

Spatially Explicit Integrated Modeling System (SEIMS)

User Manual

Version alpha-0.2.1

Liang-Jun Zhu, Junzhi Liu, Cheng-Zhi Qin, A-Xing Zhu
09/30/2019

**State Key Lab of Resources and Environmental Information System,
Institute of Geographic Sciences and Natural Resources Research,
China Academy of Science, Beijing, 100101, China**

**Key Laboratory of Virtual Geographic Environment (Nanjing Normal University),
Ministry of Education, Nanjing 210023, China**

Department of Geography, University of Wisconsin-Madison, Madison, WI 53706, USA

This page is intentionally left blank.

Table of Contents

Table of Contents	i
List of Figures	iii
List of Tables	vi
Section 1. Introduction.....	7
1:1. What is SEIMS?.....	8
1:1.1 Key procedures	8
1:1.2 Publications	9
1:1.3 Support	9
1:2. Why SEIMS?	11
1:3. Terminology	12
1:4. References	13
Section 2. Get started.....	18
2:1. Download and installation	19
2:1.1 Download.....	19
2:1.2 Prerequisite software and libraries.....	20
2:1.3 Test of the C/C++ building environment	29
2:1.4 Installation of SEIMS	30
2:2. Data preparation of demo watershed.....	32
2:2.1 Basic conventions of file formats	32
2:2.2 Spatial data.....	33
2:2.3 Precipitation data.....	35
2:2.4 Meteorological data	36
2:2.5 Observed data	37
2:2.6 Lookup tables	39
2:2.7 Management practices data	41
2:3. Data preprocessing for watershed modeling	49
2:3.1 Simple usage	49
2:3.2 Configuration file of data preprocessing	51
2:3.3 Advanced usage	54
2:3.4 Intermediate data of preprocessing	55
2:3.5 Structure of the watershed modeling database	55
2:4. Running a SEIMS-based watershed model	57
2:4.1 Simple usage	57
2:4.2 File structure of a SEIMS-based watershed model.....	58
2:4.3 Advanced usage	62
2:5. Postprocessing	71

2:5.1	Simple usage	71
2:5.2	Configuration file of postprocessing.....	72
2:5.3	Advanced usage	74
2:6.	Parameter sensitivity analysis.....	76
2:6.1	Simple usage	76
2:6.2	Configuration file of parameter sensitivity analysis	79
2:6.3	Advanced usage	80
2:7.	Auto-Calibration.....	81
2:7.1	Simple usage	81
2:7.2	Configuration file of auto-calibration	84
2:7.3	Advanced usage	85
2:8.	BMP scenarios analysis	86
2:8.1	Simple usage	86
2:8.2	Configuration file of scenario analysis	89
2:8.3	Advanced usage	92
Section 3.	Design and implementation.....	95
3:1.	Overall design of SEIMS.....	96
Section 4.	Write your own code	99
4:1.	Develop a new module of one watershed process	100
4:1.1	Review literature and/or existing model code	100
4:1.2	Create new SEIMS module	101
4:1.3	Open SEIMS solution in Visual Studio and start to code	103
4:1.4	Test the new SEIMS module	106

List of Figures

Figure 2:1-1 Clone the single master branch of SEIMS repository using Git Bash command	20
Figure 2:1-2 Directory tree of SEIMS source code	20
Figure 2:1-3 Screenshot of Robo 3T – a user-friendly and efficient MongoDB GUI	24
Figure 2:1-4 Directory tree of the compiled GDAL library	25
Figure 2:1-5 Building Visual Studio solution of CCGL library by CMake	29
Figure 2:1-6 Unit test results of CCGL (Common Cross-platform Geographic-computing Library)	30
Figure 2:1-7 Directory tree of C++ applications and Python utility tools of SEIMS	31
Figure 2:1-8 Usage of the OpenMP version of SEIMS main program	31
Figure 2:2-1 Location of the demo watershed named Youwuzhen watershed	33
Figure 2:2-2 Landuse map of the Youwuzhen watershed	34
Figure 2:2-3 Soil map of the Youwuzhen watershed	34
Figure 2:3-1 Runtime logs of data preprocessing for watershed modeling of the Youwuzhen watershed	49
Figure 2:3-2 Directory tree of intermediate data of the Youwuzhen watershed after data preprocessing	50
Figure 2:3-3 Screenshot of the watershed modeling databases of the Youwuzhen watershed	50
Figure 2:3-4 Configuration content for data preprocessing of the Youwuzhen watershed	52
Figure 2:4-1 Simple usage of the OpenMP version of the Youwuzhen watershed model	57
Figure 2:4-2 Predefined outputs of the Youwuzhen watershed model	58
Figure 2:4-3 SEIMS modules involved in the demo Youwuzhen watershed model	61
Figure 2:4-4 Usage of the MPI&OpenMP version of SEIMS main program	64
Figure 2:4-5 Runtime logs of the MPI&OpenMP version of the Youwuzhen watershed model	65
Figure 2:4-6 Run the MPI&OpenMP version of the Youwuzhen watershed model on a Linux cluster	66
Figure 2:4-7 Configuration content for model execution of the Youwuzhen watershed model	68
Figure 2:4-8 New Youwuzhen watershed model after the simple customization, i.e., replacing the Penman-Monteith method to Priestley-Taylor method for the simulation of potential evapotranspiration	69
Figure 2:4-9 Spatial distributions of average potential evapotranspiration simulated by the Youwuzhen watershed model using the (a) Penman-Monteith method and the (b) Priestley-Taylor method	70

Figure 2:4-10 Differences of the streamflow (Q, m ³ /s) at the outlet of Youwuzhen watershed derived from the Youwuzhen watershed models using the Penman-Monteith method (PET_PM) and Priestley-Taylor method (PET_PT) for the simulation of potential evapotranspiration, respectively.....	70
Figure 2:5-1 Runtime logs of postprocessing of the Youwuzhen watershed model	71
Figure 2:5-2 Hydrograph of streamflow (Q, m ³ /s) at the watershed outlet derived from the Youwuzhen watershed model with default model parameters	72
Figure 2:5-3 Configuration content for postprocessing of the Youwuzhen watershed model	74
Figure 2:5-4 Calibration and validation of the simulated streamflow (Q, m ³ /s) at the outlet of the Youwuzhen watershed derived from the demo Youwuzhen watershed model with default model parameters using Chinese font.....	75
Figure 2:5-5 Modification of configuration content of postprocessing to plot both calibration and validation period using Chinese font	75
Figure 2:6-1 Beginning (a) and ending (b) runtime logs of the predefined parameter sensitivity analysis of the Youwuzhen watershed model.....	77
Figure 2:6-2 Sampled calibration values of selected parameters for the sensitivity analysis of the Youwuzhen watershed model	78
Figure 2:6-3 Parameter sensitivity analysis result with respect to the NSE of the simulated streamflow (m ³ s ⁻¹) at the watershed outlet.....	78
Figure 2:6-4 Configuration content for parameter sensitivity analysis of the Youwuzhen watershed model.....	80
Figure 2:7-1 Runtime logs of auto-calibration of the Youwuzhen watershed model.....	82
Figure 2:7-2 Calibration values of selected parameters generated during the optimization of auto-calibration of the Youwuzhen watershed model.....	82
Figure 2:7-3 Near optimal Pareto solutions of the 1 st (a) and 2 nd (b) generations with the objectives of maximize the NSE and minimize the RSR of the simulated streamflow at the watershed outlet.....	83
Figure 2:7-4 Auto-calibration result of streamflow (Q, m ³ /s) simulation of the 2 nd generation at the calibration and validation periods	84
Figure 2:7-5 Configuration content for auto-calibration of the Youwuzhen watershed model based on NSGA-II	85
Figure 2:8-1 Runtime logs of BMP scenarios analysis of the Youwuzhen watershed model	87
Figure 2:8-2 BMP scenarios generated during the spatial optimization of BMP scenarios	87
Figure 2:8-3 Near optimal Pareto solutions of the 1 st (a) and 2 nd (b) generations with the objectives of maximizing the environmental effectiveness and minimizing the economic net-cost.....	89
Figure 2:8-4 Spatial distribution of the selected BMP scenario of the 2 nd generation of the BMP scenarios optimization.....	89

Figure 2:8-5 Configuration content for BMP scenario analysis of the Youwuzhen watershed model based on NSGA-II.....	92
Figure 3:1-1 Architecture of Spatially Explicit Integrated Modeling System (SEIMS) which consists of the SEIMS module library, two versions of SEIMS main programs (i.e., OpenMP version and OpenMP&MPI version), the watershed database, and utility tools for watershed model applications, and can be conducted on multiple parallel computing platforms	97
Figure 4:1-1 Test the Hargreaves method for the simulation of potential evapotranspiration based on the Youwuzhen watershed model customized in Section 2:4.3.3.....	107
Figure 4:1-2 Spatial distributions of average potential evapotranspiration simulated by the Youwuzhen watershed model using the (a) Priestley-Taylor method and the (b) Hargreaves method	108

List of Tables

Table 2:2-1 Fields of precipitation station.....	35
Table 2:2-2 Fields and formats of precipitation data item.....	35
Table 2:2-3 Fields and formats of meteorological data item.....	36
Table 2:2-4 Fields of observed data.....	37
Table 2:2-5 Fields and formats of observed data item	38
Table 2:2-6 Fields and description in lookup table of soil properties	39
Table 2:2-7 Fields and descriptions in the lookup table of initial landcover parameters	40
Table 2:2-8 Fields and descriptions of BMP scenario table.....	42
Table 2:2-9 Fields and descriptions of plant management practices.....	44
Table 2:2-10 Parameters defined for plant management operations	46
Table 2:2-11 Fields and descriptions of general areal structural management practices	47
Table 2:4-1 Available fields of <code>file.out</code> configuration file.....	59

Section 1. Introduction

This section is a brief introduction of the SEIMS (short for Spatially Explicit Integrated Modeling System). In the current pre-released version, this section is not yet finished. Please refer to Zhu et al. (2019a) for more information.

1:1. What is SEIMS?

SEIMS (Spatially Explicit Integrated Modeling System) is an integrated, modular, parallelized, fully-distributed watershed modeling system, which is also designed for scenario analysis, especially for evaluating and optimizing BMP scenarios.

SEIMS can be used to build a specific watershed model of a single watershed for long-term or storm-event simulation. Multiple watershed processes could be considered, e.g., hydrological processes, soil erosion, nutrient cycling, and plant growth.

- SEIMS is a fully-distributed watershed modeling system, in which grid cells with hydrological connection are used as basic simulation units within each subbasin. Support for using irregularly shaped fields, such as hydrologic response units (Arnold *et al.*, 1998) and patches (Tague and Band, 2004), as basic simulation units is under development.
- SEIMS is a multiple watershed processes integrated modular system, to which model developers could easily add their own modules (algorithms) of hydrology, soil erosion, nutrient cycling, plant growth, and BMP management, etc.
- SEIMS is a parallelized modeling system, which takes fully use of the parallelizability at both the subbasin level and the basic simulation unit level (e.g., grid cell) simultaneously. Developers can easily implement parallelized watershed model in a nearly serial programming manner.
- SEIMS is a configurable watershed modeling system, in which users can easily specify their modules, and outputs.
- SEIMS is developed by C++ with the support of GDAL, mongo-c-driver, OpenMP and MPI libraries. Python is used to organize various workflows as utility tools, e.g., preprocessing, post-processing, parameter sensitivity analysis, auto-calibration, and scenario analysis. SEIMS can be compiled by commonly used C++ compiler (e.g. MSVC 2010+, GCC4.6+, Intel C++ 12+, and Clang 8.0+) and run on multiple parallel computing platforms (e.g., multi-core computer, and symmetric multiprocessing (SMP) clusters).
- SEIMS, which uses several open source technologies (e.g., GDAL, mongo-c-driver), is also open-sourced at Github (<https://github.com/lreis2415/SEIMS>).

1:1.1 Key procedures

The following points are commonly used procedures to build a watershed model and perform scenario analysis based on SEIMS. The details of each procedures will be elaborated in “Section 2 Get started”.

- *Install the prerequisite software and libraries*
- *Download and install SEIMS*
- *Prepare data (climate, spatial, and observations, etc.) of the study area*
- *Build database for SEIMS-based model by data preprocessing scripts*

- Set up a SEIMS-based model according to model objectives and run model
- Postprocessing, e.g., analyze, plot and graph outputs
- (Optional) Parameter sensitivity analysis
- (Optional) Automatic calibration
- (Optional) Scenario analysis, e.g., BMP scenarios optimization for reducing soil erosion

1:1.2 Publications

1:1.2.1 Watershed modeling framework

Zhu, L.-J., Liu, J., Qin, C.-Z., Zhu, A-X., 2019a. [A modular and parallelized watershed modeling framework](#). *Environmental Modelling & Software* 122, 104526. doi:10.1016/j.envsoft.2019.104526

Liu, J., Zhu, A-X., Qin, C.-Z., Wu, H., Jiang, J., 2016. [A two-level parallelization method for distributed hydrological models](#). *Environmental Modelling & Software* 80, 175–184. doi: 10.1016/j.envsoft.2016.02.032

Liu, J., Zhu, A-X., Liu, Y., Zhu, T., Qin, C.-Z., 2014. [A layered approach to parallel computing for spatially distributed hydrological modeling](#). *Environmental Modelling & Software* 51, 221–227. doi: 10.1016/j.envsoft.2013.10.005

Liu, J., Zhu, A-X., Qin, C.-Z., 2013. [Estimation of theoretical maximum speedup ratio for parallel computing of grid-based distributed hydrological models](#). *Computers & Geosciences* 60, 58–62. doi: 10.1016/j.cageo.2013.04.030

1:1.2.2 BMPs scenario analysis

Zhu, L.-J., Qin, C.-Z., Zhu, A-X., Liu, J., Wu, H., 2019b. [Effects of different spatial configuration units for spatial optimization of watershed best management practices scenarios](#). *Water* 11(2), 262. doi: 10.3390/w11020262

Qin, C.-Z., Gao, H.-R., Zhu, L.-J., Zhu, A-X., Liu, J.-Z., Wu, H., 2018. [Spatial optimization of watershed best management practices based on slope position units](#). *Journal of Soil and Water Conservation* 73(5), 504–517. doi: 10.2489/jswc.73.5.504

Wu, H., Zhu, A-X., Liu, J., Liu, Y., Jiang, J., 2018. [Best Management Practices Optimization at Watershed Scale: Incorporating Spatial Topology among Fields](#). *Water Resource Management* 32, 155–177. doi: 10.1007/s11269-017-1801-8

Wu, H., Liu, Y., Liu, J., Zhu, A-X., 2014. [Representation of Agricultural Best Management Practices in a Fully Distributed Hydrologic Model: A Case Study in the Luoyugou Watershed](#). *Journal of Resources and Ecology* 5(2), 179–184. doi: 10.5814/j.issn.1674-764X.2014.02.011

1:1.3 Support

SEIMS is an open source software. Support is provided through the Github issues and Email of developers.

- SEIMS issues: <https://github.com/lreis2415/SEIMS/issues>
- Emails of developers:
 - Dr. Liang-Jun Zhu (zlj@lreis.ac.cn)
 - Dr. Junzhi Liu (liujunzhi@njnu.edu.cn)

1.2. Why SEIMS?

In the current pre-released version, this part is not yet written, please refers to Zhu *et al.* (2019a) to get the necessary information.

1:3. Terminology

SEIMS Main Program – The executable program (`seims_mpi` for MPI&OpenMP version, `seims_omp` for OpenMP version or `seims` for serial version) which would read all configuration files, load all configured modules and input data (includes climate, spatial data, and management data, etc.), and outputs specified outputs.

SEIMS Module – A dynamic link library file (i.e., `.dll` on Windows, `.so` on Unix-like systems, or `.dylib` on macOS) file which follows SEIMS module APIs and could be loaded by SEIMS main program. A SEIMS module is corresponding to one watershed subprocess.

SEIMS-based watershed model – A SEIMS-based watershed model consists of one version of SEIMS main program (i.e., OpenMP version or MPI&OpenMP version), several customized SEIMS modules, and the watershed database. A SEIMS-based watershed model is prepared as a folder (e.g., `demo_youwuzhen30m_longterm_model`) which consists of several SEIMS configuration files, e.g., the `config.fig` file designed to define the selected SEIMS modules and the execution orders during the simulation of the watershed model.

1:4. References

- Abbaspour, K.C., Rouholahnejad, E., Vaghefi, S., Srinivasan, R., Yang, H., Kløve, B., 2015. A continental-scale hydrology and water quality model for Europe: Calibration and uncertainty of a high-resolution large-scale SWAT model. *J. Hydrol.* 524, 733–752. <https://doi.org/10.1016/j.jhydrol.2015.03.027>
- Arnold, J.G., Moriasi, D.N., Gassman, P.W., Abbaspour, K.C., White, M.J., Srinivasan, R., Santhi, C., Harmel, R.D., Van Griensven, A., Van Liew, M.W., 2012. SWAT: Model use, calibration, and validation. *Trans. ASABE* 55, 1491–1508. <https://doi.org/10.13031/2013.42256>
- Arnold, J.G., Srinivasan, R., Muttiah, R.S., Williams, J.R., 1998. Large area hydrologic modeling and assessment part I: Model development. *J. Am. Water Resour. Assoc. JAWRA* 34, 73–89. <https://doi.org/10.1111/j.1752-1688.1998.tb05961.x>
- Band, L.E., Tague, C.L., Brun, S.E., Tenenbaum, D.E., Fernandes, R.A., 2000. Modelling watersheds as spatial object hierarchies: structure and dynamics. *Trans. GIS* 4, 181–196. <https://doi.org/10.1111/1467-9671.00048>
- Bieger, K., Arnold, J.G., Rathjens, H., White, M.J., Bosch, D.D., Allen, P.M., Volk, M., Srinivasan, R., 2016. Introduction to SWAT+, A Completely Restructured Version of the Soil and Water Assessment Tool. *J. Am. Water Resour. Assoc. JAWRA* 53, 115–130. <https://doi.org/10.1111/1752-1688.12482>
- Campolongo, F., Cariboni, J., Saltelli, A., 2007. An effective screening design for sensitivity analysis of large models. *Environ. Model. Softw.*, Modelling, computer-assisted simulations, and mapping of dangerous phenomena for hazard assessment 22, 1509–1518. <https://doi.org/10.1016/j.envsoft.2006.10.004>
- Chen, S., Zha, X., 2016. Evaluation of soil erosion vulnerability in the Zhuxi watershed, Fujian Province, China. *Nat. Hazards* 82, 1589–1607. <https://doi.org/10.1007/s11069-016-2258-4>
- Chen, Z.-B., Z.-Q. Chen, and H. Yue. 2013. Comprehensive research on soil and water conservation in granite red soil region: A case study of Zhuxi watershed, Changting County, Fujian province. Beijing, Science Press. (*in Chinese*)
- Clark, M.P., Bierkens, M.F.P., Samaniego, L., Woods, R.A., Uijlenhoet, R., Bennett, K.E., Pauwels, V.R.N., Cai, X., Wood, A.W., Peters-Lidard, C.D., 2017. The evolution of process-based hydrologic models: historical challenges and the collective quest for physical realism. *Hydrol. Earth Syst. Sci.* 21, 3427–3440. <https://doi.org/10.5194/hess-21-3427-2017>
- Cukier, R.I., Levine, H.B., Shuler, K.E., 1978. Nonlinear sensitivity analysis of multiparameter model systems. *J. Comput. Phys.* 26, 1–42. [https://doi.org/10.1016/0021-9991\(78\)90097-9](https://doi.org/10.1016/0021-9991(78)90097-9)
- David, O., Ascough II, J.C., Lloyd, W., Green, T.R., Rojas, K.W., Leavesley, G.H., Ahuja, L.R., 2013. A software engineering perspective on environmental modeling framework design: The Object Modeling System. *Environ. Model. Softw.*, Thematic

- Issue on the Future of Integrated Modeling Science and Technology 39, 201–213. <https://doi.org/10.1016/j.envsoft.2012.03.006>
- Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* 6, 182 – 197. <https://doi.org/10.1109/4235.996017>
- Formetta, G., Antonello, A., Franceschi, S., David, O., Rigon, R., 2014. Hydrological modelling with components: A GIS-based open-source framework. *Environ. Model. Softw.* 55, 190–200. <https://doi.org/10.1016/j.envsoft.2014.01.019>
- Foster, I., 1995. Designing and Building Parallel Programs: Concepts and Tools for Parallel Software Engineering. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.
- Freeze, R.A., Harlan, R.L., 1969. Blueprint for a physically-based, digitally-simulated hydrologic response model. *J. Hydrol.* 9, 237–258. [https://doi.org/10.1016/0022-1694\(69\)90020-1](https://doi.org/10.1016/0022-1694(69)90020-1)
- Hill, C., DeLuca, C., Balaji, Suarez, M., Silva, A. da, 2004. The Architecture of the Earth System Modeling Framework. *Comput. Sci. Eng.* 6, 18–28. <https://doi.org/10.1109/MCISE.2004.1255817>
- Hold-Geoffroy, Y., Gagnon, O., Parizeau, M., 2014. Once you SCOOP, no need to fork, in: Proceedings of the 2014 Annual Conference on Extreme Science and Engineering Discovery Environment. ACM, Atlanta, GA. <https://doi.org/10.1145/2616498.2616565>
- Iman, R.L., Shortencarier, M.J., 1984. Fortran 77 program and user's guide for the generation of Latin hypercube and random samples for use with computer models (No. NUREG/CR-3624; SAND-83-2365). Sandia National Labs., Albuquerque, NM (USA). <https://doi.org/10.2172/7091452>
- Karypis, G., Kumar, V., 1998. A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs. *SIAM J. Sci. Comput.* 20, 359–392. <https://doi.org/10.1137/S1064827595287997>
- Kneis, D., 2015. A lightweight framework for rapid development of object-based hydrological model engines. *Environ. Model. Softw.* 68, 110–121. <https://doi.org/10.1016/j.envsoft.2015.02.009>
- Leavesley, G.H., Markstrom, S.L., Viger, R.J., 2006. USGS modular modeling system (MMS)--precipitation-runoff modeling system (PRMS), in: Singh, V.P., Frevert, D.K. (Eds.), *Watershed Models*. CRC Press, Boca Raton, FL., pp. 159–177.
- Liu, J., Zhu, A.-X., Liu, Y., Zhu, T., Qin, C.-Z., 2014. A layered approach to parallel computing for spatially distributed hydrological modeling. *Environ. Model. Softw.* 51, 221–227. <https://doi.org/10.1016/j.envsoft.2013.10.005>
- Liu, J., Zhu, A.-X., Qin, C.-Z., 2013. Estimation of theoretical maximum speedup ratio for parallel computing of grid-based distributed hydrological models. *Comput. Geosci.* 60, 58–62. <https://doi.org/10.1016/j.cageo.2013.04.030>

- Liu, J., Zhu, A.-X., Qin, C.-Z., Wu, H., Jiang, J., 2016. A two-level parallelization method for distributed hydrological models. *Environ. Model. Softw.* 80, 175–184.
<https://doi.org/10.1016/j.envsoft.2016.02.032>
- Liu, Y.B., 2004. Development and application of a GIS-based distributed hydrological model for flood prediction and watershed management. Vrije Universiteit Brussel, Belgium.
- Liu, Y.B., Gebremeskel, S., De Smedt, F., Hoffmann, L., Pfister, L., 2003. A diffusive transport approach for flow routing in GIS-based flood modeling. *J. Hydrol.* 283, 91–106. [https://doi.org/10.1016/S0022-1694\(03\)00242-7](https://doi.org/10.1016/S0022-1694(03)00242-7)
- Maringanti, C., Chaubey, I., Popp, J., 2009. Development of a multiobjective optimization tool for the selection and placement of best management practices for nonpoint source pollution control. *Water Resour. Res.* 45, W06406.
<https://doi.org/10.1029/2008WR007094>
- Moore, R.V., Tindall, C.I., 2005. An overview of the open modelling interface and environment (the OpenMI). *Environ. Sci. Policy, Research & Technology Integration in Support of the European Union Water Framework Directive* 8, 279–286. <https://doi.org/10.1016/j.envsci.2005.03.009>
- Moriasi, D.N., Arnold, J.G., Van Liew, M.W., Bingner, R.L., Harmel, R.D., Veith, T.L., 2007. Model evaluation guidelines for systematic quantification of accuracy in watershed simulations. *Trans. ASABE* 50, 885–900.
<https://doi.org/10.13031/2013.23153>
- Morris, M.D., 1991. Factorial Sampling Plans for Preliminary Computational Experiments. *Technometrics* 33, 161–174.
<https://doi.org/10.1080/00401706.1991.10484804>
- Neitsch, S.L., Arnold, J.G., Kiniry, J.R., Williams, J.R., 2011. Soil and Water Assessment Tool Theoretical Documentation, Version 2009. Texas A&M University System, College Station, Texas, USA.
- O'Callaghan, J.F., Mark, D.M., 1984. The extraction of drainage networks from digital elevation data. *Comput. Vis. Graph. Image Process.* 28, 323–344.
[https://doi.org/10.1016/S0734-189X\(84\)80011-0](https://doi.org/10.1016/S0734-189X(84)80011-0)
- Peckham, S.D., Hutton, E.W.H., Norris, B., 2013. A component-based approach to integrated modeling in the geosciences: The design of CSDMS. *Comput. Geosci., Modeling for Environmental Change* 53, 3–12.
<https://doi.org/10.1016/j.cageo.2012.04.002>
- Qin, C.-Z., 2015. Cuda/GPU, in: Shekhar, S., Xiong, H., Zhou, X. (Eds.), *Encyclopedia of GIS*. Springer International Publishing, Cham, pp. 1–7.
https://doi.org/10.1007/978-3-319-23519-6_1606-1
- Qin, C.-Z., Gao, H.-R., Zhu, L.-J., Zhu, A.-X., Liu, J.-Z., Wu, H., 2018. Spatial optimization of watershed best management practices based on slope position units. *J. Soil Water Conserv.* 73, 504–517. <https://doi.org/10.2489/jswc.73.5.504>

