SIBYL manual

Version: July-2022

Simulation code powered by lattice dose-response functions

SATOH Daiki
Japan Atomic Energy Agency
satoh.daiki@jaea.go.jp

References:

D. Satoh, H. Nakayama, T. Furuta, T. Yoshihiro, K. Sakamoto, "Simulation code for estimating external gamma-ray doses from a radioactive plume and contaminated ground using a local-scale atmospheric dispersion model", PLOS ONE, 2021.
https://doi.org/10.1371/journal.pone.0245932

H. Nakayama, N. Onodera, D. Satoh, H. Nagai, Y. Hasegawa, Y. Idomura, "Development of local-scale high-resolution atmospheric dispersion and dose assessment system", Journal of Nuclear Science and Technology, 2021.
https://doi.org/10.1080/00223131.2022.2038302

1. Structure of SIBYL-distribution package

Figure 1 illustrates the directory-and-file structure of the SIBYL-distribution package. "SIBYL-manual.pdf" indicates this document. "journal.pone.0245932.pdf" is the original paper of the SIBYL code published by PLOS ONE (https://doi.org/10.1371/journal.pone.0245932). The directory named "GPM" contains a computer program to calculate a distribution of radioactivity concentrations based on the Gaussian plume model for a SIBYL simulation. The detail is found in "GPM_for_SIBYL-manual.pdf". The SIBYL code is released under the MIT license.

