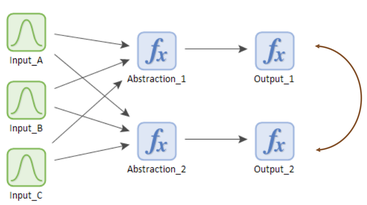
A Simpler Way to Implement Process Models in GoldSim

Aaron Bandler, Gregg Occhiogrosso, Megan Higgs, Hayley Brittingham, Will Barnett, John Tauxe | Neptune and Company, Inc.

The Challenge of Abstraction

System-level Radiological Performance Assessment (PA) models rely on specific process level models for some inputs, such as streamflow. For probabilistic PA, the value of the process model output depends on the values of input parameters that also vary in the PA model. The simplest conceptual solution is to draw values of the input parameters from their distributions, run the process level model at these values, and then use the output where it is needed in the PA model. However, this solution is unsatisfactory for multiple reasons: (1) the PA Model should be set up such that new runs can be made and distributions can change without having to go back and run the process-level model; (2) calling the process model from the system model is possible, but technical difficulties remain in coupling the two; and (3) there is no uncertainty in the process-level model results incorporated into the PA model. Therefore, **it is desirable to develop a method that maps the values of PA Model inputs to process level model outputs in real time within the PA Model set up. It is also important to use as simple of a map as possible without losing key features of the relationship.** This has been termed "model abstraction" because information from process model runs is "abstracted" to create the map used within the PA model.

A common method for model abstraction is the use of statistical models, such as linear regression, which have performed well in many settings where the process is actually driven by just a few inputs that are simply modeled. **The problem gets more challenging when the number of inputs and outputs that need to be modeled gets large and when there are physical constraints among the many process-level outputs that needs to be maintained for the PA model (e.g. mass balance).**

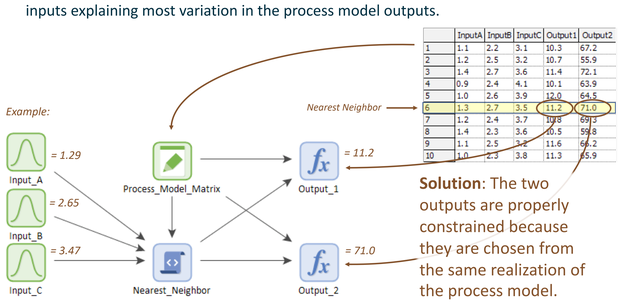


Challenge: How do we maintain relationships and constraints among Outputs? The addition of more inputs and more interrelated outputs quickly leads to a very challenging multivariate problem.

A simple example of a straightforward abstraction. Expressions for abstractions 1 and 2 may be common regression equations or other common univariate or multivariate statistical models.

A Simple, "Nearest Neighbor" Approach:

1. The process-level model is run at a collection of input values meant to adequately cover the joint input space of the GoldSim model inputs.
2. The matrix connecting the values of the inputs and outputs for each process-level run is provided to the GoldSim model. This keeps the vector of outputs together, never breaking any physical constraints coming from the process model.
3. When a GoldSim model realization is begun and values of inputs are sampled from probability distributions, GoldSim simply finds the "closest" process-level model realization in terms of values of the inputs by computing the distance in the multi-dimensional parameter space using a weighted distance metric (e.g., Euclidian or absolute distances).
4. The outputs associated with that "closest" realization are then imported for use in the GoldSim model for that realization.
5. Results of sensitivity analyses on the process model can be used to inform the weights in the distance metric so that the definition of "closest" also takes into account the relative "importance" of the inputs such that the "closest" uses more information from inputs explaining most variation in the process model outputs.