- Qin, C.-Z., Zhan, L.-J., Zhu, A.-X., Zhou, C.-H., 2014. A strategy for raster-based geocomputation under different parallel computing platforms. *Int. J. Geogr. Inf. Sci.* 28, 2127–2144. <https://doi.org/10.1080/13658816.2014.911300>
- Qin, C.-Z., Zhu, A.-X., Pei, T., Li, B., Zhou, C., Yang, L., 2007. An adaptive approach to selecting a flow - partition exponent for a multiple - flow - direction algorithm. *Int. J. Geogr. Inf. Sci.* 21, 443 – 458. <https://doi.org/10.1080/13658810601073240>
- Rouholahnejad, E., Abbaspour, K.C., Vejdani, M., Srinivasan, R., Schulin, R., Lehmann, A., 2012. A parallelization framework for calibration of hydrological models. *Environ. Model. Softw.* 31, 28–36. <https://doi.org/10.1016/j.envsoft.2011.12.001>
- Tague, C.L., Band, L.E., 2004. RHESSys: regional hydro-ecologic simulation system-an object-oriented approach to spatially distributed modeling of carbon, water, and nutrient cycling. *Earth Interact.* 8, 1–42. [https://doi.org/10.1175/1087-3562\(2004\)8<1:RRHSSO>2.0.CO;2](https://doi.org/10.1175/1087-3562(2004)8<1:RRHSSO>2.0.CO;2)
- Tarboton, D.G., 2016. Terrain Analysis Using Digital Elevation Models (TauDEM 5.3.7). <http://hydrology.usu.edu/taudem/> (accessed 19 October 2016).
- Vivoni, E.R., Mascaro, G., Mniszewski, S., Fasel, P., Springer, E.P., Ivanov, V.Y., Bras, R.L., 2011. Real-world hydrologic assessment of a fully-distributed hydrological model in a parallel computing environment. *J. Hydrol.* 409, 483–496. <https://doi.org/10.1016/j.jhydrol.2011.08.053>
- Wagener, T., Boyle, D.P., Lees, M.J., Wheater, H.S., Gupta, H.V., Sorooshian, S., 2001. A framework for development and application of hydrological models. *Hydrol. Earth Syst. Sci.* 5, 13–26. <https://doi.org/10.5194/hess-5-13-2001>
- Wang, H., Fu, X., Wang, G., Li, T., Gao, J., 2011. A common parallel computing framework for modeling hydrological processes of river basins. *Parallel Comput.* 37, 302–315. <https://doi.org/10.1016/j.parco.2011.05.003>
- Wang, H., Fu, X., Wang, Y., Wang, G., 2013. A High-performance temporal-spatial discretization method for the parallel computing of river basins. *Comput. Geosci.* 58, 62–68. <https://doi.org/10.1016/j.cageo.2013.04.026>
- Wang, S., Armstrong, M.P., 2009. A theoretical approach to the use of cyberinfrastructure in geographical analysis. *Int. J. Geogr. Inf. Sci.* 23, 169–193. <https://doi.org/10.1080/13658810801918509>
- Wenderholm, E., 2005. Eclpss: a Java-based framework for parallel ecosystem simulation and modeling. *Environ. Model. Softw.* 20, 1081–1100. <https://doi.org/10.1016/j.envsoft.2004.06.006>
- Wu, H., Zhu, A.-X., Liu, J., Liu, Y., Jiang, J., 2018. Best Management Practices Optimization at Watershed Scale: Incorporating Spatial Topology among Fields. *Water Resour Manage* 32, 155–177. <https://doi.org/10.1007/s11269-017-1801-8>
- Wu, Y., Li, T., Sun, L., Chen, J., 2013. Parallelization of a hydrological model using the message passing interface. *Environ. Model. Softw.* 43, 124–132. <https://doi.org/10.1016/j.envsoft.2013.02.002>

- Yalew, S., van Griensven, A., Ray, N., Kokoszkiewicz, L., Betrie, G.D., 2013. Distributed computation of large scale SWAT models on the Grid. Environ. Model. Softw. 41, 223–230. <https://doi.org/10.1016/j.envsoft.2012.08.002>
- Yang, J., 2011. Convergence and uncertainty analyses in Monte-Carlo based sensitivity analysis. Environ. Model. Softw. 26, 444–457. <https://doi.org/10.1016/j.envsoft.2010.10.007>
- Zhan, C.-S., Song, X.-M., Xia, J., Tong, C., 2013. An efficient integrated approach for global sensitivity analysis of hydrological model parameters. Environ. Model. Softw. 41, 39–52. <https://doi.org/10.1016/j.envsoft.2012.10.009>
- Zhang, X., Beeson, P., Link, R., Manowitz, D., Izaurrealde, R.C., Sadeghi, A., Thomson, A.M., Sahajpal, R., Srinivasan, R., Arnold, J.G., 2013. Efficient multi-objective calibration of a computationally intensive hydrologic model with parallel computing software in Python. Environ. Model. Softw. 46, 208–218. <https://doi.org/10.1016/j.envsoft.2013.03.013>
- Zhao, G., Bryan, B.A., King, D., Luo, Z., Wang, E.L., Bende-Michl, U., Song, X., Yu, Q., 2013. Large-scale, high-resolution agricultural systems modeling using a hybrid approach combining grid computing and parallel processing. Environ. Model. Softw. 41, 231–238. <https://doi.org/10.1016/j.envsoft.2012.08.007>
- Zitzler, E., Thiele, L., 1999. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. IEEE Trans. Evol. Comput. 3, 257–271. <https://doi.org/10.1109/4235.797969>
- Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., da Fonseca, V.G., 2003. Performance assessment of multiobjective optimizers: An analysis and review. IEEE Trans. Evol. Comput. 7, 117–132. <https://doi.org/10.1109/TEVC.2003.810758>

Section 2. Get started

SEIMS is mainly written by C++ with the support of GDAL (Geospatial Data Abstraction Library, <https://www.gdal.org/>), mongo-c-driver (<https://github.com/mongodb/mongo-c-driver>), OpenMP (Open Multi-Processing) and MPI (Message Passing Interface), while Python is used for organizing the utility tools such as data preprocessing, postprocessing, parameter sensitivity analysis, auto-calibration, and BMP (Best Management Practices) scenarios analysis.

SEIMS is designed to be an open-source, cross-platform, and high-performance integrated watershed modeling framework. Theoretically, SEIMS can be compiled by common used C/C++ compiler (e.g. Microsoft Visual C++ 2010+, GCC 4.6+, and Intel C++ 12.0+) as 32-bit or 64-bit programs and run on mainstream Operation Systems (e.g. Windows, Linux, and macOS).

In order to save the length of this manual, the software environments with Windows 10 64bit, Microsoft Visual C++ 2013 (MSVC 2013 for short), and Python 2.7.15 are selected for example. For the demo of parallel computing, a Linux cluster with [IBM Platform LSF](#) for workload management is adopted.

Users are encouraged to follow this manual step by step to get started with SEIMS, including download and installation, understanding the data preparation of the demo watershed, preprocessing and running the user-configured SEIMS-based watershed model, postprocessing, parameter sensitivity analysis, auto-calibration, and BMPs scenario analysis, etc.

2:1. Download and installation

2:1.1 Download

SEIMS is hosted on Github (<https://github.com/lreis2415/SEIMS>). SEIMS is dependent on several open-source software, e.g., GDAL (<https://www.gdal.org/>) and mongo-c-driver (<https://github.com/mongodb/mongo-c-driver>). Currently, there are no compiled binaries for distribution, but only through source code.

Users are encouraged to download and install (means compile from the source code) SEIMS manually. Generally, the `master` branch of SEIMS repository (<https://github.com/lreis2415/SEIMS/tree/master>) is the relative stable version, while the `dev` branch reflects the latest development changes.

Users can use `git` to clone the repository or download the compressed `zip` file directly.

- *Clone the single `master` branch using `git`.*
 1. *Download and install `git` from <https://gitforwindows.org/>.*
 2. *Open or create a local directory without spaces in File Explorer, e.g., `D:\demo`, right-click in the space and select “Git Bash Here”. Then, enter the following two commands to configure your basic settings of `git`, i.e., the `username` and `email` of your Github account.*

```
01 $ git config --global user.name "Your Name"
02 $ git config --global user.email "email@example.com"
03 e.g.,
04 $ git config --global user.name "crazyzlj"
05 $ git config --global user.email "crazyzlj@gmail.com"
```

3. *If it is the first time to use `git` in your computer, a SSH key should be created and added to your Github account.*

First, type the following command to generate a SSH key and store to your user directory, e.g., `C:/Users/XXX/.ssh/id_rsa.pub`

```
01 $ ssh-keygen -t rsa -C "email@example.com"
```

Then, copy the content of this file and add to your Github account (“Settings” -> “SSH and GPG keys” -> “New SSH key”).

4. *Clone a single branch, e.g., the `master` branch (Figure 2:1-1).*
- ```
01 $ git clone git@github.com:lreis2415/SEIMS.git --branch master --single-branch
```

```
ZhuLJ@ZHULJ MINGW64 /d/demo
$ git clone git@github.com:lreis2415/SEIMS.git --branch master --single-branch
Cloning into 'SEIMS'...
remote: Enumerating objects: 1004, done.
remote: Counting objects: 100% (1004/1004), done.
remote: Compressing objects: 100% (479/479), done.
remote: Total 26570 (delta 602), reused 788 (delta 523), pack-reused 25566
Receiving objects: 100% (26570/26570), 15.22 MiB | 2.13 MiB/s, done.
Resolving deltas: 100% (19065/19065), done.
ZhuLJ@ZHULJ MINGW64 /d/demo
$
```

**Figure 2:1-1 Clone the single master branch of SEIMS repository using Git Bash command**

- Alternatively, users can download the compressed zip file (e.g., the zip file of the master branch, <https://github.com/lreis2415/SEIMS/archive/master.zip>) directly and then decompress it to a local directory without spaces, e.g., D:\demo.

Now, the source code of SEIMS should be in D:\demo\SEIMS such as Figure 2:1-2.

| Name           | Date modified      | Type             | Size  |
|----------------|--------------------|------------------|-------|
| .git           | 12/25/2018 4:48... | 文件夹              |       |
| cmake          | 12/25/2018 4:48... | 文件夹              |       |
| data           | 12/25/2018 4:48... | 文件夹              |       |
| doc            | 12/25/2018 4:48... | 文件夹              |       |
| gui            | 12/25/2018 4:48... | 文件夹              |       |
| knowledge      | 12/25/2018 4:48... | 文件夹              |       |
| seims          | 12/25/2018 4:48... | 文件夹              |       |
| TODO           | 12/25/2018 4:48... | 文件夹              |       |
| .bookignore    | 12/25/2018 4:48... | BOOKIGNORE Fi... | 1 KB  |
| .editorconfig  | 12/25/2018 4:48... | EDITORCONFIG ... | 1 KB  |
| .gitattributes | 12/25/2018 4:48... | Text Document    | 2 KB  |
| .gitignore     | 12/25/2018 4:48... | Text Document    | 2 KB  |
| .nojekyll      | 12/25/2018 4:48... | NOJEKYLL File    | 0 KB  |
| .travis.yml    | 12/25/2018 4:48... | YML File         | 3 KB  |
| appveyor.yml   | 12/25/2018 4:48... | YML File         | 4 KB  |
| book.json      | 12/25/2018 4:48... | JSON File        | 3 KB  |
| CMakeLists.txt | 12/25/2018 4:48... | TXT File         | 9 KB  |
| LICENSE        | 12/25/2018 4:48... | File             | 35 KB |
| README.md      | 12/25/2018 4:48... | Typora           | 5 KB  |

**Figure 2:1-2 Directory tree of SEIMS source code**

## 2:1.2 Prerequisite software and libraries

Several software and libraries are required to compile and run SEIMS successfully.

- *Software*
  1. C/C++ compiler with partial support for C++11, e.g., MSVC 2010+, GCC 4.6+, Intel C++ 12.0+, and Clang 8.0+
  2. CMake 3.1+
  3. Python 2.7.x or 3.x
  4. MongoDB (community) server
- *Libraries*
  5. MPI implementation for C/C++, e.g., MS-MPI v6+, MPICH, OpenMPI or Intel MPI
  6. GDAL 1.x or 2.x for C/C++ and Python
  7. mongo-c-driver 1.5+ for C/C++
  8. third-party Python packages

The following instructions are prepared and tested on Windows 10 64bit.

### 2:1.2.1 Microsoft Visual C++ (MSVC)

SEIMS uses several features of C++11 such as `nullptr` and `auto` keywords. Therefore, the minimum support version is MSVC 2010. Microsoft Visual Studio is a powerful IDE

based on Microsoft Visual C++. If you don't want to install Microsoft Visual Studio, the [Visual C++ build tools](#) can allow you to build C++ libraries and applications targeting Windows desktop, which are the same tools that you find in Microsoft Visual Studio. Even though, Visual Studio is still highly recommended. Unless stated, the following MSVC refers to Microsoft Visual Studio.

If you want to develop parallel applications based on MPI, MSVC 2010 is the best choice since it is the last MSVC version that integrated the MPI cluster debugger. MPI cluster debugger is the most convenient and powerful tools on Windows to debug MPI-based parallel applications. More details about MPI cluster debugger can be found in [https://docs.microsoft.com/en-us/previous-versions/visualstudio/visual-studio-2010/dd560809\(v=vs.100\)](https://docs.microsoft.com/en-us/previous-versions/visualstudio/visual-studio-2010/dd560809(v=vs.100)). Otherwise, MSVC 2013 and 2015 are both recommended. All these MSVC versions can be downloaded from <https://visualstudio.microsoft.com/vs/older-downloads/>.

This manual takes "Microsoft Visual Studio 2013 update 5" as an example.

### 2:1.2.2 CMake

CMake 3.1+ is used to build the C/C++ project of SEIMS such as Microsoft Visual Studio solution (\*.sln). CMake can be downloaded from the official site <https://cmake.org/download/>. The installer is something like `cmake-3.x.x-win64-x64.msi` for Windows64-x64.

After the installation of CMake, please add the path of CMake executable, e.g., `C:\Program Files (x86)\CMake\bin`, to the system variable `PATH`. In such a way, we can run CMake in CMD (Command Prompt) directly by typing `cmake` rather than the absolute path.

### 2:1.2.3 Python

The utility tools of SEIMS is compatible with Python 2.7.x and 3.x versions. Download from <https://www.python.org/downloads/windows/> and install to a local location, e.g., [Python 2.7.15 x86-64 version](#) installed at `D:\demo\python27`. Same to CMake, please add the paths of Python, e.g., `D:\demo\python27`; `D:\demo\python27\Scripts` to the system variable `PATH`.

`Pip` (<https://pypi.org/project/pip/>), the PyPA (Python Packaging Authority, <https://www.pypa.io/en/latest/>) recommended tool for installing Python packages, should be installed right after the installation of Python. If `pip` has been selected and installed along with Python, this step will perform an update for `pip`.

- *Download `get_pip.py` from <https://bootstrap.pypa.io/get-pip.py> to a local directory, e.g., `D:\demo`*
- *Open a CMD (Start->Windows System->Command Prompt), and enter the following command.*  
`python d:\demo\get-pip.py`
- *The latest version of pip is now installed or updated! Most Python packages can be installed by running `pip install <package name>` in a CMD window. Pip also*

*support the installation of offline compiled binary file, i.e., the \*.whl format file which is the new standard built-package format for Python (<https://pypi.org/project/wheel/>) by running pip install </path/to/whl-file-of-package>.*

---

**Note:**

1. If ArcGIS has been installed on the computer, the 32-bit and 64-bit versions of Python may also be installed at the path of C:\Python27\ArcGIS10.3 and C:\Python27\ArcGISx6410.3, respectively, so to support the arcpy. However, we still recommend installing an independent Python so to avoid the confusion of Python packages that may affect the work of arcpy.
  2. Only one specific path of Python should be existed in the PATH environment. When users want to use a different version of Python, the absolute path (e.g., D:\demo\python27\python.exe) should be used instead of the single name of python.
- 

## 2:1.2.4 MongoDB server

Given the requirements of flexible data structures, elastic scalability, and high performance, a widely-used NoSQL database, MongoDB (<https://www.mongodb.com>), was adopted to manage all kinds of data in SEIMS.

The free available version of MongoDB server is MongoDB community server which can be downloaded from the official website, such as the previous stable release version 3.6.9 [https://fastdl.mongodb.org/win32/mongodb-win32-x86\\_64-2008plus-ssl-3.6.9-signed.msi](https://fastdl.mongodb.org/win32/mongodb-win32-x86_64-2008plus-ssl-3.6.9-signed.msi).

### 2:1.2.4.1 Installation and simple test

- *Install the MongoDB community server (take mongodb-win32-x86\_64-2008plus-ssl-3.2.3-signed.msi as example) to a local directory without spaces, e.g., D:\MongoDB.*
- *Create a new directory named logs (e.g., D:\MongoDB\logs) to store journal files.*
- *Create a new directory named db (e.g., D:\MongoDB\db) to store data.*
- *Open a CMD as administrator and run the following commands:*  
D:\MongoDB\bin\mongod.exe --dbpath=D:\mongodb\db
- *The last line printed in the CMD windows should be like:*  
Tue Oct 09 11:50:55 [websvr] admin web console waiting for connections on port 27017  
*This means MongoDB server has started successfully and the port 27017 is occupied by MongoDB.*
- *DO NOT close the current CMD window and open a new one as administrator, and run the following commands:*  
D:\MongoDB\bin\mongo.exe  
*If the last line is connecting to: test, the connection to MongoDB is successful. The default database named test is connected.*
- *Continue to test:*

```
01 use test
02 db.foo.save({hello:0})
03 db.foo.find()
```

*If something like { "\_id" : ObjectId("5c222fdff492e1436718e00b") , "Hello" : 0 } showed in the last line, the key-value data (i.e., key is hello and value is 0) has been inserted into the newly created collection foo in test database successfully.*

- Enter `exit` to leave MongoDB.

#### 2:1.2.4.2 Register MongoDB as system service

As is shown in previous section, a CMD window MUST be active to keep the running of MongoDB server, which is tedious. Therefore, we wish to set MongoDB as system service and start it along with Windows system, so that we do not need to start it manually each time after restarts of Windows system.

- Open a CMD as administrator and enter the following commands:  
`D:\MongoDB\bin\mongod.exe --dbpath=D:\MongoDB\db --logpath=D:\MongoDB\logs\mongodb.log --install --serviceName "MongoDB"`
- If something like “Tue Oct 09 12:05:15 Service can be started from the command line with 'net start MongoDB'” showed in the console window (or maybe nothing showed), the service has been successfully registered.
- Now, a simple command `net start MongoDB` is enough to start MongoDB server. This command will set the MongoDB server as automatically start service simultaneously.
  - Alternatively, we can also manually set the MongoDB service as automatically start service along with Windows system.  
`D:\MongoDB\bin\mongod --install --serviceName "MongoDB" --serviceDisplayName "MongoDB" --logpath D:\MongoDB\logs\mongodb.log --logappend --dbpath D:\MongoDB\db --directoryperdb`
  - If no errors occur, the automatically start service has been set successfully!

#### 2:1.2.4.3 MongoDB GUI

In order to view, query, and update data stored in MongoDB, a user-friendly and efficient GUI is urgently needed.

Robo 3T is what we need (<https://robomongo.org/download>). Download the proper version for your system and install it.

Open Robo 3T and connect to the `localhost:27017` (i.e., `127.0.0.1:27017`). The key-pair data inserted in 2:1.2.4.1 can be found (Figure 2:1-3).

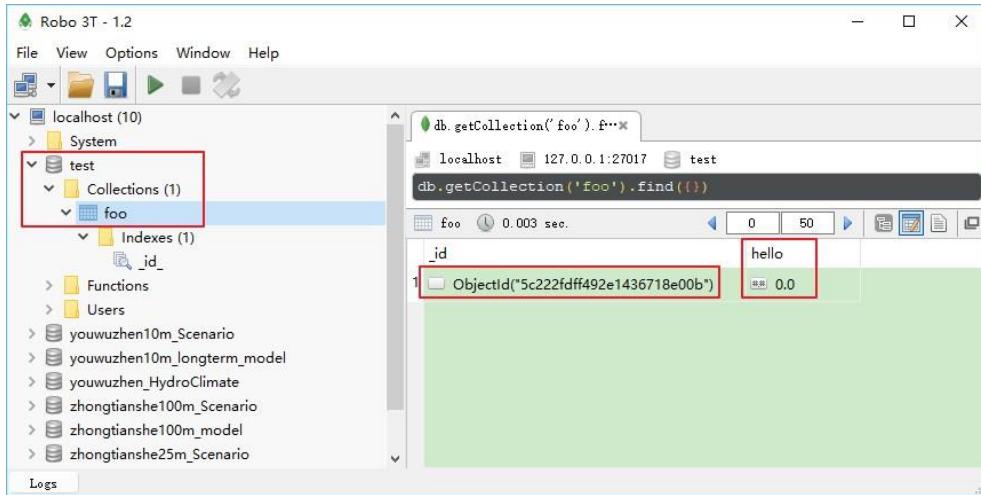


Figure 2:1-3 Screenshot of Robo 3T – a user-friendly and efficient MongoDB GUI

### 2:1.2.5 Microsoft MPI (MS-MPI)

Microsoft MPI (MS-MPI) is a Microsoft implementation of the MPI (Message Passing Interface, <https://www.mpi-forum.org/>) standard for developing and running parallel applications on the Windows platform. MS-MPI v6 and later versions are recommended.

Download MS-MPI (two installer files are required, i.e., `msmpisdk.msi` and `MSMpisetup.exe`) from Microsoft download center, e.g., <https://www.microsoft.com/en-us/download/details.aspx?id=47259> for MS-MPI v6, and install to the default locations. By default, the environment variables of MS-MPI will be set automatically. Open `System->Advanced system settings->Environment variables->System variables` to make sure the following variable-value pairs are existed. If not, please add them manually with the correct paths of your computer.

```
MSMPI_BIN=C:\Program Files\Microsoft MPI\Bin\
MSMPI_INC=C:\Program Files (x86)\Microsoft SDKs\MPI\Include\
MSMPI_LIB32=C:\Program Files (x86)\Microsoft SDKs\MPI\Lib\x86\
MSMPI_LIB64=C:\Program Files (x86)\Microsoft SDKs\MPI\Lib\x64\
```

### 2:1.2.6 GDAL library for C/C++ and Python

Download the pre-compiled library is the most convenient and successful way to install GDAL library for both C/C++ and Python.

#### 2:1.2.6.1 GDAL for C/C++

The “GISInternals support site” (<http://www.gisinternals.com/index.html>) maintained by [Tamas Szekeres](#) is the most famous site that provides compiled GDAL library by MSVC on Windows platform. Users should pick up the exact version according to your compiler and OS architecture, e.g., I have MSVC 2013 (`_MSC_VER=1800`) installed on Windows 10-64bit and I want `GDAL-1.11.4` to compile 64-bit applications, so I should download the `'release-1800-x64-gdal-1-11-4-mapserver-6-4-3.zip'` and `'release-1800-x64-gdal-1-11-4-mapserver-6-4-3-libs.zip'` from archived releases (<http://www.gisinternals.com/archive.php>).

- Unzip these two zip files to the same directory path without spaces, e.g., C:/GDAL. The directory tree of GDAL should like Figure 2:1-4.

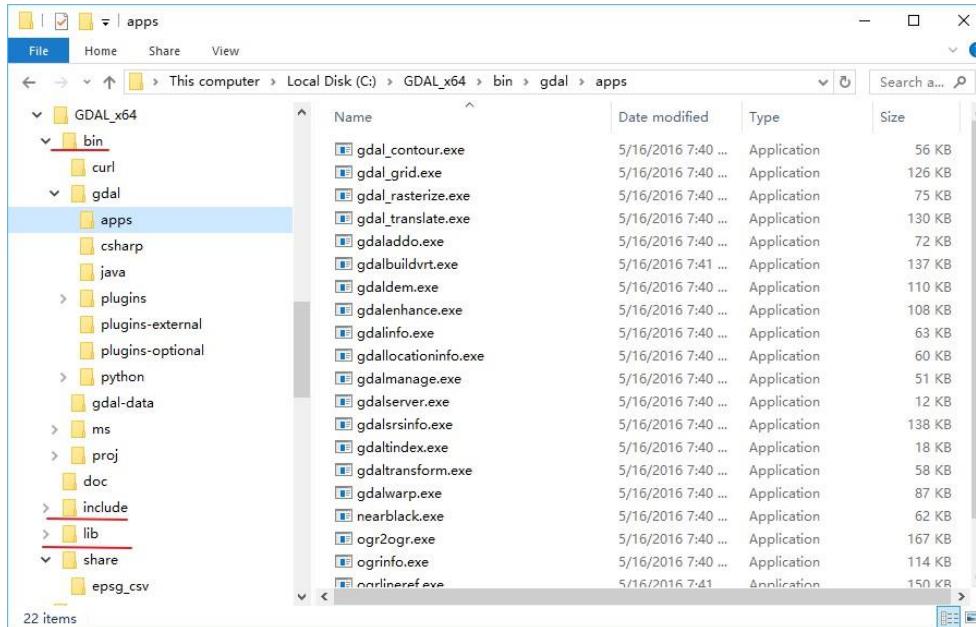


Figure 2:1-4 Directory tree of the compiled GDAL library

- Open Advanced System Properties->Environment variables, create several new system variables under the System variables pane:

```
GDAL_DIR=C:\GDAL
GDAL_DATA=C:\GDAL\bin\gdal-data
GDAL_PATHS=C:\GDAL;C:\GDAL\bin;C:\GDAL\bin\proj\apps;C:\GDAL\bin\gdal\apps;C:\GDAL\bin\ms\apps;C:\GDAL\bin\curl;
```

- Then, append %GDAL\_PATHS% to the end of the system variable PATH.
- The GDAL library for C/C++ has been installed, as well as executable utility tools of GDAL, e.g., gdalinfo. Open a new CMD window, and enter gdalinfo --version, something like GDAL 1.11.4, released 2016/01/25 should be printed.

### 2:1.2.6.2 GDAL for Python

Although the “[GISInternals support site](#)” also provides the GDAL Python bindings (e.g., [GDAL-1.11.4.win-amd64-py2.7.msi](#)), it may not work properly for unknown reason. So, the site of “Unofficial Windows Binaries for Python Extension Packages (<https://www.lfd.uci.edu/~gohlke/pythonlibs/>)” maintained by [Christoph Gohlke](#) is highly recommended to download the compiled binaries of Python packages (i.e., \*.whl files), not only the GDAL but also almost the commonly used packages.

