tran-SAS 1.0 Documentation

17-11-2017

This documentation is an exported version of the tran-SAS GitHub wiki. Up-to-date versions can be found directly at the wiki page.

Table of Contents

QUICK START

Introduction

The *tran-SAS* package includes a set of codes to model tracer transport and residence time distributions in hydrologic systems, using a StorAge Selection (SAS) function approach. The code is written in MATLAB and is intended to be an easy-to-use tool for researchers who do not have deep knowledge of SAS function approaches. The package includes a basic model of solute transport for tracer inputs entering the system through atmospheric precipitation, but it can be edited by the user to deal with more complex (e.g. non-conservative) solutes.

Running the model

The model is ready-to-go and can be run by launching the model starter (`model_STARTER.m`). The starter prepares and runs the SAS model for the specified case study. Simulation output is automatically saved and visualized through a simple plotting script.
For a quick startup, the user can open the model starter and run the default case study (provided with the code) for different SAS functions and parameter combinations. The simplest way to modify the model parameter is to go to block 3-Parameters and select or modify one of the examples.

USER MANUAL

Data Requirements

The model requires continuous hydrologic fluxes and tracer input at hourly timestep. Input timeseries can derive from measurements or hydrologic model outputs (e.g. an evapotranspiration model). Data gaps have to be filled. Optional variables like an index of system wetness and measured solute output have to be provided even if they are not used (see below how to fill-in the data table in such a case). The input should be formatted as in the case study example `testdata.dat`:

```
  date [yyyy-mm-dd HH:MM] J [mm/h] Q [mm/h] ET [mm/h]  Cin [-]   wi [-]  measC_Q [-]
         2012-09-30 00:00     0.00     0.04      0.00    56.03     0.38        50.07
         2012-09-30 01:00     0.00     0.04      0.00    56.02     0.38      -999.00
         ...
         ...
```

where the different columns indicate:

- *date*: the time frame (yyyy-mm-dd HH:MM) corresponding to each data point. This indicates the beginning of each hourly timestep
- *J*: precipitation input, in mm/h
- *Q*: discharge, in mm/h
- *ET*: evapotranspiration, in mm/h
- *Cin*: measured tracer concentration of the precipitation input. The units are chosen by the user (e.g. mg/l) and the model concentration output will be in the same units. In case an hourly input is not available (e.g. because measurements are taken daily or weekly), the timeseries need to be interpolated to hourly timesteps.
- *wi*: (*optional*) index of system wetness, which is needed in case a time-variant SAS function is used. If *wi* is not needed by the user, the column can be simply filled with a constant value (e.g. 0). The index must be normalized to the interval (0,1), where the values 0 and 1 correspond to the driest and wettest system conditions, respectively. A simple index can be obtained by normalizing the storage variations of the system or by normalizing soil moisture measurements.
- *measC_Q*: (*optional*) measured tracer concentration in streamflow. Modeled tracer outputs are compared to data for each available measurement. A no-data value of -999.00 is to be used for all time steps where measured output is not available.

Codes organization

The model package includes the following scripts and folders:

- model_STARTER.m, fundamental script where all model parameters and preferences can be manually set
- plot_results.m, script to launch some basic output display
- **case_study**, folder in which each case-study data file has to be inserted
- **functions**, folder where SAS functions and all other auxiliary functions are stored. The file show_SAS_shape.m can be used to call and display the different available SAS functions
- **results**, folder where model output files (default "all_output.mat") are saved
- **models**, folder where the age Master Equation models (default `SAS_EFs.m` and more advanced `SAS_odesolver.m`) are stored

the model STARTER

The model starter is organized in 7 blocks:

1. general simulation settings
2. data import and aggregation
3. parameters
4. (optional) select dates in which to save age computations
5. set/generate initial conditions
6. run the model
7. display some results

1. general simulation settings

Main simulation settings can be set in this block. These include the choice of the dataset and the model to be run. The standard available model is the modified Euler-Formard scheme `SAS_EFs.m`. The aggregation timestep should be small enough to avoid numerical inaccuracies but high enough to allow fast model runs. Depending on the application, the range 6-24 hours should be considered as a reference starting point. A higher order numerical solution can be obtained using the model `SAS_odesolver.m`, which has a better accuracy but it is about 50 times slower. The parameter `f_thresh` can be used in longer simulation to merge the contribution of older waters into a single undifferentiated "old" pool. This allows speeding up computations. For example, f_thresh=0.9 means that the

oldest 10% of the water storage is considered as one single water parcel (instead of hundreds of very little parcels). Use f_thresh=1, if you don't want to merge the older parcels.