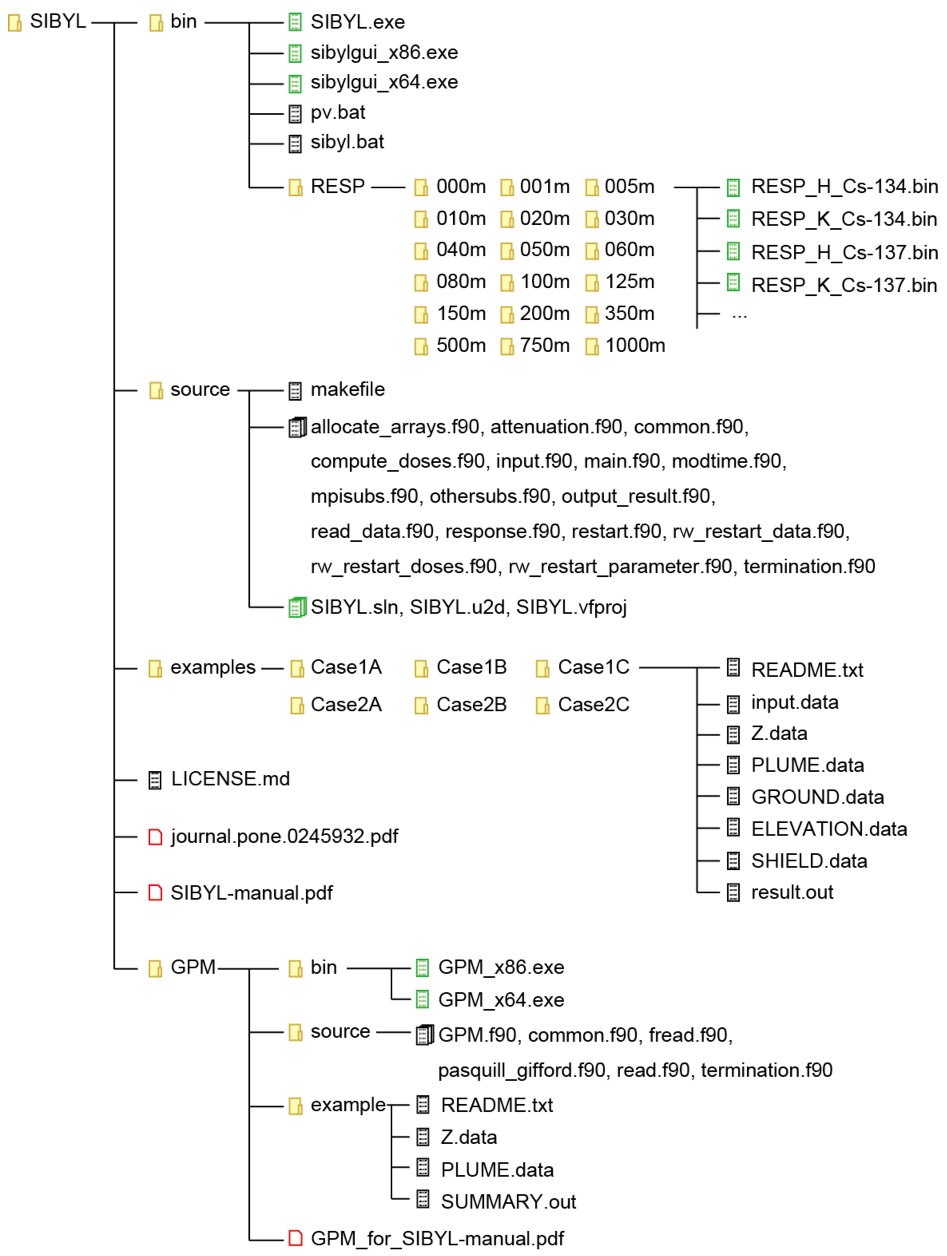


Figure 1: Directory-and-file structure of a SIBYL Package.

1.1. “bin” directory

Table 1 lists the contents of the “bin” directory.

Table 1: Files and directories contained in the “bin” directory.

Name	**Explanation**
SIBYL.exe	Executable file of SIBYL for Windows platform with OpenMP technology. The executable was compiled by Intel Fortran compiler under x86-64 environment on Windows 10.
sbylgui_x86.exe	Windows software (x86) to provide a GUI for the SIBYL code. See Section 3.2 for details.
sbylgui_x64.exe	Windows software (x86-64) to provide a GUI for the SIBYL code. See Section 3.2 for details.
pv.bat	Windows batch file to give a PATH to the ParaView software to visualize the input and output data through the GUI software.
sibyl.bat	Windows batch file to give a PATH to the SIBYL’s executable file which is used on the GUI software.
RESP	Directory containing binary files of dose-response functions. See Section 1.1.1 for details.

1.1.1. “RESP”

The data of dose-response functions are contained in the directory named “RESP”, and the “RESP” should be put at the same directory containing the SIBYL’s executable file. Those data are prepared for the radionuclides of ^{134}Cs, ^{136}Cs, ^{137}Cs, ^{131}I, ^{132}I, ^{133}I, ^{85}Kr, and ^{132}Te at altitudes of 1, 5, 10, 20, 30, 40, 50, 60, 80, 100, 125, 150, 200, 350, 500, 750, and 1000 m. The files having “H” in their names mean the data for calculating ambient dose equivalent, and the files with “K” are for air kerma free in air. **Table 2** lists the response-function files used in the current SIBYL code.

Table 2: List of the response-function data.

File name	**Explanation**
RESP_H_Cs-134.bin	Response functions of ^{134}Cs to ambient dose equivalent at the height of 1 m above ground.
RESP_H_Cs-136.bin	Response functions of ^{136}Cs to ambient dose equivalent at the height of 1 m above ground.

RESP_H_Cs-137.bin	Response functions of ^{137}Cs to ambient dose equivalent at the height of 1 m above ground.
RESP_H_I-131.bin	Response functions of ^{131}I to ambient dose equivalent at the height of 1 m above ground.
RESP_H_I-132.bin	Response functions of ^{132}I to ambient dose equivalent at the height of 1 m above ground.
RESP_H_I-133.bin	Response functions of ^{133}I to ambient dose equivalent at the height of 1 m above ground.
RESP_H_Kr-85.bin	Response functions of ^{85}Kr to ambient dose equivalent at the height of 1 m above ground. Note that the ^{85}Kr is noble gas, therefore, there is no data in the directory '000m' corresponding to ground contamination.
RESP_H_Te-132.bin	Response functions of ^{132}Te to ambient dose equivalent at the height of 1 m above ground.
RESP_K_Cs-134.bin	Response functions of ^{134}Cs to air kerma free in air at the height of 1 m above ground.
RESP_K_Cs-136.bin	Response functions of ^{136}Cs to air kerma free in air at the height of 1 m above ground.
RESP_K_Cs-137.bin	Response functions of ^{137}Cs to air kerma free in air at the height of 1 m above ground.
RESP_K_I-131.bin	Response functions of ^{131}I to air kerma free in air at the height of 1 m above ground.
RESP_K_I-132.bin	Response functions of ^{132}I to air kerma free in air at the height of 1 m above ground.
RESP_K_I-133.bin	Response functions of ^{133}I to air kerma free in air at the height of 1 m above ground.
RESP_K_Kr-85.bin	Response functions of ^{85}Kr to air kerma free in air at the height of 1 m above ground. Note that the ^{85}Kr is noble gas, therefore, there is no data in the folder '000m' corresponding to ground contamination.
RESP_K_Te-132.bin	Response functions of ^{132}Te to air kerma free in air at the height of 1 m above ground.