Please read the instructions at the head of this website very carefully.

- Select 32- (win32) or 64-bit (win\_amd64) binaries of packages according to the bit version of Python, not the version of your Windows. In other words, even if the Windows is 64-bit, the Python can be 32- or 64-bit.

- Select the correct version according to the version number of Python, e.g., cp27-cp27m for Python 2.7.x, cp34-cp34m for Python 3.4.x, ex analogia.

- Install Microsoft Visual C++ Redistributable packages first, since many binaries cannot run without them, i.e., Microsoft Visual C++ 2008 ([x64](#), [x86](#), and [SP1](#) for CPython 2.7), Visual C++ 2010 ([x64](#) and [x86](#) for CPython 3.4), or the Visual C++ 2017 ([x64 or x86](#) for CPython 3.5, 3.6, and 3.7).

- Install numpy+mkl (<https://www.lfd.uci.edu/~gohlke/pythonlibs/#numpy>), e.g., numpy-1.15.4+mkl-cp27-cp27m-win\_amd64.whl.

```

01 d:\demo>d:\demo\python27\scripts\pip install numpy-1.15.4+mkl-cp27-
cp27m-win_amd64.whl
02 Processing .\numpy-1.15.4+mkl-cp27-cp27m-win_amd64.whl
03 Installing collected packages: numpy
04 Successfully installed numpy-1.15.4+mkl

```

- Install GDAL (<https://www.lfd.uci.edu/~gohlke/pythonlibs/#gdal>), e.g. GDAL-2.2.4-cp27-cp27m-win\_amd64.whl.

```

01 d:\demo>d:\demo\python27\scripts\pip install GDAL-2.2.4-cp27-
cp27m-win_amd64.whl
02 Processing .\gdal-2.2.4-cp27-cp27m-win_amd64.whl
03 Installing collected packages: GDAL
04 Successfully installed GDAL-2.2.4

```

- (Optional) Install Esri's [FileGDB API 1.3](#) or [FileGDB 1.5](#) if you want to use the FileGDB plugin for GDAL. Otherwise, just delete the ogr\_FileGDB.dll located in D:\demo\python27\Lib\site-packages\osgeo\gdalplugins if the related errors occurred.

- Open a CMD window and enter the following commands to test if the GDAL package for Python is successfully installed.

```

01 d:\demo>d:\demo\python27\python
02 Python 2.7.15 (v2.7.15:ca079a3ea3, Apr 30 2018, 16:30:26) [MSC
v.1500 64 bit (AMD64)] on win32
03 Type "help", "copyright", "credits" or "license" for more
information.
04 >>> import osgeo
05 >>> osgeo.__version__
06 '2.2.4'
07 >>> from osgeo import ogr
08 >>> from osgeo import osr
09 >>> from osgeo import gdalconst
10 >>> from osgeo import gdal_array
11 >>> from osgeo import gdal

```

As we can see, the version of GDAL library for Python in my computer is 2.2.4, which means the GDAL version for C/C++ (which is 1.11.4) and Python are not necessarily the same.

### 2.1.2.7 mongo-c-driver library

The mongo-c-driver library for MongoDB (<https://github.com/mongodb/mongo-c-driver>) is adopted by SEIMS to handle data Input/Output (IO) with MongoDB. Since mongo-c-driver is still under active and continuous development, the building method and API (Application Program Interface) functions may different with versions. So, for the most compatibility, the [version of 1.6.1](#) is recommended, since SEIMS is mainly developed and tested based on mongo-c-driver 1.5.x ~ 1.6.x.

- *Download the released file and decompress it to a local directory, e.g., D:\demo\mongo-c-driver-1.6.1*
- *Open VS2013 Developer Command Prompt, and enter the following commands (start with ‘>’ symbol) in order. Assume that the desired destination of compiled libraries and headers of mongo-c-driver is C:\mongo-c-driver.*

```
01 >cd D:\demo\mongo-c-driver-1.6.1
02 # 1. Configure, build, and install libbson first.
03 # Note that, in the latest version, libbson is no need to be built
separately.
04 # Please visit http://mongoc.org/libmongoc/current/installing.html
for more information.
05 >cd src\libbson
06 >cmake -DCMAKE_INSTALL_PREFIX=C:\mongo-c-driver -G "Visual Studio
12 2013 Win64"
07 >msbuild.exe ALL_BUILD.vcxproj /p:Configuration=RelWithDebInfo
08 >msbuild.exe INSTALL.vcxproj /p:Configuration=RelWithDebInfo
09 # 2. Go back to the root folder, configure, build, and install
mongoc
10 >cd ..
11 >cmake -DCMAKE_INSTALL_PREFIX=C:\mongo-c-driver -
DBSON_ROOT_DIR=C:\mongo-c-driver -G "Visual Studio 12 2013 Win64" -
DCMAKE_PREFIX_PATH=C:\mongo-c-driver\lib\cmake -
DENABLE_AUTOMATIC_INIT_AND_CLEANUP:BOOL=OFF -DENABLE_SSL=WINDOWS -
DENABLE_SASL=SSPI
12 # Use Windows Native TLS, rather than OpenSSL in case of strange
link errors
13 # Use Windows Native SSPI, rather than Cyrus SASL
14 >msbuild.exe ALL_BUILD.vcxproj /p:Configuration=RelWithDebInfo
15 >msbuild.exe INSTALL.vcxproj /p:Configuration=RelWithDebInfo
```

*Notes:*

1. *If msbuild.exe cannot be found by CMD, please find it in the path of .NetFramework, e.g., C:\Windows\Microsoft.NET\Framework64\v4.0.30319\msbuild.exe.*
  2. *If you want to build the 32-bit mongo-c-driver libraries, please replace the generator "Visual Studio 12 2013 Win64" to "Visual Studio 12 2013".*
- *Now, mongo-c-driver has been built and installed, the directories of bin, include, and lib can be found in the root directory of mongo-c-driver, i.e., C:\mongo-c-driver.*

- Create two new system variables:  

```
MONGOC_ROOT=C:\mongo-c-driver
MONGOC_LIB=C:\mongo-c-driver\bin
```
- Then, append `%MONGOC_LIB%` to the end of the system variable PATH.

### 2:1.2.8 Python packages for SEIMS utility tools

SEIMS utility tools written by Python require several third-party packages. Except for Numpy and GDAL that have been installed in Section 2:1.2.6.2, there is one more package should be installed manually, i.e., PyGeoC (short for **P**ython for **G**eocomputation, <https://github.com/lreis2415/PyGeoC>). PyGeoC is designed to provide commonly used functions for Geocomputation, e.g., watershed delineation workflow based on [TauDEM](#). Besides, other required packages can be installed automatically by pip.

- The installation of PyGeoC
    1. Get the latest version of PyGeoC from the master branch (<https://github.com/lreis2415/PyGeoC/tree/master>) and save to a local directory, e.g. D:\demo\PyGeoC.  
*Alternatively, users are recommended to clone the master branch using git. With git, users can easily get the latest changes of PyGeoC by the command of 'git pull origin master'.*
    2. Open a CMD windows as administrator and using the following command to install:  

```
D:\demo\PyGeoC\reinstall.bat
```

*The reinstall.bat script also accepts one input argument to specify the exact directory path of Python, e.g.:*  

```
D:\demo\PyGeoC\reinstall.bat D:\demo\python27
```
    3. Wait for a moment, PyGeoC and its dependencies will be installed automatically, the messages of success are something like:  

```
Successfully installed PyGeoC-0.3.0 backports.functools-lru-cache-1.5 cycler-0.10.0 future-0.17.1 kiwisolver-1.0.1 matplotlib-2.2.3 pyparsing-2.3.0 python-dateutil-2.7.5 pytz-2018.7 typing-3.6.6
```
  - The installation of other required packages  

```
pip install -r D:\demo\SEIMS\seims\requirements.txt
```

*Wait for a moment, the required packages and their dependencies will be installed automatically, the messages of success are something like:*  

```
Successfully installed configparser-3.5.0 SALib-1.2 Shapely-1.6.4.post2 argparse-1.4.0 deap-1.2.2 decorator-4.3.0 greenlet-0.4.15 networkx-2.2 pandas-0.23.4 pymongo-3.7.2 pyzmq-17.1.2 scipy-1.2.0 scoop-0.7.1.1
```
- 
- Note: If you want to use the Python which is not the default version on your computer, please use the absolute path of pip to install packages, e.g., D:\demo\python27\Scripts\pip.*

Now, the Python environment for SEIMS has been set up!

## 2:1.3 Test of the C/C++ building environment

Now, we have set up the C/C++ building environment for SEIMS, i.e., MSVC 2013, MS-MPI v6, GDAL 1.11.4, and mongo-c-driver 1.6.1. In case of any unpredictable omissions or errors, users are highly recommended to test the C/C++ building environment by compiling the **Common Cross-platform Geographic-computing Library (CCGL)**, <https://github.com/crazyzlj/CCGL>) and running its unit test. CCGL has been integrated into SEIMS and no additional download is required.

Please follow the steps below to compile CCGL and run its unit test.

- Open a new VS2013 Developer Command Prompt, and navigate to the directory of CCGL, e.g., D:\demo\SEIMS\seims\src\ccgl.
- Build the Visual Studio solution by CMake with the support of Google Test (i.e., -DUNITTEST=1), such as Figure 2:1-5.

```
01 >cd D:\demo\SEIMS\seims\src\ccgl
02 >d:
03 >mkdir build
04 >cd build
05 >cmake -G "Visual Studio 12 2013 Win64" -DUNITTEST=1 ..
06 >msbuild.exe ALL_BUILD.vcxproj /p:Configuration=RelWithDebInfo
07 >msbuild.exe INSTALL.vcxproj /p:Configuration=RelWithDebInfo
```

The screenshot shows a terminal window with the title 'VS2013 开发人员命令提示'. The command history at the top includes:

```
C:\Program Files (x86)\Microsoft Visual Studio 12.0>cd D:\demo\SEIMS\seims\src\ccgl
C:\Program Files (x86)\Microsoft Visual Studio 12.0>
D:\demo\SEIMS\seims\src\ccgl>mkdir build
D:\demo\SEIMS\seims\src\ccgl>cd build
```

The main output area shows the execution of CMake and MSBuild commands:

```
D:\demo\SEIMS\seims\src\ccgl>cmake -G "Visual Studio 12 2013 Win64" -DUNITTEST=1 ..
-- The CXX compiler identification is MSVC 18.0.40629.0
-- The C compiler identification is MSVC 18.0.40629.0
-- Check for working CXX compiler: C:/Program Files (x86)/Microsoft Visual Studio 12.0/VC/bin/x86_amd64/cl.exe
-- Check for working CXX compiler: C:/Program Files (x86)/Microsoft Visual Studio 12.0/VC/bin/x86_amd64/cl.exe -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Check for working C compiler: C:/Program Files (x86)/Microsoft Visual Studio 12.0/VC/bin/x86_amd64/cl.exe
-- Check for working C compiler: C:/Program Files (x86)/Microsoft Visual Studio 12.0/VC/bin/x86_amd64/cl.exe -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Using half of the processors to compiling: 4
-- Try OpenMP C flag [-fopenmp]
-- Performing Test OpenMP_FLAG_DETECTED
-- Performing Test OpenMP_FLAG_DETECTED - Success
-- Try OpenMP CXX flag [-fopenmp]
-- Performing Test OpenMP_CXX_FLAG_DETECTED
-- Performing Test OpenMP_CXX_FLAG_DETECTED - Success
-- Found OpenMP: /openmp
-- Compiling with OpenMP...
-- Found GDAL: C:/GDAL_x64/lib/gdal_i.lib
-- Compiling with GDAL...
-- Found MONGOC: C:/mongo-c-driver/lib/mongoc-1.0.1.lib;ws2_32
-- Found BSON: C:/mongo-c-driver/lib/bson-1.0.lib;ws2_32
-- Compiling with mongo-c-driver...
-- GoogleTest Version: release-1.8.0
-- Downloading/updating googletest
-- Configuring done
```

Annotations in red highlight specific findings:

- OpenMP FOUND
- GDAL FOUND
- mongo-c-driver FOUND
- Download GoogleTest automatically

Figure 2:1-5 Building Visual Studio solution of CCGL library by CMake

- Then, navigate to the binary directory of CCGL and run its unit test.
- ```
01 >cd ..\bin
02 >UnitTests_CCGL.exe
```
- The final results should be like Figure 2:1-6.

```

VS2013 开发人员命令提示
[-----] 2 tests from MaskLayer/clsRasterDataTestIncrstMaskNoPos
[RUN] MaskLayer/clsRasterDataTestIncrstMaskNoPos.RasterIO/0
Read D:\demo\SEIMS\seims\src\ccgl\bin\./data/raster/mask1.asc...
Release raster: mask1
[OK] MaskLayer/clsRasterDataTestIncrstMaskNoPos.RasterIO/0 (19 ms)
[RUN] MaskLayer/clsRasterDataTestIncrstMaskNoPos.RasterIO/1
Read D:\demo\SEIMS\seims\src\ccgl\bin\./data/raster/mask1.tif...
Release raster: mask1
[OK] MaskLayer/clsRasterDataTestIncrstMaskNoPos.RasterIO/1 (19 ms)
[-----] 2 tests from MaskLayer/clsRasterDataTestIncrstMaskNoPos (48 ms total)

[-----] 2 tests from MaskLayer/clsRasterDataTestIncrstMaskPos
[RUN] MaskLayer/clsRasterDataTestIncrstMaskPos.RasterIO/0
Read D:\demo\SEIMS\seims\src\ccgl\bin\./data/raster/mask1.asc...
Release raster:
Release raster: mask1
[OK] MaskLayer/clsRasterDataTestIncrstMaskPos.RasterIO/0 (63 ms)
[RUN] MaskLayer/clsRasterDataTestIncrstMaskPos.RasterIO/1
Read D:\demo\SEIMS\seims\src\ccgl\bin\./data/raster/mask1.tif...
Release raster:
Release raster: mask1
[OK] MaskLayer/clsRasterDataTestIncrstMaskPos.RasterIO/1 (76 ms)
[-----] 2 tests from MaskLayer/clsRasterDataTestIncrstMaskPos (149 ms total)

[-----] Global test environment tear-down
[=====] 65 tests from 27 test cases ran. (5253 ms total)
[PASSED] 65 tests.
D:\demo\SEIMS\seims\src\ccgl\bin>

```

Figure 2:1-6 Unit test results of CCGL (Common Cross-platform Geographic-computing Library)

If any **FAILED** tests occurred, you should check the settings of prerequisite software and libraries in Section 2:1.2 carefully. If you do not sure what the errors mean, please contact the developers for supports (see Section 1:1.3).

2:1.4 Installation of SEIMS

As stated earlier, SEIMS is mainly written by C++ and Python. Python is an interpreted language which means the source code can be executed directly under the Python environment without any manual compilation. Therefore, the installation of SEIMS is the compilation and installation of C++ applications.

The C++ applications of SEIMS not only include the main programs and modules for watershed modeling, but also the integrated programs for preprocessing such as watershed delineation by TauDEM (<http://hydrology.usu.edu/taudem/taudem5/index.html>) and static task scheduling with the graph of subbasins by METIS (<http://glaros.dtc.umn.edu/gkhome/metis/metis/overview>). All the C++ applications are organized by CMake and can be built, compiled, and installed at one time.

- Open a new VS2013 Developer Command Prompt, and navigate to the directory of SEIMS, e.g., D:\demo\SEIMS.
- Build the Visual Studio solution by CMake and compile the whole SEIMS solution by msbuild.exe.

```

01 >cd D:\demo\SEIMS
02 >d:
03 >mkdir build
04 >cd build
05 >cmake -G "Visual Studio 12 2013 Win64" ..
06 >msbuild.exe ALL_BUILD.vcxproj /p:Configuration=Release
07 >msbuild.exe INSTALL.vcxproj /p:Configuration=Release

```

- After the compilation and installation, all executables and libraries are installed at the default location, i.e., D:\demo\SEIMS\bin.

- The complete SEIMS includes C++ applications located in D:\demo\SEIMS\bin and Python utility tools located in D:\demo\SEIMS\seims such as preprocess, postprocess, parameters_sensitivity, calibration, and scenario_analysis (Figure 2:1-7).

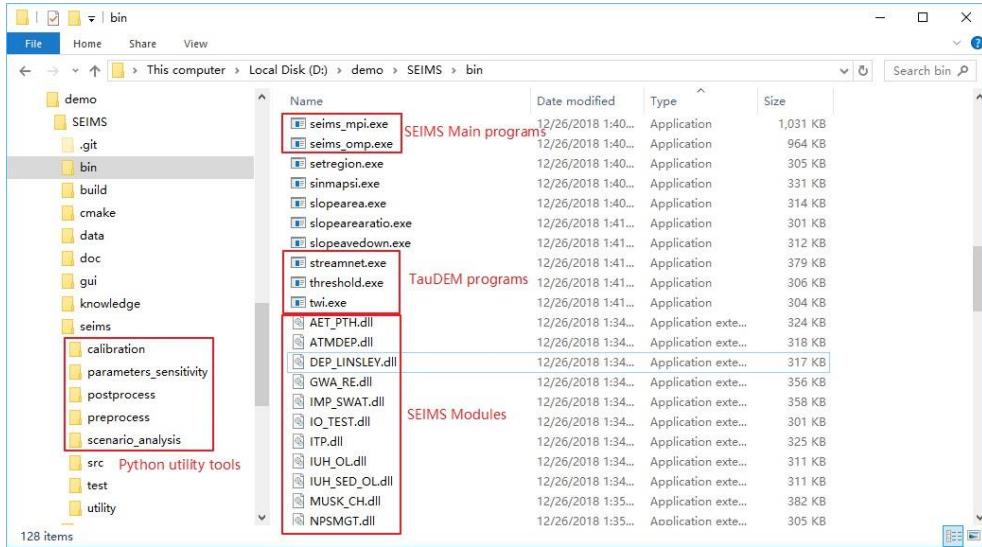


Figure 2:1-7 Directory tree of C++ applications and Python utility tools of SEIMS

- Enter D:\demo\SEIMS\bin\seims_omp.exe in a CMD window, the help information of SEIMS main program will be shown such as Figure 2:1-8. More details of running SEIMS will be introduced in “Section 2:4 Running a SEIMS-based watershed model”.

```

Command Prompt
-seali <calibrationID> -id <subbasinID>

D:\demo\SEIMS\bin>seims_omp
FAILURE: To run the program, use either the Simple Usage option or the Complete Usage option as below.
Simple Usage:
    seims_omp <modelPath> [<threadsNum> <layeringMethod> <IP> <port> <scenarioID> <calibrationID> <subbasinID>]
    <modelPath> is the path of the SBIMS-based watershed model.
    <threadsNum> is the number of thread used by OpenMP, which must be greater or equal than 1 (default).
    <layeringMethod> can be 0 and 1, which means UP_DOWN (default) and DOWN_UP, respectively.
    <IP> is the address of MongoDB database, and <port> is its port number.
    By default, MongoDB IP is 127.0.0.1 (i.e., localhost), and the port is 27017.
    <scenarioID> is the ID of BMPs Scenario which has been defined in BMPs database.
    By default, the Scenario ID is -1, which means not used.
    <calibrationID> is the ID of Calibration which has been defined in PARAMETERS table.
    By default, the Calibration ID is -1, which means not used.
    <subbasinID> is the subbasin that will be executed. 0 means the whole watershed. 9999 is reserved for Field version.

Complete and recommended Usage:
    seims_omp -wp <modelPath> [-thread <threadsNum> -1yr <layeringMethod> -host <IP> -port <port> -sce <scenarioID>
    -cali <calibrationID> -id <subbasinID>]

D:\demo\SEIMS\bin>
```

Figure 2:1-8 Usage of the OpenMP version of SEIMS main program

2:2. Data preparation of demo watershed

In simple terms, watershed modeling is to simulate the behavior of watershed such as runoff, soil erosion, and nutrient transfer using the empirical or physical formulas based on the geographic information data (e.g., digital elevation model [DEM], landuse map, and soil map), meteorological data, and management data, etc. Thus, data collection and preparation is the first key step for watershed modeling.

As a demo, data from the Youwuzhen watershed, Changting County, Fujian province, China is selected for long-term (i.e., daily time-step) simulation.

2:2.1 Basic conventions of file formats

Basically, the input data of SEIMS-based watershed model includes two categories such as spatial data and plain text.

2:2.1.1 Spatial data

Spatial data includes raster data and vector data. Theoretically, all formats of raster (https://www.gdal.org/formats_list.html) and vector (https://www.gdal.org/ogr_formats.html) data supported by GDAL are acceptable for SEIMS. Even though, the most commonly used [GeoTIFF](#) and [ESRI Shapefile](#) are highly recommended for raster and vector data, respectively.

- Please make sure all spatial data have the same projected coordinate system, **NOT** geographic coordinate system.
- The spatial extents of different raster data are not necessarily the same. However, the gridded cells at the same location are preferably coincident. Otherwise, the raster data will be interpolated based on the DEM data which may cause undesired distortion.

2:2.1.2 Plain text

Except for the spatial data, almost all the other data can be provided as plain text. The basic conventions of plain text are designed as:

- The line starts with number sign (#) will be regarded as a comment line and ignored by SEIMS. However, there is one exception when the first line of one plain text file is for recording the time-system and timezone, e.g., #LOCALTIME 8 means date time recorded in the current file is east 8th district time and #UTCTIME means Coordinated Universal Time which is also known as Greenwich Mean Time (GMT).
- The first valid line is headers if needed.
- Comma (',') is the delimiter for values within each data line.
- En dash ('-') is the primary delimiter within each value while colon (':') is the secondary delimiter.

For example, the following plain text

```

01 # This line is a comment.
02 SUBSCENARIO,NAME,LANDUSE,PARAMETERS
03 1, Closing measures,7-16,Interc_max:Maximum Interception
Capacity:AC:1-Conductivity:Soil hydraulic conductivity:RC:3.5

```

can be parsed as a Python data structure of dictionary like:

```

01 demo_dict = {
02     'SUBSCENARIO': 1,
03     'NAME': 'Closing measures'
04     'LANDUSE': [7, 16],
05     'PARAMETERS': [
06         ['Interc_max', 'Maximum Interception Capacity', 'AC', 1],
07         ['Conductivity', 'Soil hydraulic conductivity', 'RC', 3.5]
08     ]
09 }

```

2:2.2 Spatial data

The demo data named Youwuzhen watershed ($\sim 5.39 \text{ km}^2$) is located in Changting County of Fujian province, China (Figure 2:2-1). It belongs to the typical red-soil hilly region in southeastern China and suffers from severe soil erosion. The study area has hills with steep slopes (up to 52.9° and with an average slope of 16.8°) and broad alluvial valleys. The elevation ranges from 295.0 m to 556.5 m. The landuse types are mainly forest (59.8%), paddy field (20.6%), and orchard (12.8%) (Figure 2:2-2). Soil types in the study area are red soil (78.4%) and paddy soil (21.6%) which can be classified as *Ultisols* and *Inceptisols* in US Soil Taxonomy, respectively (Figure 2:2-3).

In order to improve the computational efficiency for demonstration in this manual, the DEM (`ywzdem30m.tif`), landuse (`ywzlanduse30m.tif`), and soil (`ywzsoil30m.tif`) map are all unified to be of 30 m resolution.

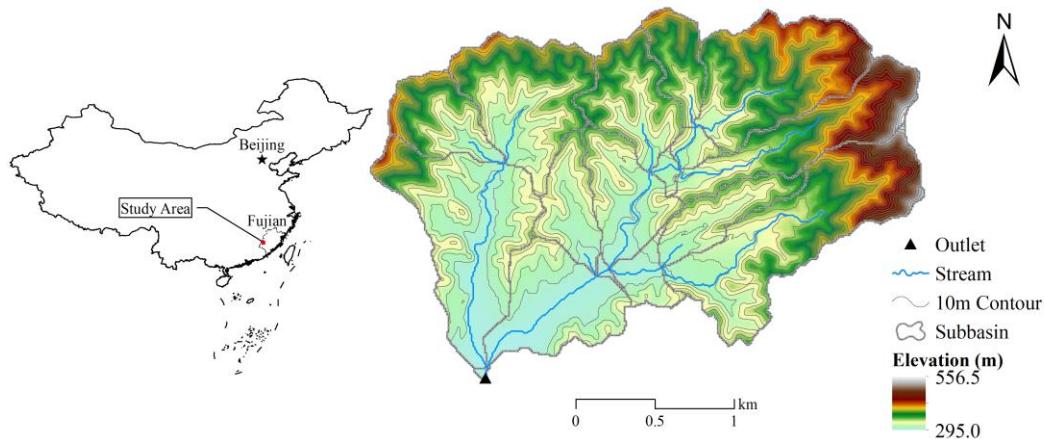


Figure 2:2-1 Location of the demo watershed named Youwuzhen watershed

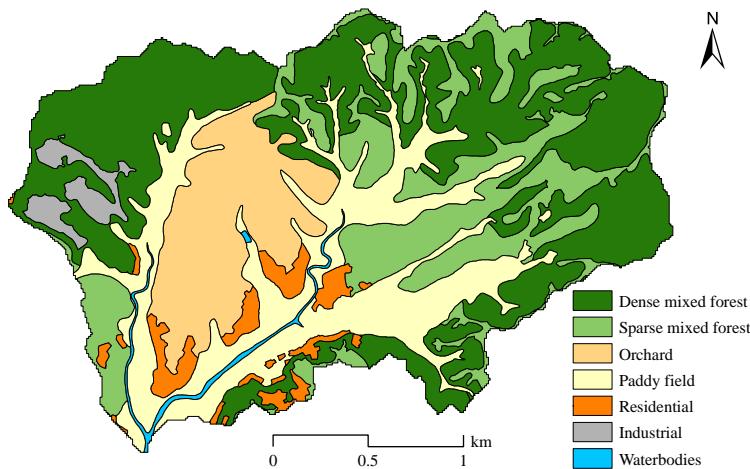


Figure 2:2-2 Landuse map of the Youwuzhen watershed

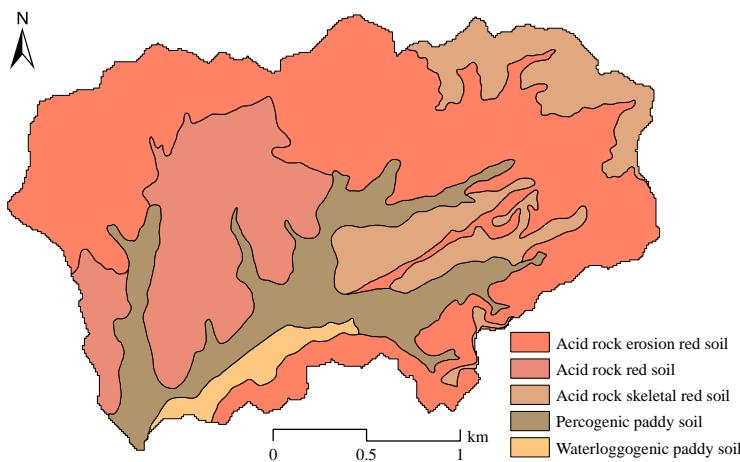


Figure 2:2-3 Soil map of the Youwuzhen watershed

The outlet location (i.e., as vector point) of the Youwuzhen watershed is prepared as ESRI Shapefile. If the outlet data cannot be predetermined, the location (i.e., center of the gridded cell) with largest flow accumulation will be marked as the outlet of the current study area.