	InputA	InputB	InputC	Output1	Output2
1	1.1	2.2	3.1	10.3	67.2
2	1.2	2.5	3.2	10.7	55.9
3	1.4	2.7	3.6	11.4	72.1
4	0.9	2.4	4.1	10.1	63.9
5	1.0	2.6	3.9	12.0	64.5
6	1.3	2.7	3.5	11.2	71.0
7	1.2	2.4	3.7	10.5	69.1
8	1.4	2.3	3.6	10.5	59.8
9	1.1	2.5	3.2	11.6	66.2
10	1.0	2.3	3.8	11.3	65.9

Solution: The two outputs are properly constrained because they are chosen from the same realization of the process model.

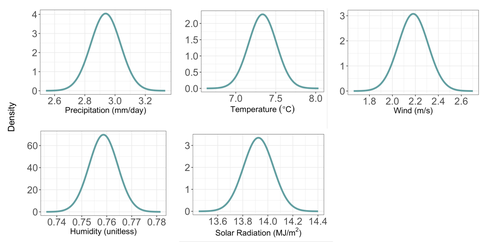
A conceptual diagram of the nearest neighbor approach, with example input and output.

Example: Watershed Modeling for the West Valley Probabilistic Performance Assessment, West Valley, New York

This watershed model is used to parameterize the Probabilistic Performance Assessment (PPA) GoldSim Model for the West Valley Site. It provides stochastic predictions of radionuclide transport rates in waterways, and links to other aspects of the PPA Model such as human exposure and ecological risk.

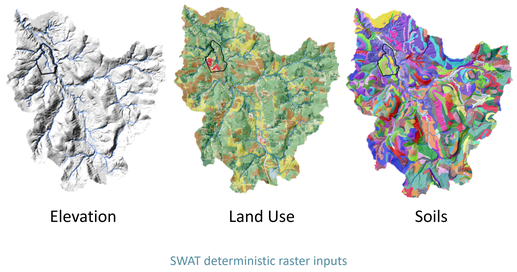
Soil and Water Assessment Tool (SWAT) Model:

Stochastic Inputs:
- Precipitation (1000y average daily total)
- Temperature (1000y daily avg)
- Solar Radiation (1000y daily avg)
- Relative Humidity (1000y daily avg)
- Wind Speed (1000y daily avg)



SWAT Input distributions

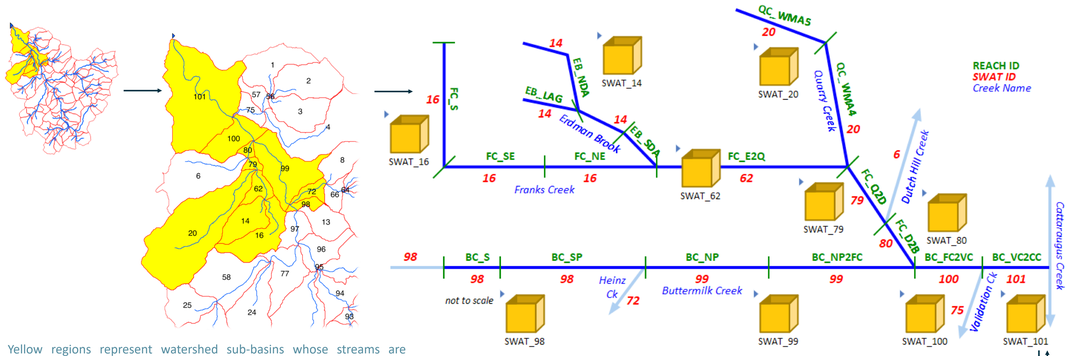
Deterministic Inputs:
- Elevation (Natl Elev Dataset 10 m raster)
- Land Use (NLCD raster)
- Soil (NRCS SSURGO raster)