2. data import and aggregation

Here, the selected datafile (e.g. testdata.dat) is scanned and aggregated by calling the external function data_aggregate.m. Errors may arise here if the data file is badly formatted. Aggregated data are then inserted in the structure 'data', which makes it easy to pass several variables to the model function. Nothing needs to be modified in this block.

3. parameters

To illustrate possible SAS functions and parameter combinations, three simple examples are provided in the code. The user can either modify the existing examples or create new ones.

Each simulation requires three types of parameters:

- SAS function parameters for Q flux (the number of parameters depends on the choice of the SAS function)
- SAS function parameters for ET flux (the number of parameters depends on the choice of the SAS function)
- other parameters (initial storage S0)

Example 2 is detailed here:

```
if example==2
    % select the SAS
    data.SASQName='fSAS_pltv'; %time-variant power-law SAS
    data.SASETName='fSAS_pl'; %power-law SAS

    % parameter names and values
    SASQparamnames={'kmin_Q','kmax_Q'}; Pars(1:2)=[0.3,0.7]; %[-]
    SASETparamnames={'beta_ET'};        Pars(3)=1; %[-]
    otherparamnames={'S0'};             Pars(4)=1000; %[mm]
end
```

SASQ and SASET were defined in example 2 as time-variant power-law ('fSAS_pltv', requiring 2 parameters) and simple power-law ('fSAS_pl', 1 parameter). Another built-in possibility would have been the beta function ('fSAS_beta', see 'choice of the SAS function' for additional details). For each parameter type, all parameter names and values need to

be specified. The parameter name is just used to identify the parameter and has no consequence on the model run.

choice of the SAS function

Three SAS function types are provided within the code:

- `fSAS_pl`: this is the simplest function, requiring just one parameter k>0. The function expresses preference for younger ages in storage when 0<k<1 and preference for older ages when k>1. The special case k=1 is equivalent to a random sampling of all the stored ages. The user is advised not to push the function to its analytical limit k-->0 to avoid incurring in numerical errors. Values k<0.2 should not be necessary in typical catchment-scale applications.
- `fSAS_pltv`: this is like a simple power law, but the parameter k changes in time ranging from a minimum value kmin to a maximum value kmax, depending on the pre-defined wetness index (wi, see input requirements). Depending on the choice of the 2 parameters and on the wetness index, this function type allows enhancing the preference for younger/older ages during wet/dry periods.
- `fSAS_beta`: this function has 2 different parameters (a>0 and b>0) that express affinity for the younger and older storage components respectively. The lower the parameter value, the higer the affinity for that storage component. For example, a beta distribution with a=0.6 and b=1 expresses a preference for younger ages while a=2 and b=0.8 has preference for older waters. The special case a=b=1 is equivalent to a random sampling of all the stored ages.

The visualization of the SAS function is eased by using the script `show_SAS_shape.m`

4. (optional) select dates in which to save age computations

To speed up computations and reduce data storage requirements, age distributions are not saved during the computations. However, the user can define a set of dates where discharge age distributions are saved. This is optional and no distributions are saved in case a non-existing date is assigned.

5. set/generate initial conditions

The model needs an initial storage concentration and age distribution. The initial storage is considered as a single water parcel with age=0 and its concentration is defined by the parameter `data.C_S0`. A spinup can be generated to warm-up the model for one or several years before the actual computations start. This is done by including some spinup data at the beginning of the dataset. The spinup data is generated by repeating a selected period

(`period_rep`) for a number (`n_rep`) of times. The use of the spinup reduces the dependence on the initial conditions.

6. run the model

This block simply launches the simulation by calling the external model function. No output is returned by the model function, and model results are saved from within the function. Nothing needs to be modified in this block.