In the current version of SEIMS, the Thiessen polygon of meteorological stations and precipitation stations that covers the entire watershed should also be provided as vector polygon data, e.g., `thiessen_meteo.shp` and `thiessen_pcp.shp`, respectively. The attributes of each polygon should include the unique ID (ID) which is coincident with station ID introduced in the following section (2:2.3.1), station name (Name), X and Y coordinates under the projected coordinate system (LocalX and LocalY), latitude and longitude under the WGS84 coordinate system (Lat and Lon), and altitude (Elevation).

TODO: The requirements of the Thiessen polygon of meteorological stations and precipitation stations should be removed since the spatial information of these stations presented as plain text (see Section 2.2.3.1) can provide similar information.

All these spatial data are located in `SEIMS\data\youwuzhen\data_prepare\spatial`.

2.2.3 Precipitation data

The annual average precipitation of the Youwuzhen watershed is 1697.0 mm and intense short-duration thunderstorm events contribute about three-quarters of annual precipitation from March to August.

2.2.3.1 Spatial information of precipitation station

The fields of spatial information of precipitation station are shown in Table 2:2-1.

Table 2:2-1 Fields of precipitation station

Field name	Datatype	Description
StationID	Integer	Unique station ID
Name	String	Station name
LocalX	Float	X coordinates (unit: m) under the projected coordinate system
LocalY	Float	Y coordinates (unit: m)
Lat	Float	Latitude (unit: degree) under the WGS84 coordinate system
Lon	Float	Longitude (unit: degree)
Elevation	Float	Altitude (unit: m)

For example, `SEIMS\data\youwuzhen\data_prepare\climate\Sites_P.csv`,

`StationID,Name,LocalX,LocalY,Lon,Lat,Elevation`

`81502750,HeTianZhan,39444759.232,2840563.152,116.4,25.683333,298`

2.2.3.2 Records of precipitation data

The first line is to state the time-system and timezone (see Section 2.2.1.2). If not provided, `#UTCTIME` will be regarded as the default.

The fields and formats of precipitation data are shown in Table 2:2-2.

Table 2:2-2 Fields and formats of precipitation data item

Field name	Datatype	Description
DATETIME	Datetime string	Date time with the format of <code>YYYY-MM-DD HH:MM:SS</code>
StationIDs	Float	Precipitation value for each Station IDs. Multiple stations can be <code>StationID1,StationID2, ... StationIDN</code>

Thus, the records of precipitation data is something like
 (SEIMS\data\youwuzhen\data_prepare\climate\ pcp_daily.csv):

```
#UTCTIME
DATETIME,81502750
2012-01-01 00:00:00,0
2012-01-02 00:00:00,0
2012-01-03 00:00:00,9.00
2012-01-04 00:00:00,16.50
2012-01-05 00:00:00,16.00
2012-01-06 00:00:00,1.50
```

2:2.4 Meteorological data

The Youwuzhen watershed is characterized by a mid-subtropical monsoon moist climate and has an annual average temperature of 18.3 °C.

The format of meteorological station is the same as that of precipitation station (see Section 2:2.3.1), e.g., SEIMS\data\youwuzhen\data_prepare\climate\Sites_M.csv.

Same to precipitation data, the first line of meteorological data text is to state the time-system and timezone (see Section 2:2.1.2). If not provided, #UTCTIME will be regarded as default. The fields and formats of meteorological data are shown in Table 2:2-3.

Note that there are no fixed order of these fields.

Table 2:2-3 Fields and formats of meteorological data item

Field name	Datatype	Description
StationID	Integer	Station ID
DATETIME	Datetime string	Date time with the format of YYYY-MM-DD HH:MM:SS
TMAX	Float	Maximum temperature (unit: degC)
TMIN	Float	Minimum temperature (unit: degC)
TMEAN	Float	(Optional) Mean temperature (unit: degC)
RM	Float	Relative moisture (unit: %)
WS	Float	Wind speed (unit: m/s)
SR or SSD	Float	Solar radiation (units: MJ/m ² /day) or sunshine duration hour (unit: hr)
PET	Float	Potential evapotranspiration (mm)

Thus, the records of meteorological data is something like
 (SEIMS\data\youwuzhen\data_prepare\climate\ meteo_daily.csv):

```
#LOCALTIME 8
StationID,DATETIME,TMEAN,TMAX,TMIN,RM,WS,SSD
58911,2012-01-01 20:00:00,10.0 ,13.6 ,7.8 ,76 ,1.2 ,1.2
58911,2012-01-02 20:00:00,10.6 ,15.7 ,7.0 ,73 ,0.7 ,1.5
58911,2012-01-03 20:00:00,7.1 ,12.0 ,4.6 ,89 ,1.6 ,0.0
58911,2012-01-04 20:00:00,3.9 ,6.6 ,2.5 ,78 ,1.7 ,0.0
```

Besides, the units of each type of data should also be provided, e.g.,
SEIMS\data\youwuzhen\data_prepare\climate\Variables.csv.

TODO: In the current version of SEIMS, the units should be the same as Table 2:2-3. However, some unit convert functions should be added in the future to make SEIMS more compatible with common used units.

2:2.5 Observed data

The periodic site-monitoring streamflow, sediment, or nutrient data collected within the watershed are regarded as observed data. The observed data is organized as one site information file and several data files corresponding to the number of monitoring sites and monitoring variables.

2:2.5.1 Spatial information of monitoring sites

The fields of spatial information of monitoring sites are shown in Table 2:2-4.

Table 2:2-4 Fields of observed data

Field name	Datatype	Description
StationID	Integer	Unique station ID
Name	String	Station name
Type	String	Monitoring variables, use En dash as separator for multiple variables. Avoid En dash in each single Type.
Unit	String	Units of monitoring variables, use En dash as separator for multiple units. Avoid En dash in each single Unit.
LocalX	Float	X coordinates (unit: m) under the projected coordinate system
LocalY	Float	Y coordinates (unit: m)
Lat	Float	Latitude (unit: degree) under the WGS84 coordinate system
Lon	Float	Longitude (unit: degree)
Elevation	Float	Altitude (unit: m)
isOutlet	Integer	Types of monitoring site: 1: The outlet of watershed 2: The outlet of one subbasin 3: The junction of multiple subbasins 0: Other spatial location

For example, `SEIMS\data\youwuzhen\data_prepare\observed\SiteInfo.csv`:

```
StationID,Name,Type,Lat,Lon,LocalX,LocalY,Unit,Elevation,isOutlet
1,hetianzhan,SED,25.680207,116.406401,440409.511725,2841541.17804,g/L,280,1
1,hetianzhan,Q,25.680207,116.406401,440409.511725,2841541.17804,m3/s,280,1
```

Note that the observed data is primarily used for postprocessing such as matching to the corresponding simulated values. Thus, the `Type` should be accord with the output of SEIMS-based watershed model. All the currently available outputs can be found in `SEIMS\seims\preprocess\database\AvailableOutputs.csv`. For example, if the total nitrogen data is monitored, the type should be `CH_TN` according to the value of the `FILENAME` field (obviously, not include the suffix, e.g., `'.txt'`) in the output item of total nitrogen amount in reach:

```
MODULE_CLASS,OUTPUTID,DESCRIPTION,UNIT,TYPE,STARTTIME,ENDTIME,INTERVAL,INTERVAL_UNIT,SUBBASIN,FILENAME,USE
NutrientTransport,CH_TN,total nitrogen amount in reach,kg,NONE,1970-01-01 00:00:00,1970-01-01 00:00:00,-9999,-9999,ALL,CH_TN.txt,0
```

2:2.5.2 Records of observed data

Similar to the format of precipitation data, the first line is optionally to state the time-system and timezone (see Section 2:2.1.2). If not provided, `#UTCTIME` will be regarded as the default.

The fields and formats of observed data are shown in Table 2:2-5.

Table 2:2-5 Fields and formats of observed data item

Field name	Datatype	Description
StationID	Integer	Station ID
DATETIME	Datetime	Date time with the format of YYYY-MM-DD HH:MM:SS
	string	
Type	String	Monitoring variable
VALUE	Float	Monitoring value of current variable at the current date time

Thus, the records of observed data is something like
(`SEIMS\data\youwuzhen\data_prepare\observed\observed_Q.csv`):

```
#UTCTIME
StationID,DATETIME,Type,VALUE
1,2012-01-14 00:00:00,Q,0.1615
1,2012-01-15 00:00:00,Q,0.578
1,2012-01-16 00:00:00,Q,0.4317
```

2:2.6 Lookup tables

Lookup tables, including crop, fertilizer, tillage, lanuse, soil, and urban, are adopted from SWAT and predefined in `SEIMS\seims\preprocess\database`. Parameters specific to study areas can be appended to these lookup tables or prepared in separate files in `SEIMS\data\youwuzhen\data_prepare\lookup`.

The details of the most common used lookup tables are as follows.

2:2.6.1 Soil properties

Soil properties include physical properties and chemical properties. The fields and descriptions are shown in Table 2:2-6. The optional parameters can be omitted.

Note that the `SEQN` and `NAME` may not consistent with soil types (`SOILCODE`), so that to represent heterogeneity of the same soil type according to different landcover or topographic positions. However, the `SEQN` MUST be consistent with the values in soil map, i.e., `ywzsoil30m.tif`. The soil properties of multiple layers are concatenated with En dash ('-') as described in Section 2:2.1.2.

Table 2:2-6 Fields and description in lookup table of soil properties

Field name	Datatype	Unit	Description
<code>SEQN</code> or <code>SOILCODE</code>	Integer	-	Unique identifier of soil map
<code>NAME</code>	String	-	Soil name
<code>SOILLAYERS</code>	Integer	-	Number of soil layers
<code>SOL_Z</code>	Float array	mm	Depth from soil surface to bottom of layer
<code>SOL_OM</code>	Float array	%	Organic matter content
<code>SOL_CLAY</code>	Float array	%	Clay content, d < 0.002 mm
<code>SOL_SILT</code>	Float array	%	Silt content, 0.002 mm < d < 0.05 mm
<code>SOL SAND</code>	Float array	%	Sand content, 0.05 mm < d < 2 mm
<code>SOL_ROCK</code>	Float array	%	Rock fragment content, d > 2 mm
<code>SOL_BD</code>	Float array	Mg/m3	Moist bulk density, value ranges 1.1 ~ 1.9
<code>SOL_AWC</code>	Float array	mm	Available water capacity
<code>SOL_ZMX</code>	Float	mm	(Optional) Maximum rooting depth of soil profile
<code>ANION_EXCL</code>	Float	-	(optional) Fraction of porosity (void space) from which anions are excluded, default is 0.5

Field name	Datatype	Unit	Description
SOL_CRK	Float	-	(optional) Potential or maximum crack volume of the soil profile expressed as a fraction of the total soil volume.
SOL_K	Float array	mm/hr	(optional) Saturated hydraulic conductivity
SOL_WP	Float array	mm	(optional) Wilting point
SOL_FC	Float array	mm	(optional) Field capacity
SOL_POROSITY	Float array	-	(optional) Porosity
SOL_USLE_K	Float array	-	(optional) USLE K factor
SOL_ALB	Float	-	(optional) Albedo when soil is moist
ESCO	Float	-	(optional) Soil evaporation compensation factor, the default is 0.95
SOL_N03	Float array	g/kg	(optional) concentration of nitrate
SOL_NH4	Float array	g/kg	(optional) concentration of ammonium-N in soil
SOL_ORGN	Float array	g/kg	(optional) concentration of organic nitrogen
SOL_ORGP	Float array	g/kg	(optional) concentration of organic phosphorus
SOL_SOLP	Float array	g/kg	(optional) concentration of soluble phosphorus

Thus, the lookup table of soil properties of specific study area is something like ([SEIMS\data\youwuzhen\data_prepare\lookup\soil_properties_lookup.csv](#)):

```
SEQN,SNAM,SOILLAYERS,SOL_ZMX,SOL_Z,SOL_BD,SOL_OM,SOL_CLAY,SOL_SILT,SOL_SAND,S  
OL_ROCK,SOL_WP,SOL_FC,SOL_POROSITY,SOL_K,SOL_AWC,SOL_N03,SOL_NH4,SOL_ORGN,SOL  
_SOLP,SOL_ORGP  
201,WNT,3,600,200-400-600,1.5-1.57-1.45,2.31-0.84-0.84,15.66-17.36-  
20.94,13.8-17.31-22.23,52.25-44.6-35.9,18.29-20.73-20.93,0.12-0.14-0.18,0.21-  
0.24-0.31,0.44-0.41-0.45,26.16-11.43-7.87,0.1-0.1-0.13,0.004-0.002-0.002,0-0-  
0,0.164-0.079-0.077,0.005-0.002-0.001,0.047-0.018-0.012
```

2:2.6.2 Initial landcover parameters

Some parameters of landcover at the beginning of simulation should be defined. The fields and descriptions are shown in Table 2:2-7.

Table 2:2-7 Fields and descriptions in the lookup table of initial landcover parameters

Field name	Datatype	Unit	Description
LANDUSE_ID	Integer	-	Landuse ID
IGRO	Integer	-	Land cover status: 0-none growing; 1-growing
LANDCOVER	Integer	-	ICNUM, Land cover ID number (required when IGRO

Field name	Datatype	Unit	Description
or ICNUM			is 1)
LAI_INIT	Float	-	Initial leaf area index (required when IGRO is 1)
BIO_INIT	Float	kg/ha	Initial biomass (required when IGRO is 1)
PHU_PLT	Float	degC	Number of heat units to bring plant to maturity (required when IGRO is 1)
EPCO	Float	-	Plant uptake compensation factor, 0.01 ~ 1
RSDIN	Float	kg/ha	Initial residue cover
CURYR_INIT	Float	year	Initial age of trees
CHT	Float	m	Initial canopy height
DORMI	Float	-	Dormancy status code: 1 - growing and 0 - dormancy
USLE_P	Float	-	Conservation practice management factor of USLE

Thus, the lookup table of initial landcover parameters of specific study area is something like

(SEIMS\data\youwuzhen\data_prepare\lookup\landcover_initial_parameters.csv):

```
LANDUSE_ID,IGRO,LANDCOVER,LAI_INIT,BIO_INIT,PHU_PLT,EPCO,RSDIN,
CURYR_INIT,CHT,DORMI,USLE_P
33,0,33,0,0,0,1,100,0,0,0,0.084
4,1,4,2,200,0,1,200,2,2,0,0.8
8,1,8,3,1000,0,1,300,5,5,0,0.8
6,1,6,2,600,0,1,200,3,2,0,0.8
18,0,18,0,0,0,0,0,0,0,0,1
104,0,-9999,0,0,0,0,0,0,0,0,1
106,0,-9999,0,0,0,0,0,0,0,0,1
```

2.2.7 Management practices data

There are three different types of Best Management Practices (BMPs) supported or will be supported by SEIMS.

- *Reach BMPs: BMPs that attached to specific reaches and will change the characters of the reach, such as point source, stream flow diversion, reservoir, riparian wetland, and riparian buffer, etc.*
- *Areal structural BMPs: BMPs that are corresponding to a specific structure in the watershed and will change the characters of the specified locations, such as grass waterway, filter strip, pond, isolated wetland, terrace, overland flow diversion, tile drain management, and urban management, etc.*
- *Areal non-structure BMPs: BMPs that are NOT corresponding to a specific structure in the watershed and will change the characters of the specified locations, such as plant management.*

In this section, the organization of BMP scenario is firstly presented, then followed by the detail parameter settings of different BMPs. In the current version of this user manual, plant management and general areal structural BMP are introduced as an example. More BMPs should be updated in the future version.

2:2.7.1 BMP scenario

A BMP scenario is a collection of different BMPs which will be applied to an SEIMS-based watershed model to affect watershed behaviors.

There can be many different BMP scenarios for the BMP scenarios analysis based on one watershed model. Each BMP scenario is identified using a unique integer ID. For each BMP of one scenario, the location and parameters must be specified. For reach BMPs, the location is the reach ID. For two areal BMPs, the location is the areas identified by a raster data, i.e., the so-called BMP configuration units. The BMP parameters are defined in separate plain text files, in which different combinations of parameters are allowed and distinguished by unique ID, i.e., SUBSCENARIO. The fields and descriptions of BMP scenario table are shown in Table 2:2-8.

Table 2:2-8 Fields and descriptions of BMP scenario table

Field name	Datatype	Description
ID	Integer	Unique ID of BMP scenario
NAME	String	Scenario name
BMPID	Integer	Predefined BMP ID (e.g., SEIMS\data\youwuzhen\scenario\BMP_index.csv)
SUBSCENARIO	Integer	Sub-scenario ID of BMP defined in specific BMP parameters
DISTRIBUTION	String	The format is <Type> <Filename>. <Type> may be REACH or RASTER for reach BMP and areal BMP, respectively. <Filename> is the corresponding reach table name or raster filename. For example, RASTER LANDUSE means the configuration of the current areal BMP is based on landuse types.
COLLECTION	String	The name of the plain text file that defines the BMP parameters (e.g., SEIMS\data\youwuzhen\scenario\plant_management.csv)
LOCATION	Integer array	Location values of DISTRIBUTION for configuring BMP, e.g., 33 means the BMP will be configured on the landuse use type of 33. Multiple location values are separated by En dash.

Thus, the BMP scenarios table of specific study area is something like
(SEIMS\data\youwuzhen\data_prepare\scenario\BMP_scenarios.csv):

```
ID,NAME,BMPID,SUBSCENARIO,DISTRIBUTION,COLLECTION,LOCATION
0,base,12,0,RASTER|LANDUSE,plant_management,33
```

The scenario ID of 0 named base includes one BMP with the BMPID of 12 which is plant management according to BMP_index.csv. The configuration unit of this BMP is based on LANDUSE map and the landuse of 33 will be configured. The parameters of this BMP can be loaded from plant_management.csv and the SUBSCENARIO of 0 will be applied.

2.2.7.2 Plant management

By drawing lessons from the SWAT model, a total of 15 different types of plant management operations are considered in the BMP module of SEIMS. A combination of several plant management operations (e.g., plant, fertilize, harvest, etc.) is regarded as a SUBSCENARIO of plant management practices such as the crop rotation practices with rice and winter wheat. Each management operation of one SUBSCENARIO has the same fields such as Table 2:2-9.

Plant management practices are scheduled according to heat units and/or operation date from local experiences. The operation can be performed if either of the conditions is met. For example, the plant operation is set at 5th, May or HUSC greater or equal to 0.2. If the BASE_HU is used and the HUSC which represents the fraction of annual total heat units (HU) has reached 0.2 at 1st, May, then the plant operation will occur since the HUSC condition is first met than MONTH/DAY.

The type of plant management operation simulated is identified by the code given for the field MGT_OP. The different codes for MGT_OP are:

1. *Plant/beginning of growing season. Initializes the growth of a specific landcover/plant type.*
2. *Irrigation operation. Applies water to the location.*
3. *Fertilizer application. Adds nutrients to the soil.*
4. *Pesticide application. Applies a pesticide to the plant and/or soil.*
5. *Harvest and kill operation. Harvest the portion of the plant designated as yield, removes the yield from the location and converts the remaining plant biomass to residue on the soil surface.*
6. *Tillage operation. Mix the upper soil layers and redistributes the nutrients/chemicals/etc. within those layers.*
7. *Harvest only operation. Harvest the portion of the plant designated as yield and removes the yield from the location, but allows the plant to continue growing. This operation is used for hay cuttings.*
8. *Kill/ending of growing season. Stop all plant growth and covert all plant biomass to residue.*
9. *Grazing operation. Remove plant biomass and allow simultaneous application of manure.*

10. *Auto irrigation initialization.* Initialize auto irrigation within the location. This operation applies water whenever the plant experiences a user-specified level of water stress.
11. *Auto fertilization initialization.* Initialize auto fertilization within the location. This operation applies nutrients whenever the plant experiences a user-specified level of nitrogen stress.
12. *Street sweeping operation.* Remove sediment and nutrient build-up on impervious areas in the location. This operation can only be used when the urban build up/wash off routines are activated for the location.
13. *Release/impound.* Release or impound water for rice or other plants.
14. *Continuous fertilization.* Apply fertilizer/manure to the soil surface on a continuous basis.
15. *Continuous pesticide.* Apply pesticides to the soil surface on a continuous basis
16. *Burning operation.* Remove the user-specified portion of plant biomass from the location.

The parameters of different plant management operations are listed in Table 2:2-10. Please refers to SWAT 2012 Input/Output Documentation¹ for more details of each parameters.

Note that, in order to simulate the water level changes in different growth stages of paddy rice, the parameters of the release/impound operation are expanded than SWAT, i.e., maximum ponding depth (**MAX_PND**), minimum fitting depth (**MIN_FIT**), and maximum fitting depth (**MAX_FIT**).

Table 2:2-9 Fields and descriptions of plant management practices

Field name	Datatype	Description
SUBSCENARIO	Integer	Unique sub-scenario ID in current BMP parameters table
NAME	String	Sub-Scenario name, e.g., Crop_rotation
LANDUSE_ID	Integer	Landuse type that the SUBSCENARIO can be applied
SEQUENCE	Integer	Sequence No. of the plant management related operations of the SUBSCENARIO, start from 1
YEAR	Integer	Year No. of the SUBSCENARIO, start from 1
MONTH	Integer	Month of the operation takes place
DAY	Integer	Day of the operation takes place
BASE_HU	Boolean	Use (1) the fraction of annual total heat units (HU) or the

¹ Arnold, J.G.; Kiniry, J.R.; Srinivasan, R.; Williams, J.R.; Haney, E.B.; Neitsch, S.L. Soil and Water Assessment Tool 2012 Input/Output Documentation; Texas Water Resources Institute, 2012, p257-296.

Field name	Datatype	Description
		fraction of total heat units of a plant to reach maturity (PHU) (0)
HUSC	Float	Heat unit scheduling for operation expressed as the fraction of PHU or the fraction of HU if BASE_HU=1
MGT_OP	Integer	Management operation number
MGT<N>	Float array	Operation related parameters. <N> ranges from 1 to 10. Multiple location values are separated by Comma.

Table 2:2-10 Parameters defined for plant management operations

NAME	OP	MGT1	MGT2	MGT3	MGT4	MGT5	MGT6	MGT7	MGT8	MGT9	MGT10
Plant	1	PLANT_ID	NONE	CURYR_MAT	HEAT_UNITS	LAI_INIT	BIO_INIT	HI_TARG	BIO_TARG	CNOP	NONE
Irrigate	2	NONE	IRR_SC	NONE	IRR_AMT	IRR_SALT	IRR_EFM	IRR_SQ	NONE	NONE	IRR_NO
Fertilize	3	FERT_ID	NONE	NONE	FRT_KG	FRT_SURFACE	NONE	NONE	NONE	NONE	NONE
Pesticide	4	PEST_ID	NONE	NONE	PST_KG	PST_DEP	NONE	NONE	NONE	NONE	NONE
Harvest & Kill	5	NONE	NONE	NONE	CNOP	HI_OVR	FRAC_HA_RVK	NONE	NONE	NONE	NONE
Tillage	6	TILL_ID	NONE	NONE	CNOP	NONE	NONE	NONE	NONE	NONE	NONE
Harvest Only	7	NONE	NONE	NONE	HARVEFF	HI_BMS	HI_RSD	NONE	NONE	NONE	NONE
Kill	8	NONE	NONE	NONE	NONE	NONE	NONE	NONE	NONE	NONE	NONE
Grazing	9	GRZ_DAYS	MANURE_ID	NONE	BIO_EAT	BIO_TRM_P	MANURE_KG	NONE	NONE	NONE	NONE
Auto Irrigation	10	WSTRS_ID	IRR_SCA	NONE	AUTO_WSTRS	IRR_EFF	IRR_MX	IRR_ASQ	NONE	NONE	IRR_NOA
Auto Fertilize	11	AFERT_ID	NSTRESS	NONE	AUTO_NSTRS	AUTO_NAPP	AUTO_NYR	AUTO_EF	AFRT_SURFACE	NONE	NONE
Sweep	12	NONE	NONE	NONE	SWEEPEFF	FR_CURB	NONE	NONE	NONE	NONE	NONE
Release/Impound	13	IMP_TRIG	MAX_PND	MIN_FIT	MAX_FIT	NONE	NONE	NONE	NONE	NONE	NONE
Cont. Fertilize	14	FERT_DAYS	CFRT_ID	IFRT_FREQ	CFRT_KG	NONE	NONE	NONE	NONE	NONE	NONE
Cont. Pesticide	15	IPST_ID	PEST_DAYS	IPST_FREQ	CPST_KG	NONE	NONE	NONE	NONE	NONE	NONE
Burning	16	NONE	NONE	NONE	BURN_FRLB	NONE	NONE	NONE	NONE	NONE	NONE

Note: **NONE** means the reserved position for further potential parameters and the default value is 0.