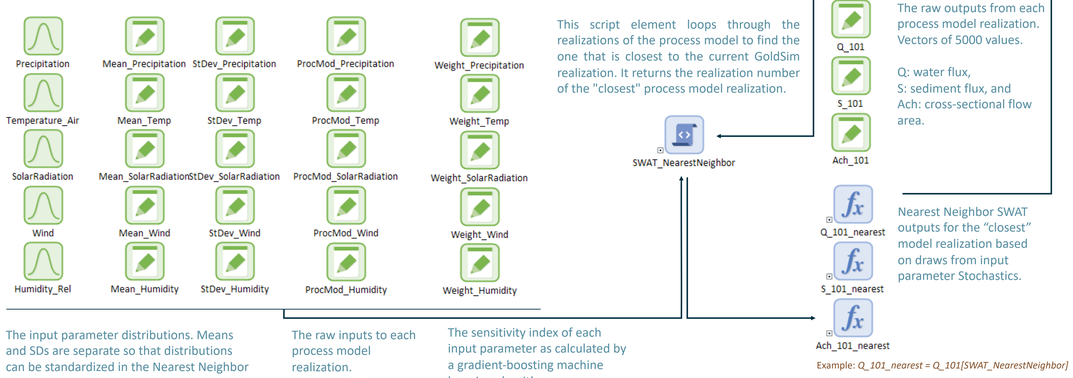
SWAT deterministic raster inputs

Outputs:
- Steady-state **Streamflow** (m^3/s), **Sediment Transport** (Mg/yr), and Cross-Sectional **Area of Flow** (m^2) in a specific subset of reaches.
- A total of 30 outputs, all closely correlated.
- Model Output is a table with columns for the five stochastic input parameters and the steady-state outputs for each reach.

GoldSim Implementation of SWAT Model:



Yellow regions represent watershed sub-basins whose streams are modeled in GoldSim. Reaches are assigned to containers by number.



This script element loops through the realizations of the process model to find the one that is closest to the current GoldSim realization. It returns the realization number of the "closest" process model realization.

The raw outputs from each process model realization. Vectors of 5000 values.

Q: water flux,
S: sediment flux, and
Ach: cross-sectional flow area.

Nearest Neighbor SWAT outputs for the "closest" model realization based on draws from input parameter Stochastics.

The input parameter distributions. Means and SDs are separate so that distributions can be standardized in the Nearest Neighbor script.

The raw inputs to each process model realization. Vectors of 5000 values.

The sensitivity index of each input parameter as calculated by a gradient-boosting machine learning algorithm.

Example: *Q_101_nearest = Q_101[SWAT_NearestNeighbor]*

A Closer Look at the Nearest Neighbor Script Element:

- Loop through each realization of the process model.
- On each iteration of the outer loop, loop through the 5 input variables. This is done by iterating from 1 to the number of process model inputs (a data element). This allows for exclusion of less important variables if desired.
- For each input variable, standardize the GoldSim stochastic draw and the current process model input. Standardization is done by subtracting the mean of the distribution from the draw and dividing by the standard deviation. This removes the units and gives each distance the same scale.
- For the weighted sum of absolute distances method used here, the distance between the GoldSim draws and the SWAT inputs is given by the following equation:

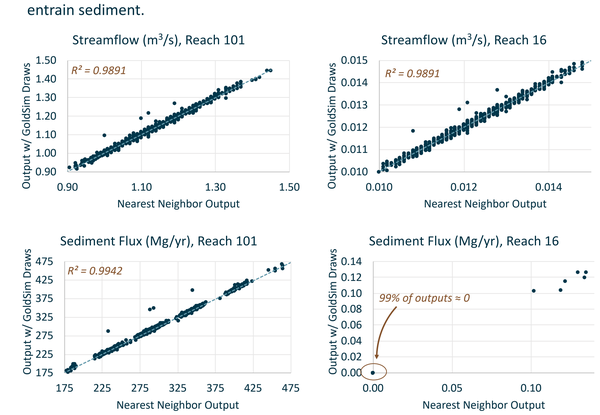
$$d = \frac{1}{n} \cdot \sum_{i=1}^{n} w_i(|x_i - y_i|)$$

where:
d = distance
n = number of inputs
w_i = sensitivity index (weight)
x_i = standardized GS draw
y_i = standardized SWAT input