1.2. “source” directory

Table 3 lists the contents of the “source” directory.

Table 3: Files in the “source” directory.

Name	**Explanation**
makefile	Makefile run on *bash*. The script assumes the Intel Fortran compiler for Linux as a default compiler. The users can change the compiler and parallel computing environment with MPI and OpenMP by modifying this script.
*.f90	Fortran-95 source files of the SIBYL code.
SIBYL.sln, SIBYL.u2d, SIBYL.vfproj	Project files of the Intel Fortran compiler for Windows run on Microsoft Visual Studio.

1.3. “examples” directory

This directory contains examples of input and output data discussed in the SIBYL’s original paper (https://doi.org/10.1371/journal.pone.0245932).

2. Input and output files

Table 4 lists input and output files of SIBYL. As input files, “input.data”, “Z.data”, and “PLUME.data” or “GROUND.data” are mandatory, and “ELEVATION.data” and “SHIELD.data” are optional. When the users execute the SIBYL code without the GUI software, those input files have to be put at the same directory containing the executable file “SIBYL.exe”. When the users use the GUI software, the software automatically copies those files to the executable directory.

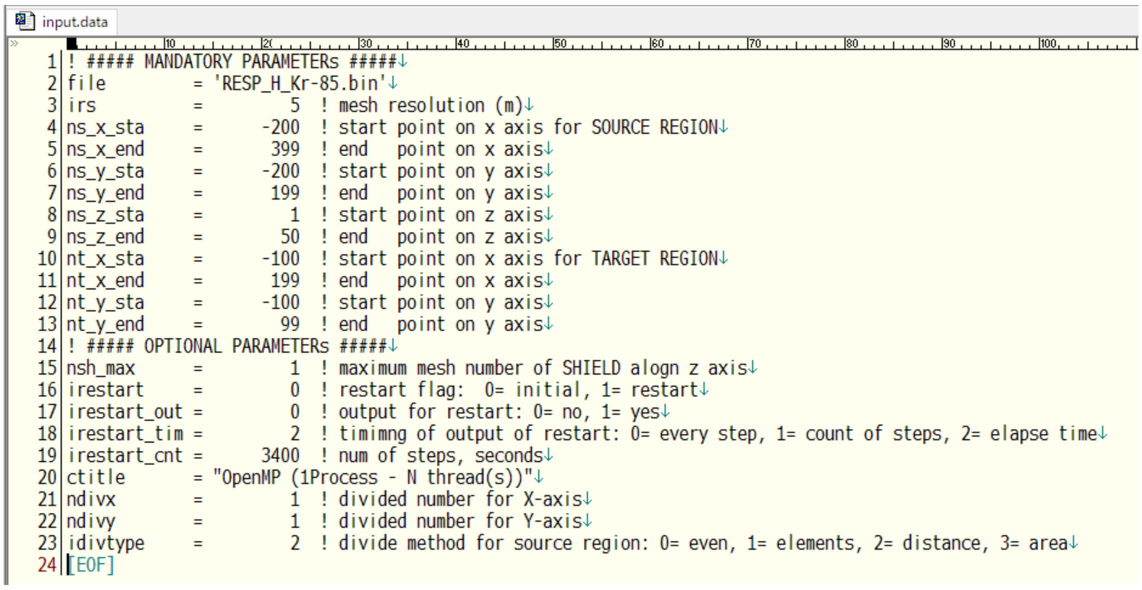
Table 4: Input and output files of SIBYL.