7. display some results

This block is used to load model results, compute some model efficiency (if streamflow solute concentrations are available), and display the model output by calling the external function 'plot_results.m'. Nothing needs to be modified in this block.

the model function SAS_EFs

The model function `SAS_EFs.m` implements the age Master Equation using a modified Euler-Forward numerical routine. The inputs to the function are the parameter vector (`Pars`) and a structure (`data`) including all data and auxiliary variables used in the code. The main elements of the function implementation are listed below.

solve the age balance and evaluate the rank storage concentration

1. the SAS function domain (`dom`) is prepared. The SAS domain in this code is the normalized rank storage computed at previous timestep, with the addition of event water in the first element (`age1`). The event water is computed through a water balance for current precipitation, using the SAS functions evaluated at previous timestep. This modification is the only difference from a classic Euler Forward scheme.
2. the SAS functions (`Omega_Q` and `Omega_ET`) are computed for each element of the vector dom.
3. the fundamental balance is implemented using vector operations. In simple terms, the rank storage at the beginning of next time step (j+1) is obtained from the rank storage at the beginning of current time step (j) by subtracting the water volumes corresponding to discharge (data.Q(j)*Omega_Q) and evapotranspiration (data.ET(j)*Omega_ET). Moreover, new precipitation inputs (data.J(j)) are introduced. To ensure consistency in the balance, the rank storage is forced to be non decreasing.

4. the solute concentration of each individual water parcel (`C_ST`) gets updated. Concentration does not change with time for tracers but, for non-conservative solutes, biogeochemical reactions can be implemented here.

5. the length of the rank storage vector can be managed to avoid unlimited growth. According to the value of the numerical parameter f_thresh, the rank storage exceeding the threshold f_thresh is merged into one parcel and the mean parcel concentration is considered. Accordingly, the length of vectors Omega_Q and Omega_ET is adjusted.

compute output: stream concentration

Stream concentration is obtained by first computing which storage portions that are released to discharge (i.e. the stream age distribution `pQ`) and then by performing a scalar product with the element concentration C_ST. To speed up computations and reduce storage needs, stream age distributions are not stored (except for an optional selection). Hence, the code needs to be edited to allow storing the whole stream age structure.

compute output: other

Many other outputs can be computed, such as ET age distributions, and timeseries of age metrics like the median age. Some of these are provided in the model function but they are commented out by default to speed up che computations. Important note: to allow saving other outputs (either built-in or new outputs created by the user), such outputs need to be included among the list of saved variables (`varlist`) at the end of the script.

the model function SAS_odesolver

The model function `SAS_odesolver.m` uses a built-in MATLAB ode solver to solve the age Master Equation. The code is organized in the same way as the SAS_EFs code and only differs in the way the rank storage S_T is computed. This higher-order implementation is slower but more accurate so it can be conveniently used as a control on the numerical accuracy of the Euler Forward scheme.

results display

The model includes a simple script to display the computed stream concentration and the selection of stream age distributions. The code is automatically launched after running the starter (provided the flags flag_plot.Cout and flag_plot.TTDs are set to 1 in the starter), but it can also be run independently. In this case, a different output filename can be specified at the beginning of the script.