2.2.7.3 General areal structural BMP

Generally, the areal structural BMP takes effects by modifying watershed modeling related parameters on the locations that configured with BMP. Thus, a general table for areal structural BMP is designed as shown in Table 2.2-11, which basically includes spatial parameters (e.g., suitable LANDUSE and SLPPOS), environmental effectiveness parameters (e.g., PARAMETERS and EFFECTIVENESS), and economic benefits (e.g., CAPEX, OPEX, and INCOME).

Note that, fields of the general areal structural BMP can be extended or modified by users according to the requirements of scenario analysis. These fields will be used in the predefined scenario analysis program (e.g., SEIMS\seims\scenario_analysis\spatialunits) or other scenario analysis program developed based on the base class in SEIMS (SEIMS\seims\scenario_analysis).

Table 2.2-11 Fields and descriptions of general areal structural management practices

Field name	Datatype	Description
SUBSCENARIO	Integer	Unique sub-scenario ID in current BMP parameters table
NAME	String	Sub-Scenario name, e.g., closing measures (CM)
DESC	String	Description of BMP
REFERENCE	String	Literature reference
LANDUSE	String	Suitable landuse types, default is 'ALL'. Multiple landuse types are concatenated by En dash, e.g., '6-8'.
SLPPOS	String	Suitable slope positions, default is 'ALL'. Multiple slope position tags are concatenated by En dash, e.g., '4-16'.
PARAMETERS	String	Spatial parameters that the BMP affects, the format MUST be: NAME1:DESC1:CHANGE1:IMPACT1- NAME2:DESC2:CHANGE2:IMPACT2- where, NAME is the parameter's ID, which MUST be one of the GridFS file in SPATIAL collection in MongoDB, DESC is the corresponding description, CHANGE is the change method (VC,RC, or AC. VC: replace, RC: multiply, and AC: add), IMPACT is the impact value.
EFFECTIVENESS	Integer	Overall environmental effectiveness (e.g., reducing soil erosion) grade, range from 1 to 5, with higher-numbered grades representing better effectiveness
CAPEX	Float	Initial implementation cost per km ²
OPEX	Float	Annual maintenance cost per km ²
INCOME	Float	Annual benefit per km ²

One of the general areal structural management practices of the Youwuzhen watershed prepared in

`SEIMS\data\youwuzhen\data_prepare\scenario\areal_struct_management.csv`
is as follows:

```
SUBSCENARIO,NAME,DESC,REFERENCE,LANDUSE,SLPPOS,PARAMETERS,EFFECTIVE  
NESS,CAPEX,OPEX,INCOME  
1,fengjin, CM (closing measures),Qin et al (2018),8-6,1-  
4,OM:Organic matter:RC:1.22-Density:bulk density:RC:0.98-  
Porosity:Total porosity:RC:1.02-USLE_K:USLE soil  
erodibility:RC:1.01-Conductivity:Soil hydraulic  
conductivity:RC:0.81-FieldCap:Soil field capacity:RC:1.02-  
Wiltingpoint:Wiltingpoint:RC:1.02-SOL_AWC:Soil available  
water:RC:1.02-SOL_UL:Soil saturated water:RC:1.02-SOL_CBN:Soil  
carbon content:RC:1.22-USLE_P:USLE practice  
factor:RC:0.9,3,15.5,1.5,2.0
```

which can be parsed as a Python data structure of dictionary like:

```
01 demo_dict = {'NAME': 'fengjin',  
02             'DESC': 'CM (closing measures)',  
03             'REFERENCE': 'Qin et al (2018)',  
04             'LANDUSE': [8, 6],  
05             'SLPPOS': [1, 4],  
06             'PARAMETERS': [  
07                 {'NAME': 'OM',  
08                     'DESC': 'ORGANIC MATTER',  
09                     'CHANGE': 'RC',  
10                     'IMPACT': 1.22  
11                 },  
12                 {'NAME': 'DENSITY',  
13                     'DESC': 'BULK DENSITY',  
14                     'CHANGE': 'RC',  
15                     'IMPACT': 0.98  
16                 }  
17             ],  
18             'EFFECTIVENESS': 3,  
19             'CAPEX': 15.5,  
20             'OPEX': 1.5,  
21             'INCOME': 2,  
22         }
```

2:3. Data preprocessing for watershed modeling

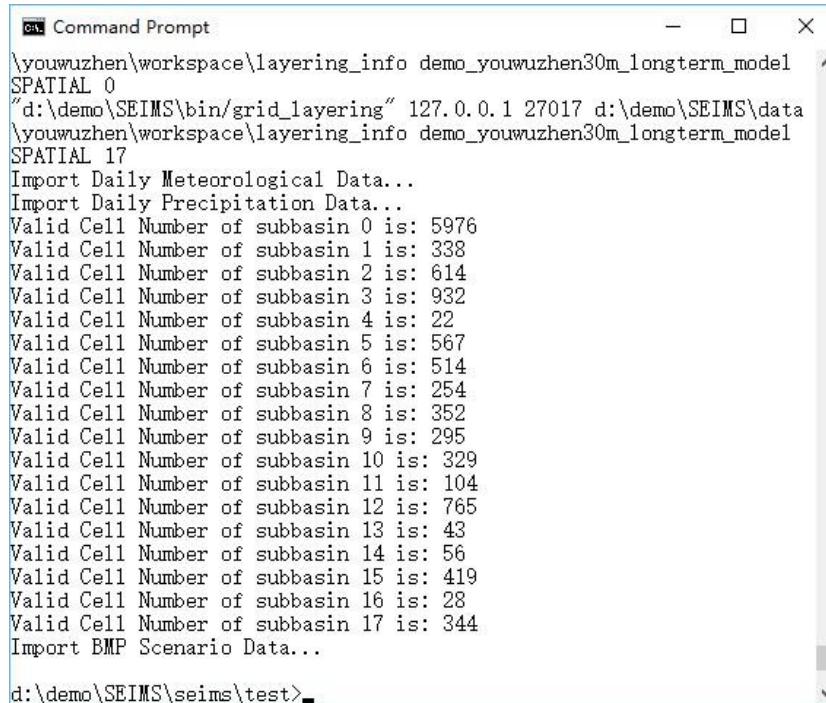
Data preprocessing for watershed modeling is a workflow to extract spatial parameters from various spatial data (e.g., DEM, landuse, and soil map), organize plaintext data (e.g., precipitation and meteorological data, site-monitoring data, and BMP scenarios data), and import these data into MongoDB database of the study area.

2:3.1 Simple usage

For simple usage, open a CMD window, enter the following commands to run the data preprocessing of the Youwuzhen watershed.

```
01 >cd D:\demo\SEIMS\seims\test
02 >D:
03 >python demo_preprocess.py -name youwuzhen
```

The end of runtime logs of the data preprocessing was shown in Figure 2:3-1.



```
Command Prompt
\youwuzhen\workspace\layering_info demo_youwuzhen30m_longterm_model
SPATIAL 0
"d:\demo\SEIMS\bin\grid_layering" 127.0.0.1 27017 d:\demo\SEIMS\data
\youwuzhen\workspace\layering_info demo_youwuzhen30m_longterm_model
SPATIAL 17
Import Daily Meteorological Data...
Import Daily Precipitation Data...
Valid Cell Number of subbasin 0 is: 5976
Valid Cell Number of subbasin 1 is: 338
Valid Cell Number of subbasin 2 is: 614
Valid Cell Number of subbasin 3 is: 932
Valid Cell Number of subbasin 4 is: 22
Valid Cell Number of subbasin 5 is: 567
Valid Cell Number of subbasin 6 is: 514
Valid Cell Number of subbasin 7 is: 254
Valid Cell Number of subbasin 8 is: 352
Valid Cell Number of subbasin 9 is: 295
Valid Cell Number of subbasin 10 is: 329
Valid Cell Number of subbasin 11 is: 104
Valid Cell Number of subbasin 12 is: 765
Valid Cell Number of subbasin 13 is: 43
Valid Cell Number of subbasin 14 is: 56
Valid Cell Number of subbasin 15 is: 419
Valid Cell Number of subbasin 16 is: 28
Valid Cell Number of subbasin 17 is: 344
Import BMP Scenario Data...
d:\demo\SEIMS\seims\test>
```

Figure 2:3-1 Runtime logs of data preprocessing for watershed modeling of the Youwuzhen watershed

After running the simple usage, the input configuration file for data preprocessing (`preprocess.ini`) is generated in the directory of intermediate data (i.e., `SEIMS\data\youwuzhen\workspace`, Figure 2:3-2) and the data for watershed modeling has been imported into MongoDB (Figure 2:3-3). The details of the configuration file, advanced usage, intermediate data, and watershed modeling databases will be introduced in the following sections.

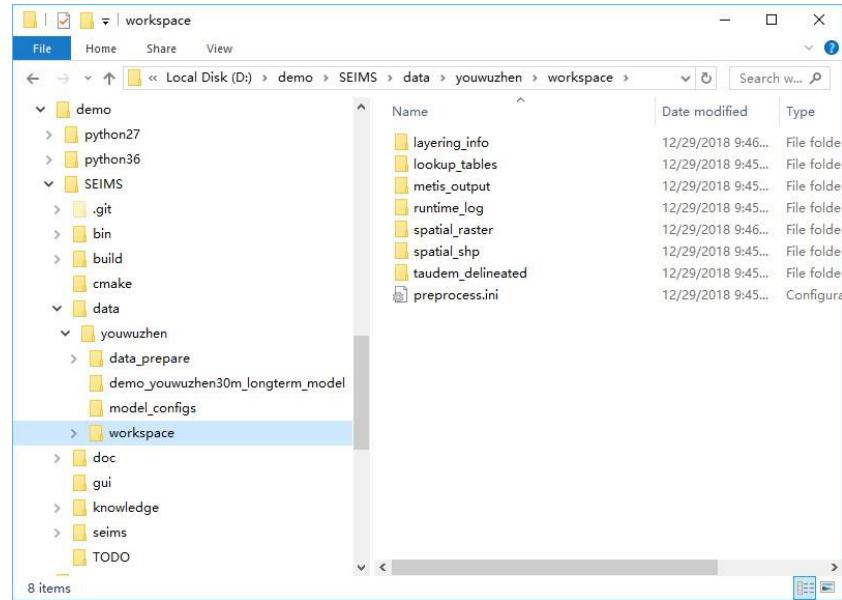


Figure 2:3-2 Directory tree of intermediate data of the Youwuzhen watershed after data preprocessing

The screenshot shows the Robo 3T interface connected to a MongoDB database. The left sidebar displays the database structure:

- demo_youwuzhen30m_HydroClimate** (Collection: 5)
 - ANNUAL_STATS
 - DATA_VALUES
 - MEASUREMENT
 - SITES
 - VARIABLES
- demo_youwuzhen30m_Scenario** (Collection: 4)
 - AREAL_STRUCT_MANAGEMENT
 - BMP_INDEX
 - BMP_SCENARIOS
 - PLANT_MANAGEMENT
- demo_youwuzhen30m_longterm_model** (Collection: 14)
 - BMPDATABASE
 - CROPOOKUP
 - FERTILIZERLOOKUP
 - FILE_IN
 - FILE_OUT
 - LANDUSELOOKUP
 - PARAMETERS
 - REACHES
 - SITELIST
 - SOILLOOKUP
 - SPATIALchunks
 - SPATIALfiles
 - TILLAGELOOKUP
 - URBANLOOKUP

The right panel shows the results of the query `db.getCollection('FILE_IN').find({})`:

_id	TAG	VALUE
1	MODE	Daily
2	INTERVAL	1
3	STARTTIME	2012-01-01 00:00:00
4	ENDTIME	2015-12-31 23:59:59

Figure 2:3-3 Screenshot of the watershed modeling databases of the Youwuzhen watershed

2.3.2 Configuration file of data preprocessing

Actually, the simple usage of the data preprocessing includes two steps such as generating the configuration file according to the local paths of SEIMS and executing data preprocessing by the advanced usage which will be introduced in the next section.

SEIMS takes the [INI file](#) as the format of configuration files. [INI](#) files are plain text files with a basic structure composed of sections and options (an option is a pair of property and value). Semicolons (';') or number sign ('#') at the beginning of the line indicate a comment which will be ignored. Figure 2.3-4 showed the data preprocessing configuration content of Youwuzhen watershed.

```
01 [PATH]
02 PREPROC_SCRIPT_DIR = d:\demo\SEIMS\seims\preprocess
03 CPP_PROGRAM_DIR = d:\demo\SEIMS\bin
04 MPIEXEC_DIR = None
05 BASE_DATA_DIR = d:\demo\SEIMS\data\youwuzhen
06 CLIMATE_DATA_DIR = d:\demo\SEIMS\data\youwuzhen\data_prepare\climate
07 SPATIAL_DATA_DIR = d:\demo\SEIMS\data\youwuzhen\data_prepare\spatial
08 MEASUREMENT_DATA_DIR = d:\demo\SEIMS\data\youwuzhen\data_prepare\observed
09 BMP_DATA_DIR = d:\demo\SEIMS\data\youwuzhen\data_prepare\scenario
10 MODEL_DIR = d:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model
11 TXT_DB_DIR = d:\demo\SEIMS\data\youwuzhen\data_prepare\lookup
12 WORKING_DIR = d:\demo\SEIMS\data\youwuzhen\workspace
13 [MONGODB]
14 HOSTNAME = 127.0.0.1
15 PORT = 27017
16 ClimateDBName = demo_youwuzhen30m_HydroClimate
17 BMPScenarioDBName = demo_youwuzhen30m_Scenario
18 SpatialDBName = demo_youwuzhen30m_longterm_model
19 [CLIMATE]
20 HydroClimateVarFile = Variables.csv
21 MeteoSiteFile = Sites_M.csv
22 PrecSiteFile = Sites_P.csv
23 MeteoDataFile = meteo_daily.csv
24 PrecDataFile = pcp_daily.csv
25 thiessenIdField = ID
26 [SPATIAL]
27 dem = ywzdem30m.tif
28 outlet_file = outlet_beijing1954.shp
29 PrecSitesThiessen = thiessen_pcp.shp
30 MeteoSitesThiessen = thiessen_meteo.shp
31 landuseFile = ywzlanduse30m.tif
32 landcoverInitFile = landcover_initial_parameters.csv
33 soilSEQNFile = ywzsoil30m.tif
34 soilSEQNText = soil_properties_lookup.csv
35 field_partition_thresh = 15
36 [OPTIONAL_PARAMETERS]
37 D8AccThreshold = 35
38 np = 4
39 D8DownMethod = Surface
40 dorm_hr = -1.
41 T_base = 0.
42 imperviousPercInUrbanCell = 0.3
43 defaultLanduse = 33
44 defaultSoil = 201
```

Figure 2:3-4 Configuration content for data preprocessing of the Youwuzhen watershed

The configuration file for data preprocessing, such as that of the Youwuzhen watershed shown in Figure 2:3-4, includes five sections, i.e., PATH, MONGODB, CLIMATE, SPATIAL, and OPTIONAL_PARAMETERS. Several property-value pairs (i.e., options) are included within each section. The names of sections and properties should not be changed. Note that section and property names are not case sensitive in the Windows implementation.

- **PATH:** *Full paths of executables, data, and intermediate workspace.*
 1. **PREPROC_SCRIPT_DIR:** *The path of Python scripts of data preprocessing.*
 2. **CPP_PROGRAM_DIR:** *The install directory of SEIMS C/C++ applications.*
 3. **MPIEXEC_DIR:** *The directory path of the MPI executable, i.e., C:\Program Files\Microsoft MPI\Bin. If it has been added to the environment path, the value can be None or commented.*
 4. **BASE_DATA_DIR:** *The base data directory of the study area.*
 5. **CLIMATE_DATA_DIR:** *(Optional) The path of climate data which include precipitation data (Section 2:2.3) and meteorological data (Section 2:2.4). If not specified (means the option line is commented), SEIMS will try to find climate data in BASE_DATA_DIR\data_prepare\climate.*
 6. **SPATIAL_DATA_DIR:** *(Optional) The path of spatial data (Section 2:2.1.1). If not specified, SEIMS will try to find spatial data in BASE_DATA_DIR\data_prepare\spatial.*
 7. **MEASUREMENT_DATA_DIR:** *(Optional) The path of observed data (Section 2:2.5). If not specified, SEIMS will try to find observed data in BASE_DATA_DIR\data_prepare\observed.*
 8. **BMP_DATA_DIR:** *(Optional) The path of BMP scenarios data (Section 2:2.7). If not specified, SEIMS will try to find scenario data in BASE_DATA_DIR\data_prepare\scenario.*
 9. **MODEL_DIR:** *The model path of the study area which includes several configuration files for the watershed modeling. For data preprocessing, the file.in and file.out are required, the calibrated parameters param.cal is optional. Details of the file structure of model folder will be introduced in Section 2:4.2.*
 10. **TXT_DB_DIR:** *(Optional) The path of additional lookup tables (Section 2:2.6). If not specified, SEIMS will try to find lookup tables in BASE_DATA_DIR\data_prepare\lookup.*
 11. **WORKING_DIR:** *(Optional) Workspace for intermediate data. If not specified, SEIMS will use the default path of BASE_DATA_DIR\workspace.*
- **MONGODB:** *MongoDB related settings.*
 1. **HOSTNAME:** *IP address of MongoDB server, e.g., 127.0.0.1 (i.e., localhost).*
 2. **PORT:** *Port of MongoDB server, e.g., 27017 by the default.*
 3. **SpatialDBName:** *Name of the main spatial database of the study area created in MongoDB server (Figure 2:3-3). The name **MUST** be the same with the folder name of MODEL_DIR.*

4. `ClimateDBName`: Name of the hydro-climate database created in MongoDB server (Figure 2:3-3).
 5. `BMPScenarioDBName`: Name of the BMP scenarios database created in MongoDB server (Figure 2:3-3).
- **CLIMATE**: Filenames of climate data located in `CLIMATE_DATA_DIR`.
 1. `HydroClimateVarFile`: Types and Units of climate data variables (Section 2:2.4).
 2. `MeteoSiteFile`: The spatial information of meteorological station (Section 2:2.4).
 3. `PrecSiteFile`: The spatial information of precipitation station (Section 2:2.3.1)
 4. `MeteoDataFile`: The meteorological data (Section 2:2.4).
 5. `PrecDataFile`: The precipitation data (Section 2:2.3.2).
 6. `thiessenIdField`: (Optional) The field ID in the Thiessen polygon files of climate sites (Section 2:2.2), the default is `ID`.
 - **Spatial**: Filenames of spatial data located in `SPATIAL_DATA_DIR` and `TXT_DB_DIR`.
 1. `dem`: The original DEM data.
 2. `outlet_file`: (Optional) The outlet of the study area.
 3. `PrecSitesThiessen`: The Thiessen polygon file of precipitation stations.
 4. `MeteoSitesThiessen`: The Thiessen polygon file of meteorological stations.
 5. `landuseFile`: The raster file of landuse types.
 6. `landcoverInitFile`: The lookup table of initial landcover parameters (Section 2:2.6.2).
 7. `soilSEQNFile`: The raster file of soil sequences.
 8. `soilSEQNText`: The lookup table of soil properties (Section 2:2.6.1).
 9. `field_partition_thresh`: (Optional) Threshold values for the delineation of hydrologically connected fields (Wu et al., 2018). Multiple thresholds can be specified such as 10, 15, 20. The partitioned fields named `fields_<threshold>.tif` (e.g., `fields_15.tif`) will be imported into the main spatial database (i.e., `SpatialDBName`). In the meantime, a json formatted file named `connected_field_units_updown_<threshold>.json` will be generated in the `MODEL_DIR`, which describes the upstream and downstream relationships of fields and landuse information, etc. The hydrologically connected fields can be used as BMP configuration units for BMP scenarios analysis (see Section 2:8).
 - **OPTIONAL_PARAMETERS**: Optional parameters.
 1. `D8AccThreshold`: Flow accumulation threshold for stream and subbasin delineation, the default is 0. In such circumstance, the determination of threshold will be performed by the [drop analysis](#) function of TauDEM automatically.
 2. `np`: Number of processes for MPI-based parallel computing of C++ applications, e.g., TauDEM functions. The default is 4.
 3. `D8DownMethod`: Calculation method of distance down to a stream, the available values are pythagoras, horizontal, vertical, and surface, or p, h, v, and s for simplification, respectively. The default is `surface` or `s`. More information can be referred to <http://hydrology.usu.edu/taudem/taudem5/help53/DInfinityDistanceDown.html>.

4. `dorm_hr`: Day length threshold hours for dormancy, the default is -1.
5. `T_base`: Base temperature (degC) for heat unit calculation, the default is 0.
6. `imperviousPercInUrbanCell`: Impervious percent in urban units, the default is 0.3.
7. `defaultLanduse`: The default landuse type for `NoData` area.
8. `defaultSoil`: The default soil sequence type for `NoData` area.

2:3.3 Advanced usage

The Python scripts of data preprocessing are located in `SEIMS/seims/preprocess`. The prefix of script names are distinguished by functionality, i.e., '`sd_`' for the spatial discretization of watershed (e.g., the delineation of subbasins), '`sp_`' for the extraction of spatial parameters (e.g., landuse and soil related parameters according to the database of SWAT model and additional lookup tables, Section 2:2.6), '`hydro_`' for the data processing of hydrology and climate data, and '`db_`' for the input and/or output of MongoDB database. The `main.py` is the entrance for the entire workflow of data preprocessing. Each script can be executed independently though the unified format:

```
python <script_name> -ini </path/to/configuration-file>.
```

For example, to run the entire workflow:

```
01 >cd D:\demo\SEIMS\seims\preprocess
02 >python main.py -ini
D:\demo\SEIMS\data\youwuzhen\workspace\preprocess.ini .
```

If the BMP scenarios related data needs to be updated, there is only need to run the script of importing BMP scenarios data into MongoDB, i.e.,

```
01 >cd D:\demo\SEIMS\seims\preprocess
02 >python db_import_bmpscenario.py -ini
D:\demo\SEIMS\data\youwuzhen\workspace\preprocess.ini .
```

The entire workflow of preprocessing can be conducted in two main steps, i.e., the spatial discretization of watershed (i.e., `sd_delineation.py`) and the creation of MongoDB database (i.e., `db_build_mongodb.py`). Users are highly recommended to execute `sd_delineation.py` with the `D8AccThreshold` set to 0 first to get a preliminary delineation of subbasins. The automatically delineated subbasin may too fine to get a proper spatial scale for watershed modeling. Therefore, users may want to adjust the proper threshold manually according to the flow accumulated raster (i.e., `SEIMS\data\youwuzhen\workspace\taudem_delineated\accTauD8WithWeight.tif`, more information about the intermediate data will be described in the following section). After a final threshold is determined and the value of `D8AccThreshold` in the configuration file updated, the `main.py` should be rerun for the entire preprocessing.

Note that the present implementation of data preprocessing tools is to support the currently available SEIMS modules of watershed processes. That means, if the input data of a newly developed module is not available from the database or outputs of other modules, the corresponding preprocessing tools should be added. For more information, please referred to the “Section 4:1 Develop a new module of one watershed process”.

2:3.4 Intermediate data of preprocessing

As is shown in Figure 2:3-2, the intermediate data of data preprocessing are organized into seven folders.

- **taudem_delineated**: The original results and intermediate data of spatial delineation of subbasins based on TauDEM such as the flow accumulation data (`accTauD8WithWeight.tif`), the delineated subbasin data (`subbasinTauM.tif`), and the D8 flow direction masked by `subbasinTauM.tif` (`flowDirTauM.tif`), etc.
- **Lookup_tables**: Lookup tables of landuse database based on the SWAT model used for generating landuse type related parameters.

TODO: In the current version of SEIMS, landuse and soil type related spatial parameters (raster data) are prepared and imported into MongoDB separately as single files (i.e., GridFS in MongoDB). This solution will lead to the low performance of reading data of SEIMS main program. Therefore, in the future version, lookup tables should be used instead of separated raster- or array-based parameters.

- **spatial_raster**: All spatial parameters (raster data) masked by the delineated watershed boundary (i.e., `subbasinTauM.tif`).
- **spatial_shp**: Spatial vector data for the whole watershed and each subbasin, e.g., `reach_<N>.shp` (`N` represents the subbasin number starts from 1, while `reach.shp` is for the whole watershed), `subbasin_<N>.shp`, `basin.shp`, and `outlet.shp`.
- **Layering_info**: The flow in and out indexes of each basic simulation units (e.g., gridded cells and irregularly shaped fields), and the routing layers of UPDOWN (default, layering from source) and DOWNUP (layering from outlet) methods based on flow direction algorithms.
- **metis_output**: Groups partitioned by METIS software for static task scheduling of MPI-based parallel computing at subbasin levels. For example, the results of `metis.part.4` located in `SEIMS\data\youwuzhen\workspace\metis_output\kmetis` is
`3 0 2 1 1 3 1 3 2 0 2 0 2 1 1 1 3`, which means that the 17 subbasins of the Youwuzhen watershed will be distributed on four processes, i.e., subbasin IDs of (2, 10, 12), (4, 5, 7, 14, 15, 16), (3, 9, 11, 13), and (1, 6, 8, 17).
- **runtime_Log**: Runtime logs (with prefix of 'status_') and several input configuration files of SEIMS C++ applications (with prefix of 'config_').

2:3.5 Structure of the watershed modeling database

The screenshot of the structure of the watershed modeling database was shown in Figure 2:3-3. Three databases were created with the specific names in the configuration **INI** file of data preprocessing (Section 2:3.2). The preprocessed data were organized as different collections (or referred to as tables) within database.

- **demo_youwuzhen30m_Longterm_model**: Main spatial database of the study area
 1. Model configuration collections, such as `FILE_IN` and `FILE_OUT`.