Name	**Explanation**	**Data dimension**
input.data	(mandatory) Input file to provide parameters for controlling a SIBYL simulation.	The parameters are set with *keyword* = *value*. See Section 2.1 for details.
Z.data	(mandatory) Input file to give heights of z-cells (m).	(1D) [z] = [1 : *ns_z_end*]
PLUME.data	(mandatory if “GROUND.data” does not exist) Input file to give activity concentration at cells inside radioactive plume (Bq/m^3).	(3D) [x, y, z] = [*ns_x_sta* : *ns_x_end*, *ns_y_sta* : *ns_y_end*, *ns_z_sta* : *ns_z_end*]

GROUND.data	(mandatory if "PLUME.data" does not exist) Input file to give activity concentration at cells on contaminated ground (Bq/m^2).	(2D) [x, y] = [*ns_x_sta* : *ns_x_end*, *ns_y_sta* : *ns_y_end*]
ELEVATION.data	(optional) Input file to give terrain elevation data (m).	(2D) [x, y] = [*nt_x_sta* : *nt_x_end*, *nt_y_sta* : *nt_y_end*]
SHIELD.data	(optional) Input file to set obstacles. The numerical value of "1" in the file means that the cell is filled with air. The number of "9999" indicates that the cell is occupied by obstacle whose density and total attenuation coefficient for photon are given by the input parameters of "den" and "att" in "input.data".	(3D) [x, y, z] = [*ns_x_sta* : *ns_x_end*, *ns_y_sta* : *ns_y_end*, *1* : *nsh_max*]
result.out	Output file to give a dose-distribution calculated by SIBYL (μSv/h for H*(10) or μGy/h for K_{air}).	(2D) [x, y] = [*nc_x_sta* : *nc_x_end*, *nc_y_sta* : *nc_y_end*]