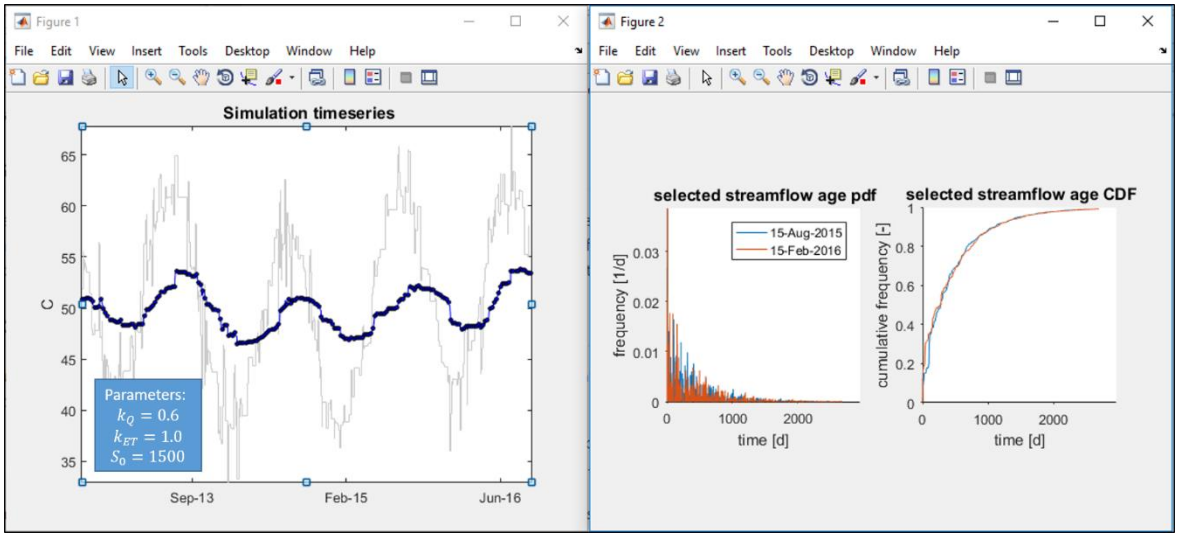
examples

The default case study can be used to assess the differences among systems with different storages and storage affinities. Two examples are here provided:

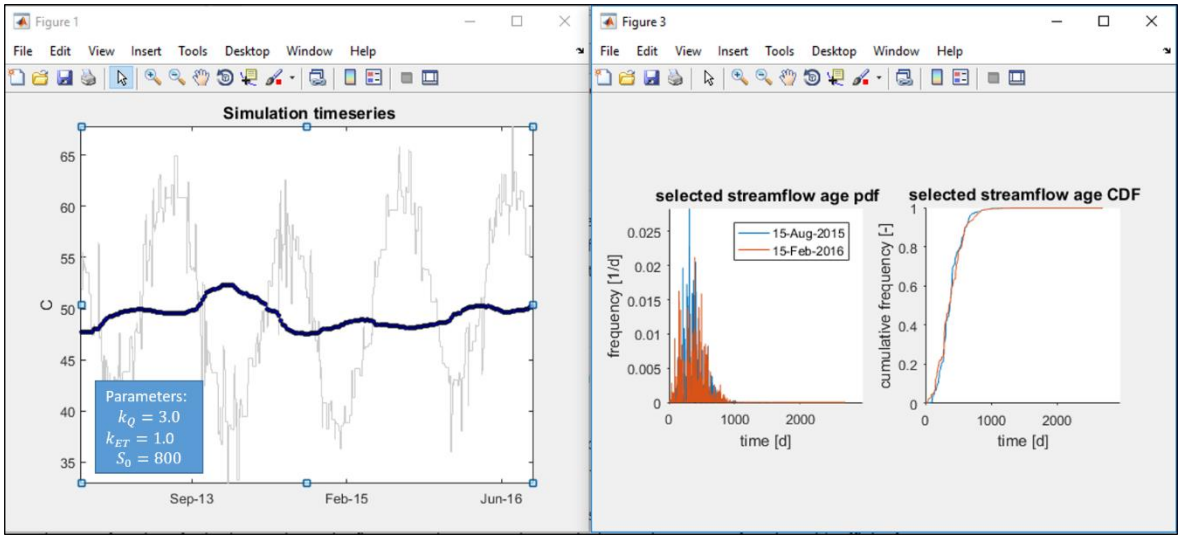
1. large storage that supplies stream with younger water parcels. This is representative of a typical catchment behavior in a wet climate
2. smaller storage that supplies stream with older water parcels. This is representative of a percolation process in e.g. a lysimeter

Both examples are run with a 4-year spinup period (obtained by repeating the first 2 years of data twice) and using parameter example 1 (simple power-law SAS functions for both Q and ET). The first example (wet catchment) is done using a SASQ function with affinity for young ages (k_Q=0.6), a SASET function with neutral affinity (k_ET=1) and a 1.5 meter storage (S0=1500). The second example is done using a SASQ function with affinity for older ages (k_Q=3), a SASET function with neutral affinity (k_ET=1) and a 0.8 meter storage (S0=800). The default output figures are shown below for the two examples. To simplify visualization, measured stream concentration was removed from the plots.

In the first example, stream concentration is damped and the signal is almost synchronous with precipitation concentration. The corresponding age distributions include many young elements but also a long old water tail (20% of the distribution is older than 3 years).



In the second example, stream tracer response is strongly damped and delayed (about 6 months) with respect to precipitation concentration signal. Age distributions are less spread out than in the previous case, having low young water contribution and also low contributions from water older than 3 years.



editing the codes

The codes are intended to be an illustrative tool to be edited. The code structure makes it easy to implement different model functions (e.g. with a different numerical implementation, or with a solute cycling component) and call the different functions from the same model starter. The user is encouraged to edit the codes!