- The script returns the row number of the closest realization.

```
Statement List
1  Define: distance_squared = 0.0
2  Define: distance = 0
3  Define: distance_addition = 0.0
4  // Vars for info about the GS draw and GS distribution and PM input value, and weights
5  Define: mean = 0.0
6  Define: dev = 0.0
7  Define: weight = 0.0
8  // Vars for the "standardized" values of the GS draw and PM input value
9  Define: draw_standard = 0.0
10 Define: PMinput_standard = 0.0
11 // Vars for keeping track of the "winning" process model realization
12 Define: closest_real = 1
13 Define: closest_distance = 10^30
14 // Loop over realizations of the process model and compute the "distance" from the GoldSim re
15 FOR (real = 1; ~real <=GetRowCount(ProcMod_Precipitation); real = ~real + 1)
16   distance_squared = 0.0
17   // Loop over input variables and add to the distance variable
18   FOR (input_var = 1; ~input_var <= Number_of_ProcMod_Inputs; input_var = ~input_var + 1)
19     IF (~input_var = 1) THEN
20       draw_standard = (Precipitation-Mean_Precipitation)/StDev_Precipitation
21       PMinput_standard = (ProcMod_Precipitation[~real]-Mean_Precipitation)/(StDev_Precipitati
22       weight = Weight_Precipitation
23     END IF
24     IF (~input_var = 2) THEN
25       draw_standard = (Wind-Mean_Wind)/StDev_Wind
26       PMinput_standard = (ProcMod_Wind[~real]-Mean_Wind)/(StDev_Wind)
27       weight = Weight_Wind
28     END IF
29     IF (~input_var = 3) THEN
30       draw_standard = (Humidity_Rel-Mean_Humidity)/StDev_Humidity
31       PMinput_standard = (ProcMod_Humidity[~real]-Mean_Humidity)/(StDev_Humidity)
32       weight = Weight_Humidity
33     END IF
34     IF (~input_var = 4) THEN
35       draw_standard = (Temperature_Air-Mean_Temp)/StDev_Temp
36       PMinput_standard = (ProcMod_Temp[~real]-Mean_Temp)/(StDev_Temp)
37       weight = Weight_Temp
38     END IF
39     IF (~input_var = 5) THEN
40       draw_standard = (SolarRadiation-Mean_SolarRadiation)/StDev_SolarRadiation
41       PMinput_standard = (ProcMod_SolarRadiation[~real]-Mean_SolarRadiation)/(StDev_Solar
42       weight = Weight_SolarRadiation
43     END IF
44     // Now compute the 1-D distance for this input. GoldSim's sampled value is "draw".
45     // Process model input value is PMinput
46     // MODIFED FOR SUM OF ABSOLUTE DISTANCE METRIC
47     distance_addition = ~weight*abs(~draw_standard-~PMinput_standard)
48     distance_squared = ~distance_squared + ~distance_addition
49   END FOR
50   // MODIFED FOR S.A.D METRIC --- NO SQUARE ROOT TAKEN
51   distance = ~distance_squared/Number_of_ProcMod_Inputs
52   IF (~distance < ~closest_distance) THEN
53     closest_distance = ~distance
54     closest_real = ~real
55   END IF
56 END FOR
57 Result = ~closest_real
```

How Well Does It Work?

To test how well the nearest neighbor selection method works, we repeat the process model using the values of input parameters drawn in GoldSim. By comparing the outputs of each of these process model runs to the chosen nearest neighbor realization, we can assess how accurately this method approximates the process model outcomes, and how we might improve exploration of the parameter space in the initial runs of the process model and sensitivity analysis.

In the figures below, 1000 GoldSim draws are used to force the process model, and the output is compared to the original nearest neighbor values. These figures demonstrate that for both the outlet of the watershed (reach 101), and a headwater stream (reach 16), **the nearest neighbor approach provides a satisfactory estimate of both streamflow and sediment flux.** Note that for reach 16, the majority of sediment flux values are near zero. This is caused by streamflow values which are too low to entrain sediment.