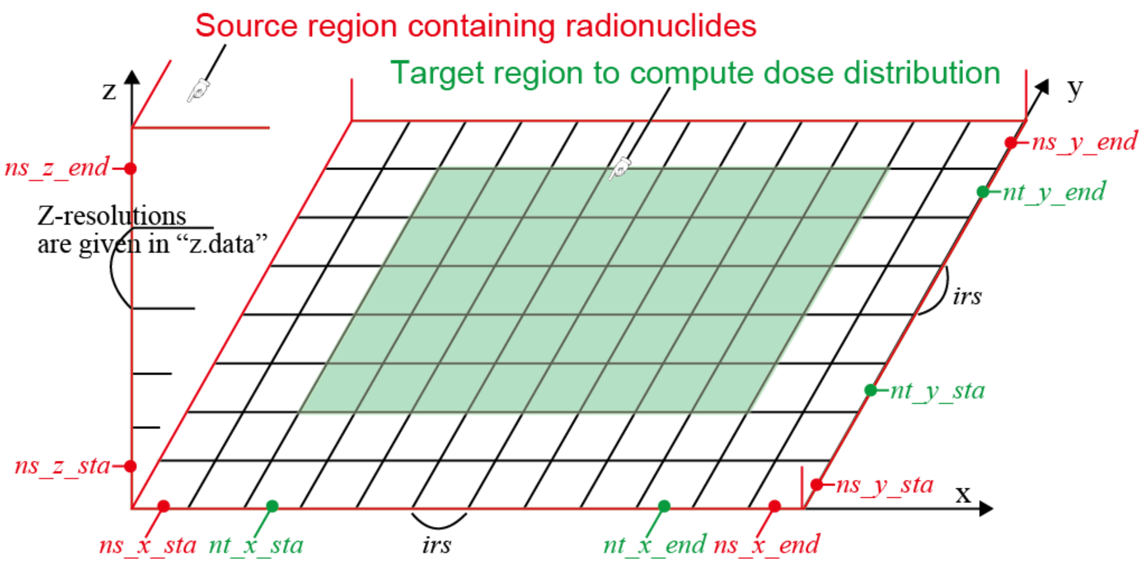
2.1. Input parameters

The input parameters for a SIBYL simulation are described in "input.data". **Figure 2** shows an example of "input.data". The users can insert their comments by putting the character "!" before them. The schematic of the simulation geometry configurated with input parameters is drawn in **Figure 3**. The SIBYL code requires the mandatory parameters listed in **Table 5** for the simulation. In addition, the code utilizes the optional parameters listed in **Table 6** to configure the computational conditions and algorithm.



```
! ##### MANDATORY PARAMETERs #####
file          = 'RESP_H_Kr-85.bin'
irs           =        5  ! mesh resolution (m)
ns_x_sta      =     -200  ! start point on x axis for SOURCE REGION
ns_x_end      =      399  ! end   point on x axis
ns_y_sta      =     -200  ! start point on y axis
ns_y_end      =      199  ! end   point on y axis
ns_z_sta      =        1  ! start point on z axis
ns_z_end      =       50  ! end   point on z axis
nt_x_sta      =     -100  ! start point on x axis for TARGET REGION
nt_x_end      =      199  ! end   point on x axis
nt_y_sta      =     -100  ! start point on y axis
nt_y_end      =       99  ! end   point on y axis
! ##### OPTIONAL PARAMETERs #####
nsh_max       =        1  ! maximum mesh number of SHIELD alogn z axis
irestart      =        0  ! restart flag:  0= initial, 1= restart
irestart_out  =        0  ! output for restart: 0= no, 1= yes
irestart_tim  =        2  ! timimng of output of restart: 0= every step, 1= count of steps, 2= elapse time
irestart_cnt  =     3400  ! num of steps, seconds
ctitle        = "OpenMP (1Process - N thread(s))"
ndivx         =        1  ! divided number for X-axis
ndivy         =        1  ! divided number for Y-axis
idivtype      =        2  ! divide method for source region: 0= even, 1= elements, 2= distance, 3= area
[EOF]
```

Figure 2: Example of "input.data".



Source region containing radionuclides
Target region to compute dose distribution
z
y
x
ns_z_end
Z-resolutions are given in "z.data"
ns_z_sta
ns_y_end
nt_y_end
irs
nt_y_sta
ns_y_sta
ns_x_sta
nt_x_sta
irs
nt_x_end
ns_x_end

Figure 3: Schematic of a simulation geometry set by input parameters.

Table 5: Mandatory input parameters.

Keyword	Example	Explanation
file	RES_H_Kr-85.bin	File name of the response-function data used in the simulation. See also **Table 2**.
irs	5	(integer) Horizontal (x-y plane) resolution of the computational cell (m).

ns_x_sta	-200	(integer) Start-cell number of a source region on x axis.
ns_x_end	399	(integer) End-cell number of a source region on x axis.
ns_y_sta	-200	(integer) Start-cell number of a source region on y axis.
ns_y_end	199	(integer) End-cell number of a source region on y axis.
ns_z_sta	1	(integer) Start-cell number of a source region on z axis. The cell resolutions along the z axis are given with numerical data in "z.data". "ns_z_sta = 0" means a ground contamination that the radionuclides put on the surface of ground.
ns_z_end	50	(integer) End-cell number of a source region on z axis.
nt_x_sta	-100	(integer) Start-cell number of a target region on x axis. The dose distribution is calculated for this region, and the results of the doses at cells in the target region are outputted to "result.out".
nt_x_end	199	(integer) End-cell number of a target region on x axis.
nt_y_sta	-100	(integer) Start-cell number of a target region on y axis.
nt_y_end	99	(integer) End-cell number of a target region on y axis.

Table 2: Optional input parameters.

Keyword	**Default value**	**Explanation**
imode	0	(integer) Calculation mode. If the users set this parameter to "1", SIBYL calculates the dose distribution only for ground contamination. If the value is set to "2", the dose distribution from a plume is calculated. If the value is "0", the code calculates the doses from ground and