2. Initial model parameters including calibration information, i.e., **PARAMETERS**.
 3. Reach related parameters, i.e., **REACHES**, including geometric parameters, default chemical parameters, and subbasin groups for task scheduling of MPI-based parallel computing, etc.
 4. Climate station information for each subbasin with the corresponding *HydroClimate* database name, i.e., **SITELIST**.
 5. BMP scenario database name, i.e., **BMPDATABASE**.
 6. Spatial parameters, i.e., **SPATIAL**.
 7. Lookup tables, i.e., **CROPLOOKUP**, **FERTILIZERLOOKUP**, **LANDUSELOOKUP**, **SOILLOOKUP**, **TILLAGELOOKUP**, and **URBANLOOKUP**.
- **demo_youwuzhen30m_HydroClimate**: Hydro-climate database
 1. Climate and hydrology stations, i.e., **SITES**.
 2. Climate data (**DATA_VALUES**) and hydrologic monitoring data (**MEASUREMENT**).
 3. Annual statistics of climate data, i.e., **ANNUAL_STATS**.
 4. Types and units of hydro-climate variables, i.e., **VARIABLES**.
 - **demo_youwuzhen30m_Scenario**: BMPs scenario database
 1. BMP IDs, i.e., **BMP_INDEX**.
 2. BMP scenarios, i.e., **BMP_SCENARIOS**.
 3. Various BMPs parameters, the collection names are depend on the plain text filename located in management practices data (e.g., **SEIMS\data\youwuzhen\data_prepare\scenario**), such as **PLANT_MANAGEMENT** and **AREAL_STRUCT_MANAGEMENT** in this demo study.

2:4. Running a SEIMS-based watershed model

2:4.1 Simple usage

For simple usage, open a CMD window, enter the following commands to execute the predefined SEIMS-based watershed model of the Youwuzhen watershed (hereafter referred to as the Youwuzhen watershed model).

```
01 >cd D:\demo\SEIMS\seims\test
02 >D:
03 >python demo_runmodel.py -name youwuzhen
```

The simulation will be finished in a ~28 seconds (Figure 2:4-1). The predefined output information can be found in the model folder (i.e., MODEL_DIR\OUTPUT0), such as Figure 2:4-2.

```
Microsoft Windows [Version 10.0.17134.472]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\ZhuLJ>cd D:\demo\SEIMS\seims\test
C:\Users\ZhuLJ>D:
D:\demo\SEIMS\seims\test>D:\demo\python27\python demo_runmodel.py -name youwuzhen
SEIMS binary location: D:\demo\SEIMS\bin
Demo data location: D:\demo\SEIMS\data\youwuzhen
Data preprocess location: D:\demo\SEIMS\data\youwuzhen\workspace
D:\demo\SEIMS\bin\seims_omp.exe -wp D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_1ongterm_model
-thread 4 -lyr 0 -host 127.0.0.1 -port 27017 -sce 0
The file O_SLOPE_LENGTH does not exist.
O_SLOPE_LENGTH is not existed or get file timed out!
[TIMESPAN][IO ][Input] 3.849
Simulation year: 2012
Simulation year: 2013
Simulation year: 2014
Simulation year: 2015
[TIMESPAN][COMP][ALL] 24.516
[TIMESPAN][COMP][TSD_RD_P] 0.001
[TIMESPAN][COMP][TSD_RD_TMEAN] 0.000
[TIMESPAN][COMP][TSD_RD_TMAX] 0.000
[TIMESPAN][COMP][TSD_RD_TMIN] 0.000
[TIMESPAN][COMP][TSD_RD_SR] 0.000
[TIMESPAN][COMP][TSD_RD_WS] 0.000
[TIMESPAN][COMP][TSD_RD_RM] 0.000
[TIMESPAN][COMP][ITP_P] 0.030
[TIMESPAN][COMP][ITP_TMEAN] 0.025
[TIMESPAN][COMP][ITP_TMAX] 0.022
[TIMESPAN][COMP][ITP_TMIN] 0.022
[TIMESPAN][COMP][ITP_SR] 0.021
```

Figure 2:4-1 Simple usage of the OpenMP version of the Youwuzhen watershed model

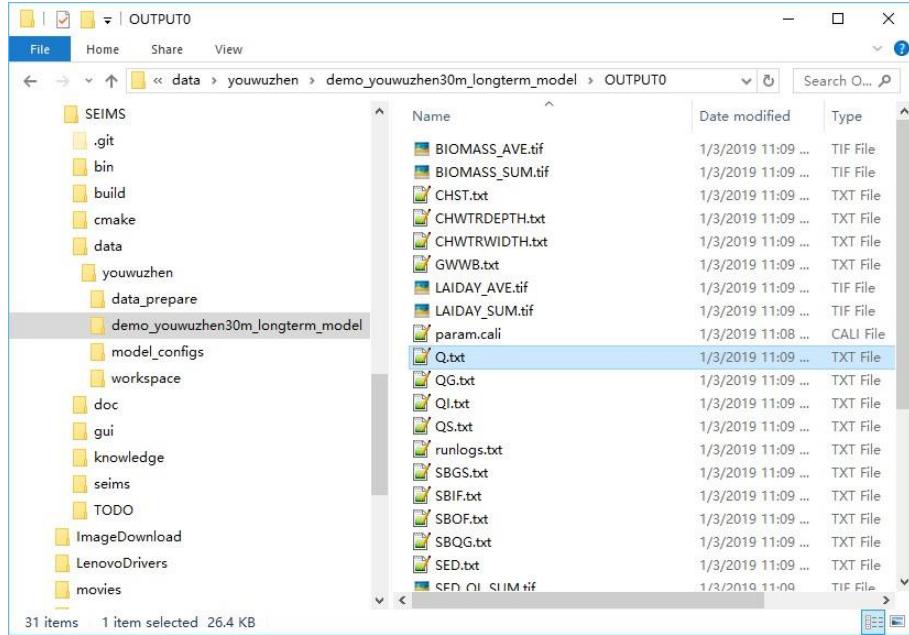


Figure 2:4-2 Predefined outputs of the Youwuzhen watershed model

2:4.2 File structure of a SEIMS-based watershed model

In simple terms, a SEIMS-based watershed model is a folder (e.g., `SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model`) which consists of several SEIMS configuration files such as `file.in`, `file.out`, `param.cal`, and `config.fig`.

The `file.in` and `file.out` for basic model configuration are required, while `param.cal` for calibrated model parameters is optional. These three plain text files will be read and imported into the main spatial database during data preprocessing (Section 2:3.2). The MongoDB also can be updated by only running a single preprocessing Python script (i.e., `SEIMS\seims\preprocess\db_import_model_parameters.py`) when these files are changed such as during the manual calibration.

The `config.fig` file, the most important configuration file for constructing a SEIMS-based watershed model, is responsible for selecting SEIMS modules to participate in the watershed simulation, as well as specifying the simulation order of SEIMS modules.

2:4.2.1 file.in

The `file.in` file is designed to define the simulation mode with time-step and the simulation period. Different from the plain text format described in Section 2:2.1.2, the basic format is `[TAG] | [VALUE]`. Currently, there are four required TAGs which must appear in this file.

- `MODE`: A string that indicates a long-term or storm event simulation. The string here can only be `Daily` or `Storm` (case insensitive).

- **INTERVAL**: An integer to specify the time-step (or the so-called temporal resolution) of simulation according to the simulation **MODE**. For long-term simulation, the unit is day, e.g., **INTERVAL | 1** means the time-step is one day. For storm event simulation, the unit is second, e.g., **INTERVAL | 3600** means the time-step is one hour. Specifically, two different time-steps can be assigned for storm event simulation considering the differences between hillslope scale processes and channel scale process. For example, **INTERVAL | 60, 3600** means that the time-step for hillslope scale processes is one minute and one hour for channel scale processes.
- **STARTTIME**: Starting date time of simulation with the format of **YYYY-MM-DD HH:MM:SS**
- **ENDTIME**: Ending date time of simulation with the format of **YYYY-MM-DD HH:MM:SS**

For the Youwuzhen watershed model, which performs a long-term simulation with a time-step of one day and runs from Jan 1, 2012, to Dec 31, 2015, the content of **file.in** file is as follows.

```
MODE|Daily  
INTERVAL|1  
STARTTIME|2012-01-01 00:00:00  
ENDTIME|2015-12-31 23:59:59
```

2:4.2.2 file.out

The **file.out** file is designed to define the outputs of a SEIMS-based watershed model. The format follows the plain text format described in Section 2:2.1.2. Each line is corresponding to one output variable. The available fields of each line are listed in Table 2:4-1.

Table 2:4-1 Available fields of **file.out** configuration file

Field name	Datatype	Description
OUTPUTID	String	Unique output identifier, e.g., QRECH for the streamflow at reach outlet.
TYPE	String	Data aggregation type for spatial data outputs, i.e., SUM , AVE , MAX , and MIN . Multiple types should be concatenated by En dash. For example, SUM-AVE means output the sum and average simultaneously. For time-series outputs, the TYPE can be NONE .
STARTTIME	Datetime string	Starting date time for the current output with the format of YYYY-MM-DD HH:MM:SS
ENDTIME	Datetime string	Ending date time for the current output with the format of YYYY-MM-DD HH:MM:SS
INTERVAL	Integer	Time-step for output. The default is -9999 , which means

Field name	Datatype	Description
INTERVAL_UNIT	String	the same time-step with the simulation will be used. Unit for INTERVAL, can be DAY and SEC. The default is -9999, which means the same with the simulation.
FILENAME	String	Output filename with suffix, e.g., Q.txt and PET.tif.
SUBBASIN	String or Integer	Selected subbasins for output, can be ALL, OUTLET, or the subbasin IDs concatenated with En dash, e.g., 1-4-5.

Among these fields, only the OUTPUTID is required. The values of missed fields will adopted the default values defined in the available output database which can be extended by users, i.e., SEIMS\seims\preprocess\database\AvailableOutputs.csv. So, users can specify the outputs of QRECH (Streamflow at reach outlet of each time step, m³/s) and PET (Potential evapotranspiration, mm) by simply specifying the output IDs, e.g.,

OUTPUTID

QRECH

PET , or specifying more detail information, e.g.,

OUTPUTID,TYPE,STARTTIME,ENDTIME,INTERVAL,INTERVAL_UNIT,SUBBASIN
QRECH,NONE,2012-01-01 00:00:00,2015-12-31 23:59:59,1,DAY,OUTLET
PET,SUM-AVE,2012-01-01 00:00:00,2015-12-31 23:59:59,-9999,-9999,ALL .

2:4.2.3 param.cal

The param.cal file is designed to define calibrated parameters. The absent of this file or a blank file is allowed and indicates the model will be executed with default parameter values. The format MUST be NAME, IMPACT, and CHANGE, while CHANGE is optional and no header line is allowed. The NAME MUST be one of the predefined list of parameters which can be extended by users, i.e.,

SEIMS\seims\preprocess\database\model_param_ini.csv. The CHANGE means the type of impact on the parameter, which can be RC (relative change) or AC (absolute change). The IMPACT is a float value of change, for RC, the new parameter value will be initial_value * impact, while for AC it is initial_value + impact.

Thus, three styles are accepted in param.cal file, such as:

```
Runoff_co,1.5
gw0,-50,
K_pet,-0.3,AC .
```

2:4.2.4 config.fig

The config.fig file is designed to define SEIMS modules used in a SEIMS-based watershed model. The sequences of these modules will be the execution order during simulation. The basic order of SEIMS modules should be driver factors (i.e., climate

data) related modules, modules of hillslope processes, and modules of channel and groundwater routing processes, such as the `config.fig` content of the Youwuzhen watershed model in Figure 2:4-3. More details about the selected modules of the Youwuzhen watershed model can be found in Qin *et al.* (2018).

```
01 ### Driver factors, including meteorological data and precipitation
02 0 | TimeSeries || TSD_RD
03 0 | Interpolation_0 | Thiessen | ITP
04 ### Surface processes
05 0 | Soil temperature | Finn Plauborg | STP_FP
06 0 | PET | PenmanMonteith | PET_PM
07 0 | Interception | Maximum Canopy Storage | PI_MCS
08 0 | Snow melt | Snowpeak Daily | SNO_SP
09 0 | Infiltration | Modified rational | SUR_MR
10 0 | Depression and Surface Runoff | Linsley | DEP_LINSLEY
11 0 | Hillslope erosion | MUSLE | SERO_MUSLE
12 0 | Plant Management Operation | SWAT | PLTMGT_SWAT
13 0 | Percolation | Storage routing | PER_STR
14 0 | Subsurface | Darcy and Kinematic | SSR_DA
15 0 | SET | Linearly Method from WetSpa | SET_LM
16 0 | PG | Simplified EPIC | PG_EPIC
17 0 | ATMDEP | Atmosphere deposition | ATMDEP
18 0 | NUTR_TF | Nutrient Transformation of C, N, and P | NUTR_TF
19 0 | Water overland routing | IUH | IUH_OL
20 0 | Sediment overland routing | IUH | IUH_SED_OL
21 0 | Nutrient | Attached nutrient loss | NUTRSED
22 0 | Nutrient | Soluble nutrient loss | NUTRMV
23 0 | Pothole | SWAT cone shape | IMP_SWAT
24 0 | Soil water | Water balance | SOL_WB
25 ### Route Modules, including water, sediment, and nutrient
26 0 | Groundwater | Linear reservoir | GWA_RE
27 0 | Nutrient | groundwater nutrient transport | NUTRGW
28 0 | Water channel routing | MUSK | MUSK_CH
29 0 | Sediment channel routing | Simplified Bagnold equation | SEDR_SBAGNOLD
30 0 | Nutrient | Channel routing | NutrCH_QUAL2E
```

Figure 2:4-3 SEIMS modules involved in the demo Youwuzhen watershed model

As shown in Figure 2:4-3, different from the plain text format described in Section 2:2.1.2, the basic format for each selected SEIMS module is `[MODULE NO.]|[PROCESS NAME]| [METHOD NAME] | [MODULE ID]`, in which,

- `MODULE NO.` could be any number
- `PROCESS NAME` is the name of the corresponding watershed process
- `METHOD NAME` is the name of the algorithm to simulate the watershed process
- `MODULE ID` is the ID of the SEIMS module, i.e., the file name of the corresponding dynamic library (dll, so, or dylib) file. If the module cannot be located or loaded, SEIMS will exit and report an error.

SEIMS main program only uses the `PROCESS NAME` and `MODULE ID`, while the `MODULE NO.` and `METHOD NAME` are just designed for readability. The `MODULE ID` should match exactly with the ID of SEIMS module. The `PROCESS NAME` may contains extra configuration information for some modules.

- For interpolation modules, the vertical interpolation method (0 means do not perform vertical interpolation based on lapse rate, while 1 means do) must be specified as suffix of the `PROCESS NAME`, e.g.,

`0 | Interpolation_1 | Thiessen | ITP`

TODO: In the current version of SEIMS, the lapse rate of precipitation and temperature are defined as constants, i.e., 0.03 mm/100 m for precipitation and -0.65 degC/100 m for temperature. In the future development, the lapse rate of various meteorological variables should be allowed as inputs according to the study area.

2:4.3 Advanced usage

The SEIMS main programs include an OpenMP version (i.e., `seims_omp`) and a version which is a hybrid of OpenMP and MPI (hereafter referred to as MPI&OpenMP version, i.e., `seims_mpi`). In the OpenMP version, parallel computing is conducted at the basic-unit level (e.g., gridded cells or irregularly shaped fields) in each module based on OpenMP. In the MPI&OpenMP version, the watershed is first decomposed into subbasins, and MPI-based parallel computation is conducted at the subbasin level. And then within each subbasin, same as the OpenMP version, OpenMP-based parallel computation is conducted at the basic-unit level.

Note: If OpenMP is not supported by the C/C++ compiler, the OpenMP-based parallel computing will not take affect and the executable name of SEIMS main program will be `seims`.

2:4.3.1 Run with command line interface

The Command Line Interface (CLI) is the recommended way to run SEIMS-based watershed model. The invoking commands are slightly different between the OpenMP version and MPI&OpenMP version.

2:4.3.1.1 OpenMP version

As is shown in Figure 2:1-8, the complete and recommended usage of the OpenMP version of SEIMS main program is as follows.

```
seims_omp -wp <modelPath> [-thread <threadsNum> -lyr
<layeringMethod> -host <hostname> -port <port> -sce
<scenarioID> -cali <calibrationID> -id <subbasinID>]
```

In which,

1. `modelPath` is the path of the SEIMS-based watershed model (see `MODEL_DIR` in the configuration file of data preprocessing, Section 2:3.2).
2. `threadsNum` (Optional) is the number of threads used for OpenMP-based parallel computing, which must be greater or equal than 1 (default).

3. `layeringMethod` (*Optional*) can be 0 and 1, which means the routing layering method based on flow direction algorithms of UPDOWN (default, layering from source) and DOWNUP (layering from outlet), respectively.
4. `hostname` is the IP address of MongoDB server. The default is 127.0.0.1 (i.e., localhost).
5. `port` is the port number of MongoDB server, and the default is 27017.
6. `scenarioID` is the ID of BMP scenario which has been defined in the BMP_SCENARIOS collection of Scenario database. By default, the `scenarioID` is -1, which means no scenario will be applied.
7. `calibrationID` is the ID (i.e., index) of calibration data which has been defined in PARAMETERS collection of the main database. By default, the `calibrationID` is -1, which means no calibration will be applied.
8. `subbasinID` is the subbasin that will be executed. 0 means the whole watershed. 9999 is reserved for Field version.

For the Youwuzhen watershed, one of the complete usages is like:

```
01 >cd D:\demo\SEIMS\bin  
02 >D:  
03 >seims_omp -wp  
D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model  
-thread 4 -lyr 1 -host 127.0.0.1 -port 27017 -sce 0 .
```

The runtime logs of the OpenMP version was shown in Figure 2:4-1, in which the time-consuming of data Input/Output (IO), computation of each module, and entire simulation were shown.

The output folder was created in the model directory with the folder name followed a naming format of `OUTPUT-<scenarioID>-<calibrationID>`. For example,

- `OUTPUT-1` means no scenario and calibration ID is 1
- `OUTPUT100` means scenario ID is 100 and no calibration
- `OUTPUT100-2` means scenario ID is 100 and calibration ID is 2.

2:4.3.1.2 MPI&OpenMP version

The MPI&OpenMP version of SEIMS main program has the same arguments with the OpenMP version but the different way of invoking it (Figure 3:1-1). The basic format to run a MPI program is:

```
mpiexec -<hostsopt> <hostfile> -n <processNum> seims_mpi -  
wp <modelPath> [-thread <threadsNum> -lyr <layeringMethod>  
-host <hostname> -port <port> -sce <scenarioID> -cali  
<calibrationID> -id <subbasinID>]
```

In which,

1. **hostopt** (*Optional*) is the identifier of the input argument of MPI to specify a host list file which is various with different MPI implementations, e.g., `-machinefile` or `-hostfile`.

2. **hostfile** (*Optional*) is the host list file which may different with MPI implementation. For example, OpenMPI takes the format like:

```
b10n08.cluster.com slots=3 max-slots=3
b08n01.cluster.com slots=4 max-slots=4
b10n05.cluster.com slots=4 max-slots=4 ,
```

and IntelMPI uses:

```
b10n08.cluster.com 3
b08n01.cluster.com 4
b10n05.cluster.com 4
```

3. **processNum** (*Optional*) is the number of processes.

```
D:\demo\SEIMS\bin>seims_mpi
FAILURE: To run the program, use either the Simple Usage option or the Complete Usage option as below.
Simple Usage:
  seims_mpi <modelPath> [<threadsNum> <layeringMethod> <IP> <port> <scenarioID> <calibrationID> <subbasinID>]
    <modelPath> is the path of the SEIMS-based watershed model.
    <threadsNum> is the number of thread used by OpenMP, which must be greater or equal than 1 (default).
    <layeringMethod> can be 0 and 1, which means UP_DOWN (default) and DOWN_UP, respectively.
    <IP> is the address of MongoDB database, and <port> is its port number.
    By default, MongoDB IP is 127.0.0.1 (i.e., localhost), and the port is 27017.
    <scenarioID> is the ID of BMPs Scenario which has been defined in BMPs database.
    By default, the Scenario ID is -1, which means not used.
    <calibrationID> is the ID of Calibration which has been defined in PARAMETERS table.
    By default, the Calibration ID is -1, which means not used.
    <subbasinID> is the subbasin that will be executed. 0 means the whole watershed. 9999 is reserved for Field version.

Complete and recommended Usage:
  <executable of MPI (e.g., mpiexec and mpirun)> -hosts(or machinefile, configfile, etc) <hosts_list_file> -n <process numbers> seims_mpi -wp <modelPath> [-thread <threadsNum> -lyr <layeringMethod> -host <IP> -port <port> -sce <scenarioID> -cali <calibrationID> -id <subbasinID>]
```

Figure 2:4-4 Usage of the MPI&OpenMP version of SEIMS main program

For the Youwuzhen watershed, one of the complete usages for the MPI&OpenMP version on Windows platform is like:

```
01 >cd D:\demo\SEIMS\bin
02 >D:
03 >mpiexec -n 4 seims_mpi -wp
D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model
-thread 1 -lyr 1 -host 127.0.0.1 -port 27017 -sce 0
```

The corresponding runtime output logs was shown in Figure 2:4-5. Different from the output information of the OpenMP version (Figure 2:4-1), the maximum ([TIMESPAN][MAX]), minimum ([TIMESPAN][MIN]), and average ([TIMESPAN][AVG]) time-consuming of data IO, computation of all modules, and entire simulation derived from each process (i.e., four processes of MPI in this example) are given. The maximum time-consuming is the most commonly used statistics for the evaluation of parallel performance.

```
Command Prompt
Simulation year: 2012
Simulation year: 2013
Simulation year: 2014
Simulation year: 2015

[TIMESPAN][MAX][COMP][Slope] 18.352
[TIMESPAN][MAX][COMP][Channel] 6.259
[TIMESPAN][MAX][COMP][Barrier] 1.578
[TIMESPAN][MAX][COMP][ALL] 21.248
[TIMESPAN][MAX][IO ][Input] 30.887
[TIMESPAN][MAX][IO ][Output] 1.693
[TIMESPAN][MAX][IO ][ALL] 32.580
[TIMESPAN][MAX][SIMU][ALL] 53.828

[TIMESPAN][MIN][COMP][Slope] 14.975
[TIMESPAN][MIN][COMP][Channel] 1.308
[TIMESPAN][MIN][COMP][Barrier] 0.003
[TIMESPAN][MIN][COMP][ALL] 21.241
[TIMESPAN][MIN][IO ][Input] 30.887
[TIMESPAN][MIN][IO ][Output] 1.114
[TIMESPAN][MIN][IO ][ALL] 32.000
[TIMESPAN][MIN][SIMU][ALL] 53.241

[TIMESPAN][AVG][COMP][Slope] 16.572
[TIMESPAN][AVG][COMP][Channel] 3.836
[TIMESPAN][AVG][COMP][Barrier] 0.826
[TIMESPAN][AVG][COMP][ALL] 21.245
[TIMESPAN][AVG][IO ][Input] 30.887
[TIMESPAN][AVG][IO ][Output] 1.402
[TIMESPAN][AVG][IO ][ALL] 32.289
[TIMESPAN][AVG][SIMU][ALL] 53.534

d:\demo\SEIMS\bin>
```

Figure 2:4-5 Runtime logs of the MPI&OpenMP version of the Youwuzhen watershed model

To demonstrate the parallel performance of the MPI&OpenMP version, a Linux cluster with IBM Platform LSF for workload management is used. One example that utilize 12 cores (i.e., 12 processes created by MPI) from 4 computing node and 2 threads per process is shown in Figure 2:4-6. A job script (i.e.,

`run_ywz30m_mpi_proc12_thread2.ls` in Figure 2:4-6) was first created which includes the generation of a hosts list file that allocated by LSF and the settings of paths and commands for the execution of the Youwuzhen watershed model. Then the job script was submitted by `bsub` command. As is shown in Figure 2:4-6, four computing node (i.e., b07n04, b07n05, n09n12, and b06n14) were allocated to execute the job. According to the runtime logs, the maximum computing time of simulation among the 12 processes is 7.684 s which is much more effective than those single model runs executed on one laptop (Figure 2:4-1 and Figure 2:4-5).

More information about the parallel performance tests of a single run of the SEIMS-based watershed model can be referred to Zhu *et al.* (2019a).

The screenshot shows a terminal window titled '192.168.253.203 - SecureCRT'. The user has run several commands to submit and monitor a job to a cluster queue. The output includes:

```

[zhulj@login03 seims_phd]$ bsub < run_ywz30m_mpi_proc12_thread2.1sf
Job <445144> is submitted to queue <rhel6normal>.
[zhulj@login03 seims_phd]$ bjobs
JOBID USER STAT QUEUE FROM_HOST EXEC_HOST JOB_NAME SUBMIT_TIME
445144 zhulj RUN rhel6norma login03.clu 3*b07n04.cl *2_Thread4 Jan 4 11:12
3*b07n05.cluster.com
3*b09n12.cluster.com
3*b06n14.cluster.com

[zhulj@login03 seims_phd]$ vi ./demo_youwuzhen30m_longterm_model/stdout.log
Simulation year: 2012
Simulation year: 2013
Simulation year: 2014
Simulation year: 2015

[[TIMESPAN][MAX][COMP][slope], 5.408
[[TIMESPAN][MAX][COMP][channel], 4.797
[[TIMESPAN][MAX][COMP][Barrier], 5.236
[[TIMESPAN][MAX][COMP][ALL], 7.684
[[TIMESPAN][MAX][IO][Input], 6.048
[[TIMESPAN][MAX][IO][Output], 2.945
[[TIMESPAN][MAX][IO][ALL], 8.993
[[TIMESPAN][MAX][SIMU][ALL], 16.677

[[TIMESPAN][MIN][COMP][slope], 1.857
[[TIMESPAN][MIN][COMP][channel], 0.561
[[TIMESPAN][MIN][COMP][Barrier], 0.002
[[TIMESPAN][MIN][COMP][ALL], 7.666
[[TIMESPAN][MIN][IO][Input], 3.025
[[TIMESPAN][MIN][IO][Output], 2.153
[[TIMESPAN][MIN][IO][ALL], 5.179
[[TIMESPAN][MIN][SIMU][ALL], 12.844

[[TIMESPAN][AVG][COMP][slope], 3.174
[[TIMESPAN][AVG][COMP][channel], 1.879
[[TIMESPAN][AVG][COMP][Barrier], 2.619
[[TIMESPAN][AVG][COMP][ALL], 7.675
[[TIMESPAN][AVG][IO][Input], 5.529
[[TIMESPAN][AVG][IO][Output], 2.242
[[TIMESPAN][AVG][IO][ALL], 7.771
[[TIMESPAN][AVG][SIMU][ALL], 15.445

```

Figure 2:4-6 Run the MPI&OpenMP version of the Youwuzhen watershed model on a Linux cluster

2:4.3.2 Run with Python

Except for the CLI way, SEIMS also provides a configuration file based method to execute model with Python. Same with the simple usage of data preprocessing (2:3.2), a configuration file was also generated according to the local paths of SEIMS during the simple usage of running the Youwuzhen watershed model, such as `SEIMS\data\youwuzhen\workspace\runmodel.ini` (Figure 2:4-7).

