decay, where that term in the relevant mass balance equation involves compartment volumes . For our example, we assume a decay rate of 0.05 for all modeled compartments (2-11). The file is set up to accommodate multiple, interacting SVs. A process involving a 1st order transfer from SV i to SV j would enter the rate constant with a negative sign with respect to i (it is a sink for i) and a positive sign with respect to j (where it is a source). Such an across-SV reaction would require two records – one for source compartment i and another for sink compartment j. For our simple example with a single SV with decay, the “First SV” and “Second SV” have the same values.



***Input File 9: AreaSrcSnks.csv***

This file is analogous to ***VolumeSrcSnks.csv*** and provides parameters related to “area-based” sources or sinks. For our example, there is a single compartment that involves an across-compartment transfer due to volatilization from compartment 1 to compartment 2. That term in the mass balance equation related to volatilization involves the intercompartmental area. Again, a sink with respect to First Compartment has a negative sign Because our compartment 1 is not a modeled compartment (it is a boundary compartment), we do not have a corresponding record for compartment 1 showing volatilization from 2 to 1 as a sink with respect to compartment 1. (There is not a mass balance equation for boundary compartments.) If compartment 1 were a modeled compartment, two records would be needed, with the rate constant having negative (sink) and positive (source) signs.



***Input File 10: LinearKdandTempCoeff.csv***

This file provides the state variable and compartment-specific Kd values and temperature coefficients (for the Van’t Hoff-type temperature correction). In our example, the same Kd value is assumed for all modeled compartments. For each SV listed in field 1, all (modeled and boundary/dummy) compartments must be included in sequential order as shown. Again, “-999” is used for irrelevant data.



***Input File 11: Boundary.csv***

This file is SV- and compartment-specific and provides the fixed concentration boundary values (g/m3) for type 2 boundary compartments as specified in the third fields of ***Compartments.csv***. Again, for each SV specified in field 1, all compartments must be sequentially listed. “-999” denotes irrelevancy. Our example uses fixed concentrations of 0 g/m3 in the top (1) and bottom (12) boundary compartments. Dynamic updating of these values is possible by shelling out to an external program if specified in record 63 of ***Control.csv***. (Essentially, all static parameters can be dynamically updated using this functionality if so specified in ***Control.csv***.)



***Input File 12: Initial.csv***

This file provides SV- and compartment-specific initial conditions. For each SV listed in field 1, all compartments must be sequentially listed. In our example, an initial condition in compartment 2 of 100 g/m3 is used to represent a “spill” while all other (modeled) compartments begin with no contamination.



***Input File 13: Loads.csv***

This file provides functionality to input external daily mass loadings (g/day) into (modeled) compartments dynamically. Our example has no external loadings and the “forcing function” is simply the compartment 2 initial condition discussed above. Nonetheless this file must exist, and the data shown below represent the default (no external loading) format. If there were external loads, this file would be of user-specified length. For each loading window associated with the TimeStart and TimeEnd values (days) shown in record 2 below, the related loads would be entered following. For example, if we had an external load of 1000 g/day for our SV 1 in compartment 5, the fourth record below would be 5,1,1000. Loads for an additional SV in compartment 5 would continue on the same record. (The example below shows only a single additional SV but more can be added on the same record.) The “-999” value shown below in field 4 tells the GEM to stop reading SV-specific loads for compartment “i”. The final “-999” value in field 5 below tells the GEM to stop reading compartmental records for that time window and close. If a different time window with different loadings were relevant, those records would be inserted (with the two descriptive records shown) prior to the final “-999”. (This file has the most complex structure of all input files, and more explanation is available in the CRC Press document [Little, 2018]. Some simplification is probably possible, but we have not pursued alternatives at this time.)



***Output Files***

Several output files are written by the GEM to the same folder containing the 13 input files. ***AllTimeSeries.csv*** contains simulated concentrations for all modeled compartments at all time steps. Two other, smaller, output options are available as specified in ***Control.csv***. MassBalance.csv provides mass balance statistics by SV for the simulation duration. (There is a steady-state option for GEM as parameterized in ***Control.csv***. For the steady-state option, no mass balance data are provided.) ***TransportFluxToBoundaryCompartments.csv*** provides mass transfer information (both from fate and transport processes) from modeled compartments to boundary/dummy compartments at each time step of the simulation, for each SV. Finally, several diagnostic files (\*.dng) are written to assist the user in debugging/understanding. Among these, the “A” matrix of coefficients and the “forcing function” vector of external loadings/initial conditions are written for the user-specified time step in record 24 of ***Control.csv*** as ***Aout.dng*** and *Bout.dng*, respectively. If the user adequately understands the underlying set of mass balance equations that are being solved, these files may be used to verify that the problem has been correctly parameterized in the various input files.