		plume.
att	8.118E-2	(float) Total attenuation coefficient of obstacles (cm^2/g).
den	2.400E-1	(float) Effective density of obstacles (g/cm^3).
ctitle	'SIBYL…'	Title of the simulation for echo back in output file.
nc_x_sta	nt_x_sta	(integer) Start-cell number to narrow the range of dose-calculation region on x axis. Basically, the dose calculation region is corresponding to the target region, and the data of obstacles are given to this region using a file "SHIELD.data". If the users want to narrow the range of the calculation region without modification of "SHIELD.data" as a test calculation, SIBYL can do that with these parameters.
nc_x_end	nt_x_end	(integer) End-cell number to narrow the range of dose-calculation region on x axis.
nc_y_sta	nt_y_sta	(integer) Start-cell number to narrow the range of dose-calculation region on y axis.
nc_y_end	nt_y_end	(integer) End-cell number to narrow the range of dose-calculation region on y axis.
nsh_max	ns_z_end	(integer) Maximum number of cells containing obstacles along z axis. SIBYL can skip a search for obstacles in the region above the cell designated with this parameter. It would make the computational burden light.
cdirname	'RESTART'	Directory name for storing the data used in a re-start calculation.
irestart	0	(integer) Parameter for restart calculation. If the value is set to 1, SIBYL restarts the calculation using the data stored in the restart directory. If the value is set to 0, SIBYL newly calculates the dose distribution.

irestart_out	1	(integer) Parameter to output restart data. If the value is set to 1, SIBYL outputs the restart data at fixed timing. If the value is set to 0, SIBYL does not output any restart data.
irestart_tim	1	(integer) Parameter for output timing of restart data. The data are outputted with 0: every *TGT_PY* step, 1: *TGT_PY* steps indicated with *irestart_cnt*, and 2: elapsed time indicated with *irestart_cnt*.
irestart_cnt	1	(integer) Parameter for output timing of restart data. If *irestart_tim* is set to 1, this parameter means the number of *TGT_PY* lines. If *irestart_tim* is set to 2, this parameter means elapsed time in sec.
ndivx	1	(integer) Number of divisions for *LOOP_TGT_PX* for MPI computing.
ndiby	1	(integer) Number of divisions for *LOOP_TGT_PY* for MPI computing.
idivtype	1	(integer) Parameter to choose the MPI algorithm to divide the MPI process. 0: simple even dividing, 1: elements count of source region, 2: distance from center of target region to source region, 3: overlap area of source region and target region.

2.2. Output data

The output file named “result.out” is generated when the SIBYL simulation finished normally.

3. How to execute the SIBYL code

In the “bin” directory of the SIBYL-distribution package, the users can find an executable file of SIBYL named “SIBYL.exe” for Windows platform with OpenMP technology. If the users want to compile the code by themselves, the project files of the Intel Fortran compiler on Microsoft Visual Studio are prepared for Windows platform in the “source” directory. In addition, if the users want to compile that on Linux, a makefile for bash is also provided in that directory.

When the users use the OpenMP technology, the environmental valuable "OMP_NUM_THREADS" has to be set to designate the number of computing threads used in the simulation. As an example, the command to set that valuable to "12" for bash on Linux is given as follows:

```
$ export OMP_NUM_THREADS=12
```

When the users execute SIBYL with the MPI technology, they have to compile the code with the MPI flag using "makefile" and run the MPI command as follows:

```
$ mpirun -np 12 SIBYL.exe
```

3.1. Execution of SIBYL without GUI software

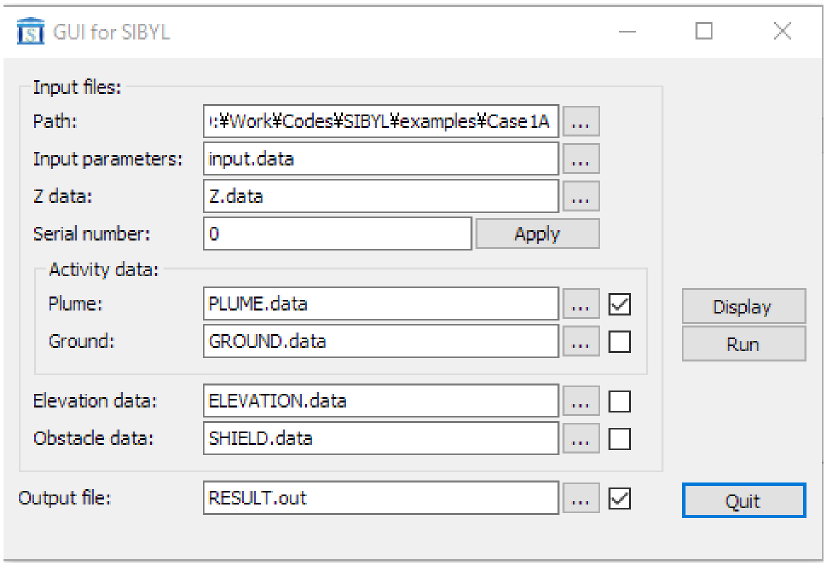
The users have to put the executable file of SIBYL, the response directory "RESP", and the input files at the same directory. Then, they enter the executable file name on a terminal or double click the file on a file browser. The users obtain the output file "result.out" at that directory after the calculation.