The configuration file of running SEIMS-based watershed model, such as that of the Youwuzhen watershed model shown in Figure 2:4-7, includes several options within one section (i.e., `SEIMS_Model`) to specify the arguments of SEIMS main programs introduced in previous sections. The full list of the arguments are as follows.

- `SEIMS_Model`:
 1. `MODEL_DIR`: *The model path of the study area which includes several configuration files for the watershed modeling.*
 2. `BIN_DIR`: *The path of SEIMS modules and main programs, which is the same with `CPP_PROGRAM_DIR` defined in the configuration file of data preprocessing (Section 2:3.2) by default.*
 3. `HOSTNAME`: *IP address of MongoDB server, e.g., 127.0.0.1 (i.e., localhost).*

4. **PORT**: Port of MongoDB server, e.g., 27017 by the default.
5. **VERSION**: (Optional) Version of SEIMS main program which can be **OMP** and **MPI** for OpenMP version and MPI&OpenMP version, respectively. The default is **OMP**.
6. **MPI_BIN**: (Optional) Full path of the executable of MPI, e.g., **C:\Program Files\Microsoft MPI\Bin\mpiexec.exe**.
7. **hostopt**: (Optional) The identifier of the input argument of MPI to specify a host list file which is various with different MPI implementations, e.g., **-machinefile** or **-hostfile**.
hostfile: (Optional) The host list file which may different with MPI implementation.
8. **processNum**: (Optional) The number of processes, the default is 1.
9. **threadsNum**: (Optional) The number of threads per process, the default is 4.
10. **layeringMethod**: (Optional) Routing layering method based on flow direction algorithms, can be 0 or 1.
11. **scenarioID**: (Optional) The ID of BMP scenario.
12. **calibrationID**: (Optional) The ID (i.e., index) of calibration data.
13. **subbasinID**: (Optional) The ID of subbasin, 0 for the whole watershed and 9999 is reserved for Field version.
14. **Sim_Time_start**: (Optional) Starting date time of simulation with the format of **YYYY-MM-DD HH:MM:SS**.
15. **Sim_Time_end**: (Optional) Ending date time of simulation with the format of **YYYY-MM-DD HH:MM:SS**. If not specified, the starting and ending date time prepared in database (i.e., **FILE_IN** collection) will be applied.
16. **Output_Time_start**: (Optional) Starting date time of output items with the format of **YYYY-MM-DD HH:MM:SS**.
17. **Output_Time_end**: (Optional) Ending date time of output items with the format of **YYYY-MM-DD HH:MM:SS**. If specified, the time period of all selected output IDs in **FILE_OUT** collection will be updated and applied.

To run the SEIMS-based watershed model defined by the configuration file described above, please follow the unified format of running SEIMS Python scripts, e.g.,

```
01 >cd D:\demo\SEIMS\seims  
02 >python run_seims.py -ini  
D:\demo\SEIMS\data\youwuzhen\workspace\runmodel.ini
```

```
01 [SEIMS_Model]
02 MODEL_DIR = D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model
03 BIN_DIR = D:\demo\SEIMS\bin
04 HOSTNAME = 127.0.0.1
05 PORT = 27017
06 version = OMP
07 mpi_bin = mpiexec
08 # hostopt = ...
09 # hostfile = ...
10 processNum = 2
11 threadsNum = 4
12 layeringMethod = 1
13 scenarioID = 0
14 calibrationID = -1
15 subbasinID = 0
16 # Simulation period (UTCTIME)
17 Sim_Time_start = 2012-01-01 00:00:00
18 Sim_Time_end = 2015-12-31 23:59:59
19 # Time period of ALL selected outputs
20 Output_Time_start = 2012-01-01 00:00:00
21 Output_Time_end = 2015-12-31 00:00:00
```

Figure 2:4-7 Configuration content for model execution of the Youwuzhen watershed model

2:4.3.3 Customize the demo Youwuzhen watershed model

To demonstrate the ability of SEIMS to construct different watershed models, a simple change is made to the Youwuzhen watershed model, i.e., replacing the Penman-Monteith method (PET_PM) to the Priestley-Taylor method (PET_PT) for the simulation of potential evapotranspiration. Note that, except for the potential evapotranspiration, the PET_PM module also simulates the potential plant transpiration while the PET_PT not, thus the module named AET_PTH is added for the simulation of potential plant transpiration and potential and actual soil evaporation. Meanwhile, the soil evaporation module (SET_LM) is no longer need. The newly customized Youwuzhen watershed model (i.e., the config.fig file) was shown in Figure 2:4-8.

```
01 ### Driver factors, including climate and precipitation
02 0 | TimeSeries | | TSD_RD
03 0 | Interpolation_0 | Thiessen | ITP
04 #### Surface processes
05 0 | Soil temperature | Finn Plauborg | STP_FP
06 0 | PET | PriestleyTaylor | PET_PT
07 #0 | PET | PenmanMonteith | PET_PM
08 0 | Interception | Maximum Canopy Storage | PI_MCS
09 0 | Snow melt | Snowpeak Daily | SNO_SP
10 0 | Infiltration | Modified rational | SUR_MR
11 0 | Depression and Surface Runoff | Linsley | DEP_LINSLEY
12 0 | Hillslope erosion | MUSLE | SERO_MUSLE
13 0 | Plant Management Operation | SWAT | PLTMGT_SWAT
14 0 | Percolation | Storage routing | PER_STR
15 0 | Subsurface | Darcy and Kinematic | SSR_DA
16 0 | AET | Hargreaves and PriestleyTaylor | AET_PTH
17 #0 | SET | Linearly Method from WetSpa | SET_LM
18 0 | PG | Simplified EPIC | PG_EPIC
19 0 | ATMDEP | Atmosphere deposition | ATMDEP
20 0 | NUTR_TF | Nutrient Transformation of C, N, and P | NUTR_TF
21 0 | Water overland routing | IUH | IUH_OL
22 0 | Sediment overland routing | IUH | IUH_SED_OL
23 0 | Nutrient | Attached nutrient loss | NUTRSED
24 0 | Nutrient | Soluble nutrient loss | NUTRMV
25 0 | Pothole | SWAT cone shape | IMP_SWAT
26 0 | Soil water | Water balance | SOL_WB
27 #### Route Modules, including water, sediment, and nutrient
28 0 | Groundwater | Linear reservoir | GWA_RE
29 0 | Nutrient | groundwater nutrient transport | NUTRGW
30 0 | Water channel routing | MUSK | MUSK_CH
31 0 | Sediment channel routing | Simplified Bagnold equation | SEDR_SBAGNOLD
32 0 | Nutrient | Channel routing | NutrCH_QUAL2E
```

Figure 2:4-8 New Youwuzhen watershed model after the simple customization, i.e., replacing the Penman-Monteith method to Priestley-Taylor method for the simulation of potential evapotranspiration.

With all model parameters remain the default values, the demo Youwuzhen watershed model and the new customized model were executed respectively. Figure 2:4-9 showed the spatial distributions of average potential evapotranspiration simulated by the Penman-Monteith method (Figure 2:4-9a) and Priestley-Taylor method (Figure 2:4-9b), which have a very similar spatial pattern but different values. The differences of the simulated potential evapotranspiration can also be reflected in the simulated streamflow at the watershed outlet (Figure 2:4-10).

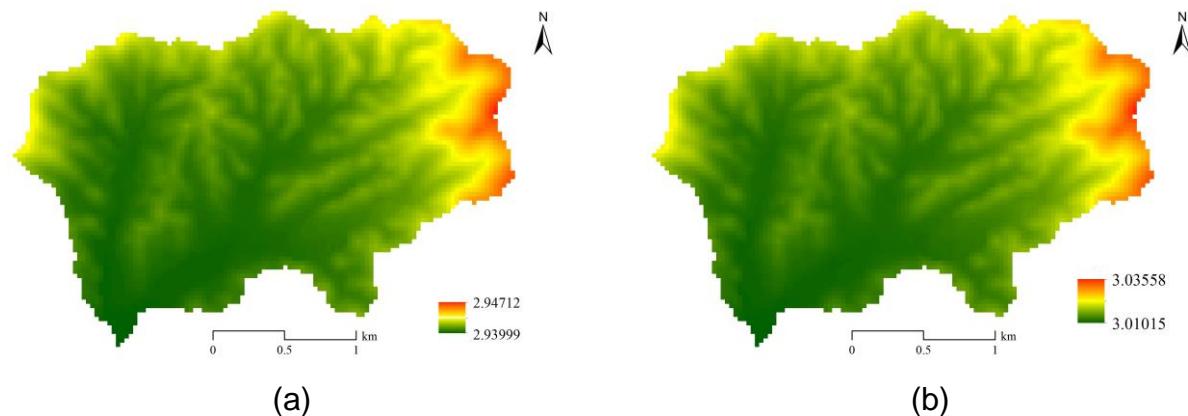


Figure 2:4-9 Spatial distributions of average potential evapotranspiration simulated by the Youwuzhen watershed model using the (a) Penman-Monteith method and the (b) Priestley-Taylor method

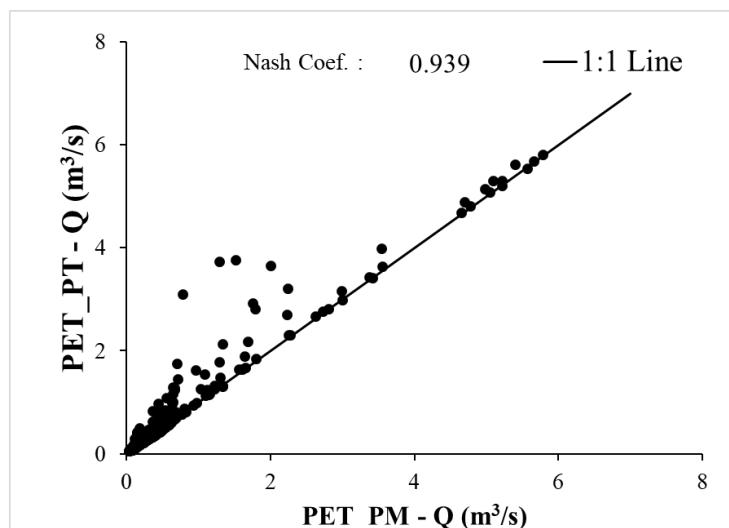


Figure 2:4-10 Differences of the streamflow (Q, m^3/s) at the outlet of Youwuzhen watershed derived from the Youwuzhen watershed models using the Penman-Monteith method (PET_PM) and Priestley-Taylor method (PET_PT) for the simulation of potential evapotranspiration, respectively.

2:5. Postprocessing

After running a SEIMS-based watershed model, the postprocessing tool may be used for the calculation of model performance statistics such as Nash-Sutcliffe Efficiency (NSE), and the plotting of hydrographs of time-series data at the reach outlet such as streamflow and sediment.

2:5.1 Simple usage

For simple usage, open a CMD window, enter the following commands to execute the postprocessing of the Youwuzhen watershed model.

```
01 >cd D:\demo\SEIMS\seims\test  
02 >D:  
03 >python demo_postprocess.py -name youwuzhen
```

The runtime logs of postprocessing showed input paths and output model performance statistics (Figure 2:5-1).



The screenshot shows a Windows Command Prompt window titled "Command Prompt". The window contains the following text:

```
D:\demo\SEIMS\seims\test>d:\demo\python27\python demo_postprocess.py -name youwuzhen
SEIMS binary location: D:\demo\SEIMS\bin
Demo data location: D:\demo\SEIMS\data\youwuzhen
Data_preprocess location: D:\demo\SEIMS\data\youwuzhen\workspace
Warning: vali_time_start is NOT defined in OPTIONAL_PARAMETERS!
Warning: vali_time_end is NOT defined in OPTIONAL_PARAMETERS!
Read precipitation from 01/01/13 00:00:00 to 12/31/15 23:59:59 done.
Read simulation from 01/01/13 00:00:00 to 12/31/15 23:59:59 done.
Read observation data of Q, SED from 01/01/13 00:00:00 to 12/31/15 23:59:59 done.
Match observation and simulation done.
Cali-Q-NSE, Cali-Q-RSR, Cali-Q-PBIAS, Cali-Q-R2,
-3.842, 2.200, -20.772, 0.141,
d:\demo\python27\lib\site-packages\matplotlib\font_manager.py:1331: UserWarning: findfont: Font family [u'cursive'] not found. Falling back to DejaVu Sans
(prop.get_family(), self.defaultFamily[fontext]))
Cali-SED-NSE, Cali-SED-RSR, Cali-SED-PBIAS, Cali-SED-R2,
-0.155, 1.075, 94.177, 0.186,
```

The command prompt shows the path "D:\demo\SEIMS\seims\test>" at the bottom.

Figure 2:5-1 Runtime logs of postprocessing of the Youwuzhen watershed model

By default, the hydrographs of streamflow (e.g., `Q.txt`) and sediment (e.g., `SED.txt`) at the watershed outlet were plotted and saved as figure files with the formats of `PNG` and `PDF`, e.g.,

`SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model\OUTPUT0\png\Q-2013-01-01-2015-12-31.png` (Figure 2:5-2).

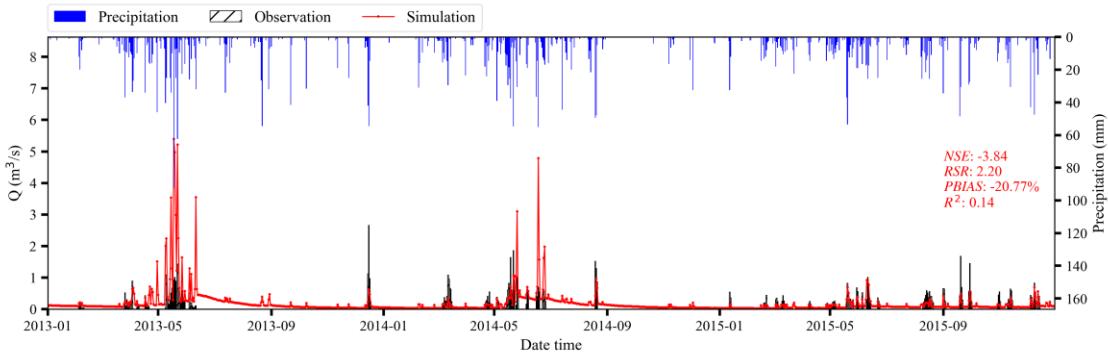


Figure 2:5-2 Hydrograph of streamflow (Q , m^3/s) at the watershed outlet derived from the Youwuzhen watershed model with default model parameters

2:5.2 Configuration file of postprocessing

Same to the data preprocessing, the simple usage of postprocessing also includes generating the configuration **INI** file according to the local paths of SEIMS such as Figure 2:5-3, and executing postprocessing by the advanced usage (See Section 2:5.3).

The configuration file of postprocessing, such as that of the Youwuzhen watershed model shown in Figure 2:5-3, includes four sections, i.e., **SEIMS_Model**, **PARAMETERS**, **OPTIONAL_PARAMETERS**, and **OPTIONAL_MATPLOT_SETTING**. Among them, **OPTIONAL_PARAMETERS** and **OPTIONAL_MATPLOT_SETTING** are optional. The names of sections and options should not be changed.

- ***SEIMS_Model***: Basic settings of the SEIMS-based watershed model, including **HOSTNAME**, **PORT**, **MODEL_DIR**, **BIN_DIR**, **scenarioID**, and **calibrationID**. The meanings of these options have been introduced in Section 2:4.3.2.
- ***PARAMETERS***
 1. **PLOT_SUBBASINID**: The subbasin ID to be plotted.
 2. **PLOT_VARIABLES**: The variables to be plotted, such as **Q**, **SED**, and **CH_TN**. Multiple variables can be separated by comma. Note that, the specified subbasin ID and the corresponding variables MUST be set as time-series output data in **file.out** (Section 2:4.2.2) and/or **FILE_OUT** collection of the main MongoDB database. If there is no matched observed data for the specified variable, the model performance statistics will not able to be calculated.
- ***OPTIONAL_PARAMETERS***: (Optional) Time period of model calibration and validation. The starting and ending date times of either calibration or validation MUST be presented or omitted simultaneously.
 1. **Cali_Time_start**: Starting date time of model calibration with the format of **YYYY-MM-DD HH:MM:SS**.
 2. **Cali_Time_end**: Ending date time of model calibration.
 3. **Vali_Time_start**: Starting date time of model validation.
 4. **Vali_Time_end**: Ending date time of model validation.

- **OPTIONAL_MATPLOT_SETTINGS**: (Optional) Plot settings for matplotlib.
 1. **FIGURE_FORMATS**: The format(s) of output figure(s) which can be one or multiple of PNG, JPG, TIF, PDF, EPS, SVG, and PS. Multiple formats are separated by comma, e.g., PNG, JPG, EPS.
 2. **LANG_CN**: Whether Chinese are included (True) or not (False) in labels and legend texts, etc.
 3. **FONT_TITLE**: Font title (font name) for the output figures. Below are tips to check the available fonts or use new ones.
 - a. Get the currently available font titles by the following Python code.

```
01 #! /usr/bin/env python
02 # -*- coding: utf-8 -*-
03 from matplotlib import font_manager
04 flist = font_manager.get_fontconfig_fonts()
05 font_names = list()
06 for fname in flist:
07     try:
08         font_names.append(font_manager.FontProperties(fname=fname).get_name())
09     except IOError or Exception:
10         continue
11 print(font_names)
```
 - b. To add and use a new font:
 - Download a font with .ttf format such as [YaHei Mono](#) which supports a hybrid of Chinese and English. If you have only .ttc formatted font file, you may want to convert it to .ttf file using [transfonter](#).
 - Copy the font file to the font directory of matplotlib, e.g., D:\demo\python27\Lib\site-packages\matplotlib\mpl-data\fonts\ttf
 - Delete the cache files of matplotlib font located in the Windows user's directory, e.g., C:\Users\ZhuLJ\.matplotlib. The cache files may be fontList.cache and fontList.json for Python 2, and fontList.py3k.cache and fontList-v300.json for Python 3.
 - Now the new font should be available for plotting. If the new font is not rendered, it may be because matplotlib needs to refresh its font cache. Run the following Python code to rebuild it.

```
01 from matplotlib import font_manager
02 font_manager._rebuild()
```
 4. Font sizes of title (TITLE_FONTSIZE), legend (LEGEND_FONTSIZE), axis-tick label (TICKLABEL_FONTSIZE), axis label (AXISLABEL_FONTSIZE), and universal label (LABEL_FONTSIZE) in points.
 5. **DPI**: The resolution in dots (Integer) per inch, which is applied to non-vector formats.

```

01 [SEIMS_Model]
02 HOSTNAME = 127.0.0.1
03 PORT = 27017
04 MODEL_DIR = D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model
05 BIN_DIR = D:\demo\SEIMS\bin
06 scenarioID = 0
07 calibrationID = -1
08
09 [PARAMETERS]
10 PLOT_SUBBASINID = 4
11 PLOT_VARIABLES = Q SED
12
13 [OPTIONAL_PARAMETERS]
14 # Calibration period (UTCTIME)
15 Cali_Time_start = 2013-01-01 00:00:00
16 Cali_Time_end = 2015-12-31 23:59:59
17 # Validation period (UTCTIME)
18 #Vali_Time_start = 2013-01-01 00:00:00
19 #Vali_Time_end = 2013-12-31 23:59:59
20
21 [OPTIONAL_MATPLOT_SETTINGS]
22 FIGURE_FORMATS = PDF,PNG
23 LANG_CN = False
24 FONT_TITLE = Times New Roman
25 TITLE_FONTSIZE = 14
26 LEGEND_FONTSIZE = 12
27 TICKLABEL_FONTSIZE = 12
28 AXISLABEL_FONTSIZE = 12
29 LABEL_FONTSIZE = 14
30 DPI = 300

```

Figure 2:5-3 Configuration content for postprocessing of the Youwuzhen watershed model

2:5.3 Advanced usage

The Python scripts of postprocessing are located in `SEIMS/seims/postprocess`. The `main.py` is the entrance of postprocessing which can be executed through the unified format of running SEIMS Python scripts, e.g.,

```

01 >cd D:\demo\SEIMS\seims\postprocess
02 >python main.py -ini
D:\demo\SEIMS\data\youwuzhen\workspace\postprocess.ini .

```

To illustrate the functionality of postprocessing, Figure 2:5-4 showed the calibration and validation of the simulated streamflow (Q, m³/s) at the watershed outlet derived from the demo Youwuzhen watershed model with default model parameters with Chinese labels and legend texts based on `YaHei Mono` font (changes of the configuration was shown in Figure 2:5-5).

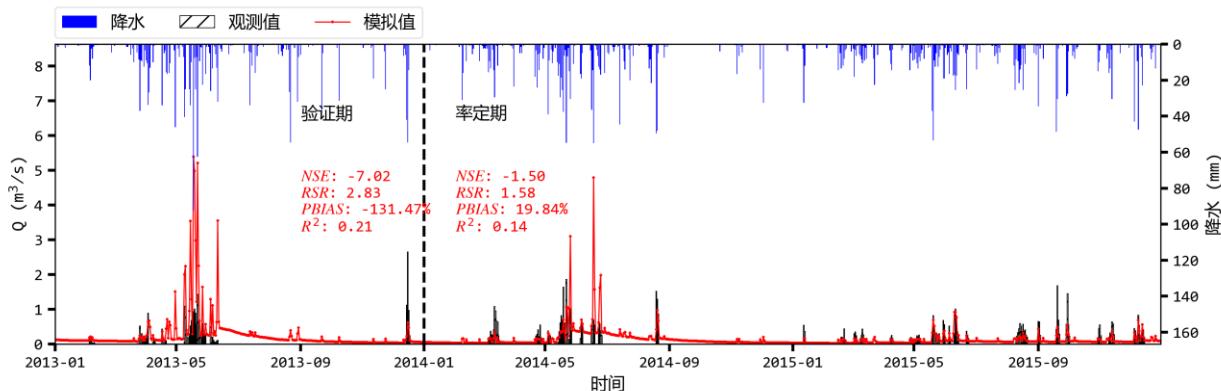


Figure 2:5-4 Calibration and validation of the simulated streamflow (Q , m^3/s) at the outlet of the Youwuzhen watershed derived from the demo Youwuzhen watershed model with default model parameters using Chinese font

```

13 [OPTIONAL_PARAMETERS]
14 # Calibration period (UTCTIME)
15 Cali_Time_start = 2014-01-01 00:00:00
16 Cali_Time_end = 2015-12-31 23:59:59
17 # Validation period (UTCTIME)
18 Vali_Time_start = 2013-01-01 00:00:00
19 Vali_Time_end = 2013-12-31 23:59:59
20
21 [OPTIONAL_MATPLOT_SETTINGS]
22 FIGURE_FORMATS = PDF,PNG
23 LANG_CN = True
24 FONT_TITLE = YaHei Mono

```

Figure 2:5-5 Modification of configuration content of postprocessing to plot both calibration and validation period using Chinese font

2:6. Parameter sensitivity analysis

Parameter sensitivity analysis is useful for identifying the most important or influential parameters for a specified simulation target (e.g., streamflow at the watershed outlet) (Zhan *et al.*, 2013). A typical sensitivity analysis procedure includes sampling from the value ranges of determined model inputs, evaluating the model with the generated samples and saving the interested outputs, and calculating sensitivity indices by various methods such as the Morris screening method (Morris, 1991) and FAST (Fourier Amplitude Sensitivity Test, Cukier *et al.*, 1978). The sensitivity analysis tool in SEIMS is organized as separated functions according to these steps, which means it is easy to incorporate any sampling method and sensitivity analysis method. The most time-consuming step is repetitive model evaluations, which is parallelized at model level by parallel job management based on [SCOOP](#) (Hold-Geoffroy *et al.*, 2014).