3.2. Execution with GUI software on Windows

The SIBYL distribution package includes the GUI software for Windows to launch SIBYL and visualize the data using the ParaView software (https://www.paraview.org/).

The executable file of SIBYL, the response directory "RESP", the GUI software "sibylgui_x64.exe" or "sibylgui_x86.exe", and the configuration files "pv.bat" and "sibyl.bat" have to be put at the same directory.

Figure 4 shows a main window of the GUI software. The path to a directory including input files is set at <*Path*>. The value can be pasted directly into the blank. <*Input parameters*> and <*Z data*> are chosen graphically by clicking "…" buttons next to the blanks. When <*Apply*> button is clicked with <*Serial number*> = 0, the default names are given for <*Plume*>, <*Ground*>, <*Elevation data*>, <*Obstacle data*>, and <*Output file*>. If a positive integer is given to <*Serial number*>, the serial number is added to those file names. In addition, the users can set arbitrary names to those files by modifying the names on GUI directly.


GUI for SIBYL

Input files:

Path:	:¥Work¥Codes¥SIBYL¥examples¥Case1A	
Input parameters:	input.data	
Z data:	Z.data	
Serial number:	0	

Activity data:

Plume:	PLUME.data	✓
Ground:	GROUND.data	☐
Elevation data:	ELEVATION.data	☐
Obstacle data:	SHIELD.data	☐
Output file:	RESULT.out	✓

Figure 4: Main window of the SIBYL-GUI software.

When checkboxes for the files are checked and <*Display*> button is clicked, the ParaView is launched and the data are visualized on it. **Figure 5** shows an example of visualization using ParaView for "Case2A" in the "example" directory.

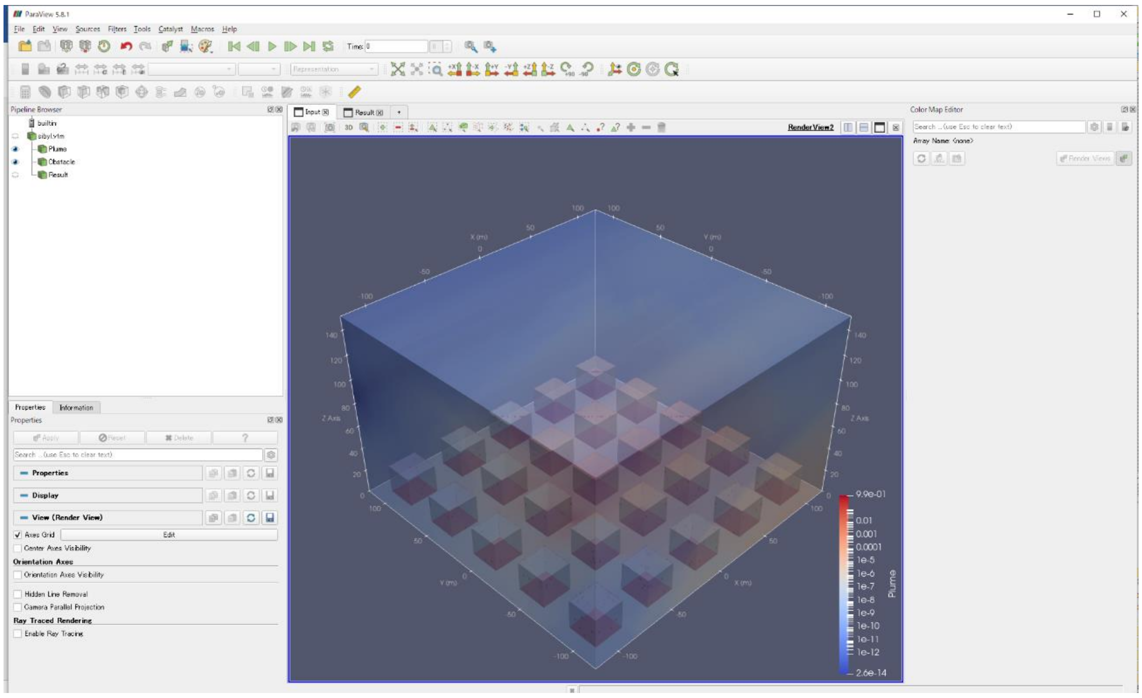


Figure 5: An example of visualization on ParaView for "Case2A" input data.

By clicking <*Run*> button, the input files which are checked in the checkboxes are copied to the current directory, and then the SIBYL simulation is started. After the simulation, an output file designated in <*Output file*> on the GUI window is generated in the current directory. The result is also visualized on ParaView by checking the checkbox for the output file and clicking <*Display*> button. **Figure 6** shows an example of visualization for a computed result. The ParaView window has two tabs, namely “Input” and “Result”, and the “Input” tab is focused in the default action of the SIBYL’s GUI software. Therefore, the users have to change the tab from “Input” to “Result” to see the output data. The users can analyze those data on ParaView using various functions supplied by ParaView. See “ParaView User’s Guide” (https://docs.paraview.org/en/latest/) for details.

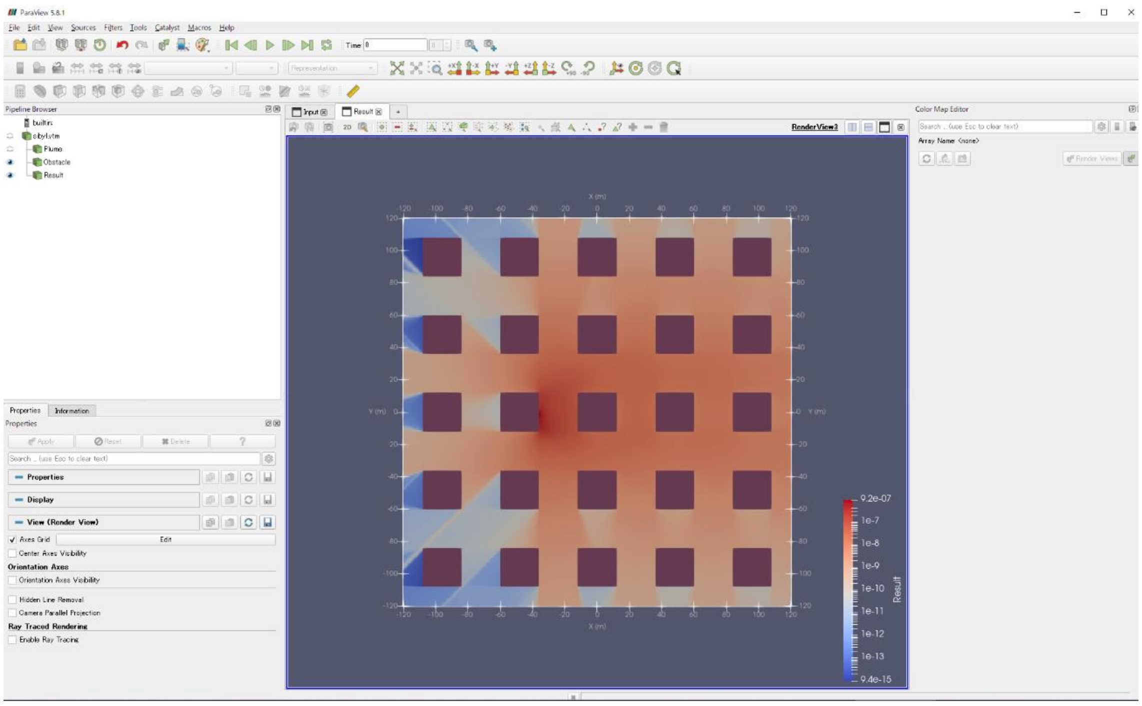


Figure 6: An example of visualization on ParaView for “Case2A” output data.