2:6.1 Simple usage

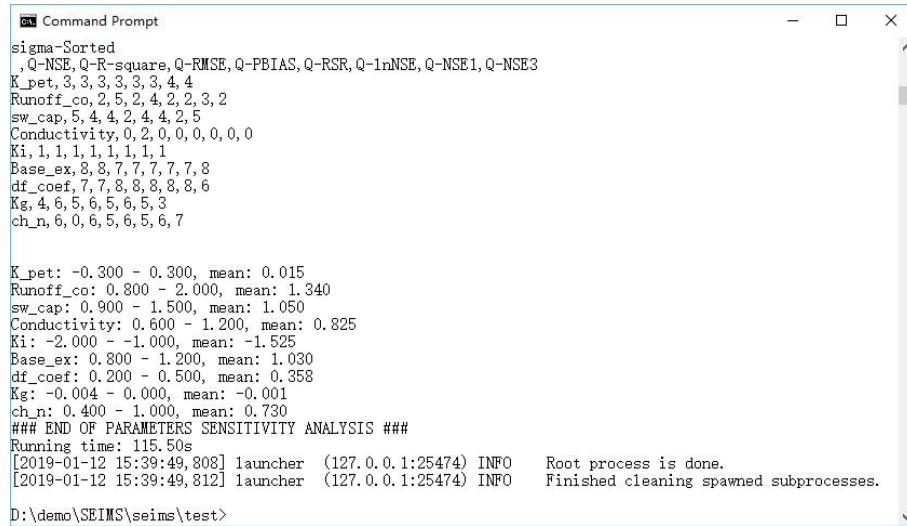
For simple usage, open a CMD window, enter the following commands to execute the predefined parameter sensitivity analysis of the Youwuzhen watershed model.

```
01 >cd D:\demo\SEIMS\seims\test
02 >D:
03 >python -m scoop -n 2 demo_parameters_sensitivity.py -name youwuzhen
```

The runtime logs of the parameter sensitivity analysis including the commands of each model run and the results of sensitivity analysis were showed in Figure 2:6-1.

```
D:\demo\SEIMS\seims\test>d:\demo\python27\python -m scoop -n 2 demo_parameters_sensitivity.py -name yo
youwuzhen
[2019-01-12 15:37:48,756] launcher INFO    SCOOP 0.7 1.1 on win32 using Python 2.7.15 (v2.7.15:ca079a
3ea3, Apr 30 2018, 16:30:26) [MSC v.1500 64 bit (AMD64)], API: 1013
[2019-01-12 15:37:48,757] launcher INFO    Deploying 2 worker(s) over 1 host(s).
[2019-01-12 15:37:48,759] launcher INFO    Worker distribution:
[2019-01-12 15:37:48,759] launcher INFO    127.0.0.1:      1 + origin
SEIMS binary location: D:\demo\SEIMS\bin
Demo data location: D:\demo\SEIMS\data\youwuzhen
Data preprocessing location: D:\demo\SEIMS\data\youwuzhen\workspace
### START TO PARAMETERS SENSITIVITY ANALYSIS ###
D:\demo\SEIMS\bin\seims_omp.exe -wp D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model -thr
ead 2 -lyr 1 -host 127.0.0.1 -port 27017 -sce 0 -cali 0 -id 0
D:\demo\SEIMS\bin\seims_omp.exe -wp D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model -thr
ead 2 -lyr 1 -host 127.0.0.1 -port 27017 -sce 0 -cali 39 -id 0
D:\demo\SEIMS\bin\seims_omp.exe -wp D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model -thr
ead 2 -lyr 1 -host 127.0.0.1 -port 27017 -sce 0 -cali 1 -id 0
D:\demo\SEIMS\bin\seims_omp.exe -wp D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model -thr
ead 2 -lyr 1 -host 127.0.0.1 -port 27017 -sce 0 -cali 2 -id 0
D:\demo\SEIMS\bin\seims_omp.exe -wp D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model -thr
ead 2 -lyr 1 -host 127.0.0.1 -port 27017 -sce 0 -cali 3 -id 0
D:\demo\SEIMS\bin\seims_omp.exe -wp D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model -thr
ead 2 -lyr 1 -host 127.0.0.1 -port 27017 -sce 0 -cali 4 -id 0
D:\demo\SEIMS\bin\seims_omp.exe -wp D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model -thr
ead 2 -lyr 1 -host 127.0.0.1 -port 27017 -sce 0 -cali 5 -id 0
D:\demo\SEIMS\bin\seims_omp.exe -wp D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model -thr
ead 2 -lyr 1 -host 127.0.0.1 -port 27017 -sce 0 -cali 6 -id 0
D:\demo\SEIMS\bin\seims_omp.exe -wp D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model -thr
```

(a)



```

Command Prompt
sigma-Sorted
,Q-NSE,Q-R-square,Q-RMSE,Q-PBIAS,Q-RSR,Q-1nNSE,Q-NSE1,Q-NSE3
K_pet,3,3,3,3,3,3,4,4
Runoff_co,2,5,2,4,2,2,3,2
sw_cap,5,4,4,2,4,4,2,5
Conductivity,0,2,0,0,0,0,0,0
Ki,1,1,1,1,1,1,1,1
Base_ex,8,8,7,7,7,7,7,8
df_coef,7,7,8,8,8,8,6
Kg,4,6,5,6,5,6,5,3
ch_n,6,0,6,5,6,5,6,7

K_pet: -0.300 - 0.300, mean: 0.015
Runoff_co: 0.800 - 2.000, mean: 1.340
sw_cap: 0.900 - 1.500, mean: 1.050
Conductivity: 0.600 - 1.200, mean: 0.825
Ki: -2.000 - -1.000, mean: -1.525
Base_ex: 0.800 - 1.200, mean: 1.030
df_coef: 0.200 - 0.500, mean: 0.358
Kg: -0.004 - 0.000, mean: -0.001
ch_n: 0.400 - 1.000, mean: 0.730
### END OF PARAMETERS SENSITIVITY ANALYSIS ###
Running time: 115.50s
[2019-01-12 15:39:49,808] launcher (127.0.0.1:25474) INFO Root process is done.
[2019-01-12 15:39:49,812] launcher (127.0.0.1:25474) INFO Finished cleaning spawned subprocesses.

D:\demo\SEIMS\seims\test>

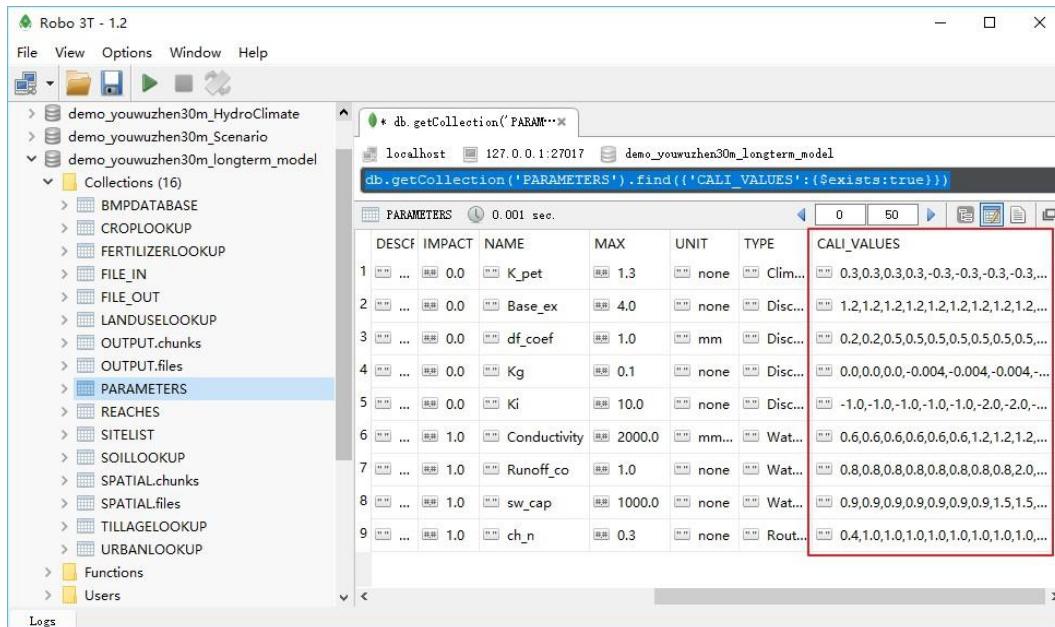
```

(b)

Figure 2:6-1 Beginning (a) and ending (b) runtime logs of the predefined parameter sensitivity analysis of the Youwuzhen watershed model

Firstly, the sampled calibration values of selected parameters were organized as float arrays and imported as `CALI_VALUES` field into the `PARAMETERS` collection of the main spatial database. The SEIMS-based watershed model under a combination of calibrated parameters can be evaluated with the `calibrationID` argument set to the corresponding array index (start from 0) (Figure 2:6-1a). The calibration values can be queried by the following command in the console window of Robo 3T (Figure 2:6-2).

```
01 db.getCollection('PARAMETERS').find({'CALI_VALUES':{$exists:true}})
```



DESCF	IMPACT	NAME	MAX	UNIT	TYPE	CALI_VALUES
1	...	K_pet	1.3	none	Clim...	0.3,0.3,0.3,0.3,-0.3,-0.3,-0.3,...
2	...	Base_ex	4.0	none	Disc...	1.2,1.2,1.2,1.2,1.2,1.2,1.2,1.2,...
3	...	df_coef	1.0	mm	Disc...	0.2,0.2,0.5,0.5,0.5,0.5,0.5,0.5,...
4	...	Kg	0.1	none	Disc...	0.0,0.0,0.0,-0.004,-0.004,-0.004,...
5	...	Ki	10.0	none	Disc...	-1.0,-1.0,-1.0,-1.0,-1.0,-2.0,-2.0,...
6	...	Conductivity	2000.0	mm...	Wat...	0.6,0.6,0.6,0.6,0.6,0.6,1.2,1.2,1.2,...
7	...	Runoff_co	1.0	none	Wat...	0.8,0.8,0.8,0.8,0.8,0.8,0.8,0.8,2.0,...
8	...	sw_cap	1000.0	none	Wat...	0.9,0.9,0.9,0.9,0.9,0.9,0.9,1.5,1.5,...
9	...	ch_n	0.3	none	Rout...	0.4,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,...

Figure 2:6-2 Sampled calibration values of selected parameters for the sensitivity analysis of the Youwuzhen watershed model

Then, all combinations of calibrated parameters were evaluated in parallel according to the workers' number of SCOOP, e.g., two workers were allocated in this simple usage (`-m scoop -n 2`). After the evaluation of each calibrated model, the interested model performance indicators will be calculated and saved. In current version of SEIMS, *NSE* (Nash-Sutcliffe efficiency), *RSR* (root mean square error-standard deviation ratio), and *PBIAS* (percent bias) (Moriasi *et al.*, 2007) were used as model performance indicators.

Finally, the sensitivity analysis method (Morris screening method in this demo) will be executed according to the above introduced model outputs. For example, Figure 2:6-3 showed the parameter sensitivity analysis result with respect to the *NSE* of the simulated streamflow (m^3s^{-1}) at the watershed outlet. Each point in the screening plot represents one parameter of the Youwuzhen watershed model. The larger the modified means μ^* (Campolongo *et al.*, 2007), the more sensitive the watershed response is to variation in the parameter (Yang, 2011; Zhan *et al.*, 2013). More results can be found in the output directory, e.g.,

`SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model\PSA_Morris_N4L2.`

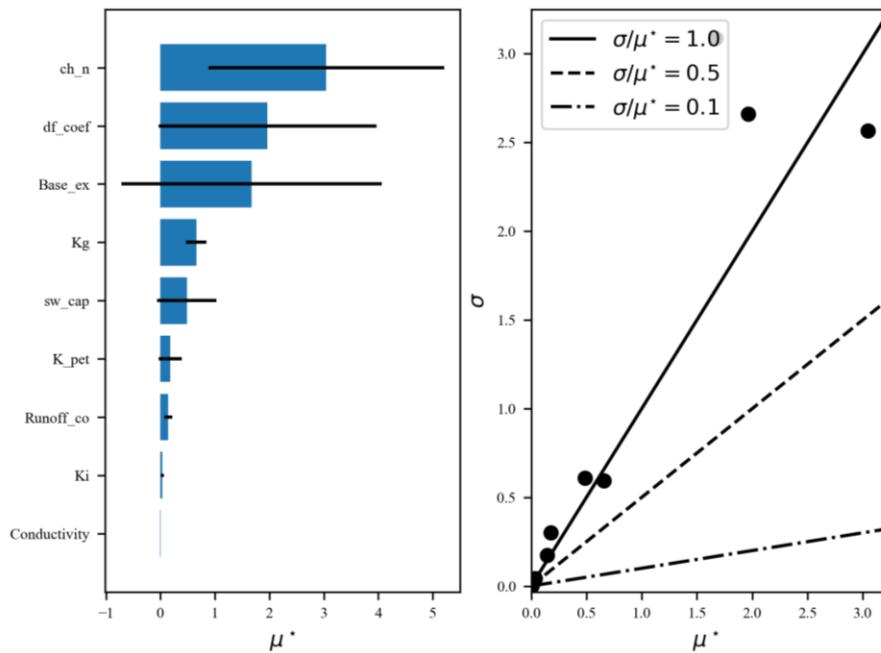


Figure 2:6-3 Parameter sensitivity analysis result with respect to the NSE of the simulated streamflow (m^3s^{-1}) at the watershed outlet

2:6.2 Configuration file of parameter sensitivity analysis

The configuration file of parameter sensitivity analysis, such as that of the Youwuzhen watershed model shown in Figure 2:6-4, includes four sections, i.e., SEIMS_Model, PARAMETERS, OPTIONAL_PARAMETERS, and OPTIONAL_MATPLOT_SETTING. Among them, OPTIONAL_PARAMETERS and OPTIONAL_MATPLOT_SETTING are optional. The names of sections and options should not be changed.

- **SEIMS_Model:** Basic settings of the SEIMS-based watershed model, see Section 2:4.3.2 for details. Note that the OpenMP version of SEIMS main program is used by default. However, the options of `version`, `MPI_BIN`, and `processNum` are supported if the MPI&OpenMP version is wanted.
- **PSA_PARAMETERS:** Basic settings for parameter sensitivity analysis.
 1. **evaluateParam:** Simulation target(s) to be evaluated for sensitivity analysis, e.g., `Q` for streamflow, `SED` for sediment, and `CH_TN` for total nitrogen. This option is similar to `PLOT_VARIABLES` of postprocessing in Section 2:5.2. Multiple parameters should be separated by comma.
 2. **paramRngDef:** Filename of the definition of parameters' range. The file follows the basic plain text file format of SEIMS. Each line indicates one parameter to be analyzed with the format of `NAME`, `lower_bound`, `upper_bound`. For example, `K_pet, -0.3, 0.3` means that the impact value of `K_pet` ranges from -0.3 to 0.3. The type of `CHANGE` is the same with that defined in `PARAMETERS` collection. Relative information please refer to Section 2:4.2.3.
 3. **PSA_Time_start:** Starting date time of the calculation of model performance for the sensitivity analysis with the format of `YYYY-MM-DD HH:MM:SS`.
 4. **PSA_Time_end:** Ending date time of the calculation of model performance for the sensitivity analysis with the format of `YYYY-MM-DD HH:MM:SS`.
- Sections of specific sensitivity analysis methods supported by [SALib](#). Currently, the Morris screening method including groups and optimal trajectories (`Morris_Method`) and Fourier Amplitude Sensitivity Test (`FAST_Method`) have been integrated and tested. In the future version of SEIMS, more methods and not limited to the methods of [SALib](#) should be integrated.
 1. **Morris_Method:**
 - `N`: The number of trajectories to generate.
 - `num_levels`: The number of grid levels. The resulting samples have a row number of $(D+1)*N$, and column number of `D`, where `D` is the number of parameters.
 - `optimal_trajectories`: The number of optimal trajectories to sample (between 2 and `N`) which can also be `None`.
 - `local_optimization`: Flag whether to use local optimization. Speeds up the process tremendously for bigger `N` and `num_levels`. If set to `False` brute force method is used.
 2. **FAST_Method:**
 - `N`: The number of samples to generate.

- M: The interference parameter, i.e., the number of harmonics to sum in the Fourier series decomposition, the default is 4.
- OPTIONAL_MATPLOT_SETTINGS: Plot settings for matplotlib, see Section 2:5.2.

```

01 [SEIMS_Model]
02 MODEL_DIR = D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model
03 BIN_DIR = D:\demo\SEIMS\bin
04 HOSTNAME = 127.0.0.1
05 PORT = 27017
06 threadsNum = 2
07 layeringMethod = 1
08 scenarioID = 0
09 # Simulation period (UTCTIME)
10 Sim_Time_start = 2012-01-01 00:00:00
11 Sim_Time_end = 2012-03-09 23:59:59
12
13 [PSA_Settings]
14 evaluateParam = Q
15 paramRngDef = morris_param_rng-Q.def
16 # Objective calculation period (UTCTIME)
17 PSA_Time_start = 2012-02-27 00:00:00
18 PSA_Time_end = 2012-03-09 23:59:59
19
20 [Morris_Method]
21 N = 4
22 num_levels = 2
23 optimal_trajectories = None
24 local_optimization = True
25
26 [OPTIONAL_MATPLOT_SETTINGS]
27 FIGURE_FORMATS = PDF,PNG
28 FONT_TITLE = Times New Roman
29 DPI = 300

```

Figure 2:6-4 Configuration content for parameter sensitivity analysis of the Youwuzhen watershed model

2:6.3 Advanced usage

The Python scripts of parameter sensitivity analysis are located in `SEIMS/seims/parameters_sensitivity`. The `main.py` is the entrance which can be executed through the unified format of running SEIMS Python scripts, e.g.,

```

01 >cd D:\demo\SEIMS\seims\parameters_sensitivity
02 >python main.py -ini
D:\demo\SEIMS\data\youwuzhen\workspace\sensitivity_analysis.ini .

```

2:7. Auto-Calibration

After parameter sensitivity analysis, a small number of the most sensitive parameters can be selected for auto-calibration. The value ranges of parameters may be the same with those defined in the parameter sensitivity analysis, or they may be narrowed by excluding values that result in an unacceptable model performance, such as in cases where NSE is less than zero.

In the current version of SEIMS, the NSGA-II algorithm (Non-dominated Sorting Genetic Algorithm II) was used for auto-calibration. During NSGA-II execution, the Latin-hypercube sampling method (Iman and Shortencarier, 1984) was used to generate initial samples of calibrated parameters. The crossover and mutation operations were similar to the original implementations in NSGA-II by Deb *et al.* (2002).

2:7.1 Simple usage

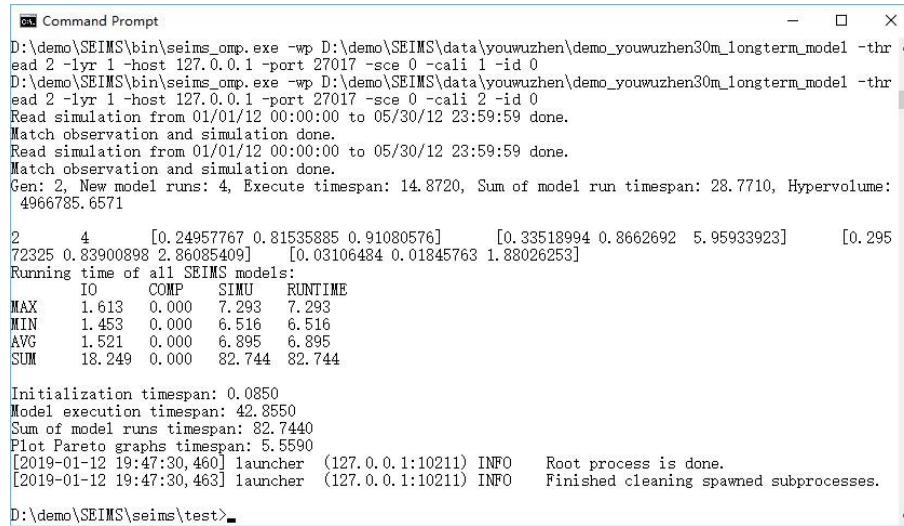
For simple usage, open a CMD window, enter the following commands to execute the auto-calibration of the streamflow modeling of the Youwuzhen watershed model.

```
01 >cd D:\demo\SEIMS\seims\test
02 >D:
03 >python -m scoop -n 2 demo_calibration.py -name youwuzhen
```

The runtime logs of auto-calibration including commands of each model run, the average model performance indicators of each generation, and the time-consuming were showed in Figure 2:7-1.

```
01 >cd D:\demo\SEIMS\seims\test>d:\demo\python27\python -m scoop -n 2 demo_calibration.py -name youwuzhen
[2019-01-12 19:46:27,388] launcher INFO SCOOP 0.7 1.1 on win32 using Python 2.7.15 (v2.7.15:ca079a
3ea3, Apr 30 2018, 16:30:26) [MSC v. 1500 64 bit (AMD64)], API: 1013
[2019-01-12 19:46:27,388] launcher INFO Deploying 2 worker(s) over 1 host(s).
[2019-01-12 19:46:27,390] launcher INFO Worker distribution:
[2019-01-12 19:46:27,391] launcher INFO 127.0.0.1: 1 + origin
SEIMS binary location: D:\demo\SEIMS\bin
Demo data location: D:\demo\SEIMS\data\youwuzhen
Data preprocessing location: D:\demo\SEIMS\data\youwuzhen\workspace
Population: 4, Generation: 2
Read observation data of Q from 01/01/12 00:00:00 to 05/30/12 23:59:59 done.
D:\demo\SEIMS\bin\seims_omp.exe -wp D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model -thr
ead 2 -1yr 1 -host 127.0.0.1 -port 27017 -sce 0 -cali 0 -id 0
D:\demo\SEIMS\bin\seims_omp.exe -wp D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model -thr
ead 2 -1yr 1 -host 127.0.0.1 -port 27017 -sce 0 -cali 3 -id 0
Read simulation from 01/01/12 00:00:00 to 05/30/12 23:59:59 done.
Match observation and simulation done.
Read simulation from 01/01/12 00:00:00 to 05/30/12 23:59:59 done.
D:\demo\SEIMS\bin\seims_omp.exe -wp D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model -thr
ead 2 -1yr 1 -host 127.0.0.1 -port 27017 -sce 0 -cali 1 -id 0
Match observation and simulation done.
D:\demo\SEIMS\bin\seims_omp.exe -wp D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model -thr
ead 2 -1yr 1 -host 127.0.0.1 -port 27017 -sce 0 -cali 2 -id 0
Read simulation from 01/01/12 00:00:00 to 05/30/12 23:59:59 done.
Match observation and simulation done.
Read simulation from 01/01/12 00:00:00 to 05/30/12 23:59:59 done.
Match observation and simulation done.
d:\demo\python27\lib\site-packages\matplotlib\font_manager.py:1331: UserWarning: findfont: Font family
```

(a)



```

Command Prompt
D:\demo\SEIMS\bin\seims_omp.exe -wp D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model -thr ^ead 2 -lyr 1 -host 127.0.0.1 -port 27017 -sce 0 -cali 1 -id 0
D:\demo\SEIMS\bin\seims_omp.exe -wp D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model -thr ^ead 2 -lyr 1 -host 127.0.0.1 -port 27017 -sce 0 -cali 2 -id 0
Read simulation from 01/01/12 00:00:00 to 05/30/12 23:59:59 done.
Match observation and simulation done.
Read simulation from 01/01/12 00:00:00 to 05/30/12 23:59:59 done.
Match observation and simulation done.
Gen: 2, New model runs: 4, Execute timespan: 14.8720, Sum of model run timespan: 28.7710, Hypervolume: 4966785.6571

2          4      [0.24957767 0.81535885 0.91080576]      [0.33518994 0.8662692 5.95933923]      [0.295
72325 0.83900898 2.86085409]      [0.03106484 0.01845763 1.88026253]
Running time of all SEIMS models:
    IO   COMP   SIMU   RUNTIME
MAX   1.613  0.000  7.293  7.293
MIN   1.453  0.000  6.516  6.516
AVG   1.521  0.000  6.895  6.895
SUM   18.249 0.000  82.744 82.744

Initialization timespan: 0.0850
Model execution timespan: 42.8550
Sum of model runs timespan: 82.7440
Plot Pareto graphs timespan: 5.5590
[2019-01-12 19:47:30,460] launcher (127.0.0.1:10211) INFO Root process is done.
[2019-01-12 19:47:30,463] launcher (127.0.0.1:10211) INFO Finished cleaning spawned subprocesses.

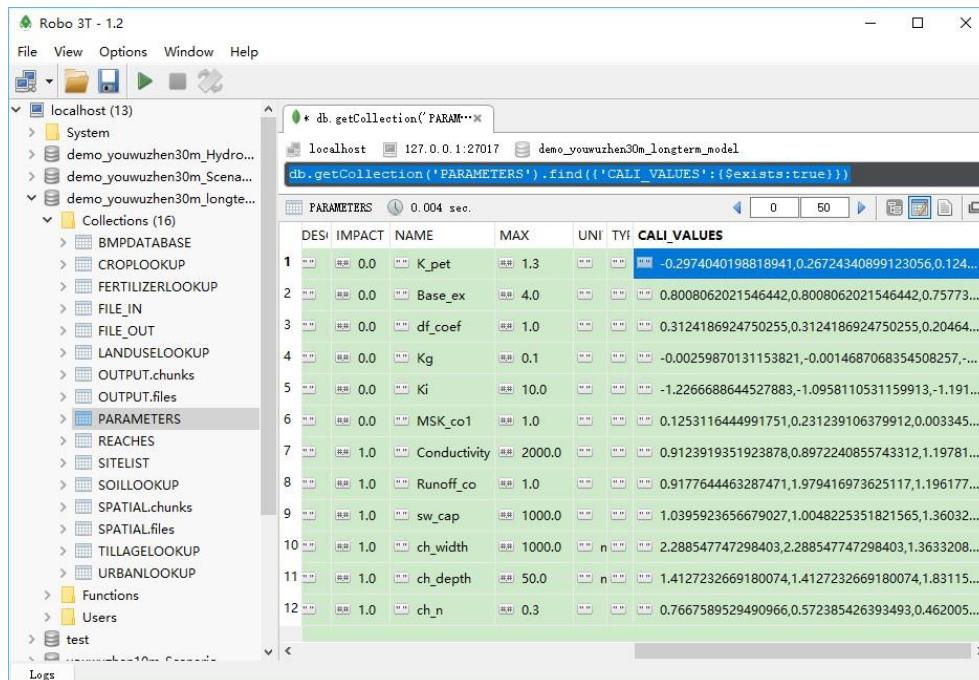
D:\demo\SEIMS\seims\test>_

```

(b)

Figure 2:7-1 Runtime logs of auto-calibration of the Youwuzhen watershed model

Similar to the parameter sensitivity analysis, the calibration values of selected parameters will be generated and imported as `CALI_VALUES` field into the `PARAMETERS` collection of the main spatial database for each generation during the optimization of auto-calibration (Figure 2:7-2).



DES	IMPACT	NAME	MAX	UNI	TYF	CALI_VALUES
1	0.0	K_pet	1.3			-0.2974040198818941,0.26724340899123056,0.124...
2	0.0	Base_ex	4.0			0.8008062021546442,0.8008062021546442,0.75773...
3	0.0	df_coef	1.0			0.3124186924750255,0.3124186924750255,0.20464...
4	0.0	Kg	0.1			-0.00259870131153821,-0.0014687068354508257,-...
5	0.0	Ki	10.0			-1.2266688644527883,-1.0958110531159913,-1.191...
6	0.0	MSK_co1	1.0			0.1253116444991751,0.231239106379912,0.003345...
7	1.0	Conductivity	2000.0			0.9123919351923878,0.8972240855743312,1.19781...
8	1.0	Runoff_co	1.0			0.9177644463287471,1.979416973625117,1.19617...
9	1.0	sw_cap	1000.0			1.0395923656679027,1.0048225351821565,1.36032...
10	1.0	ch_width	1000.0	n		2.28854774298403,2.28854774298403,1.3633208...
11	1.0	ch_depth	50.0	n		1.4127232669180074,1.4127232669180074,1.83115...
12	1.0	ch_n	0.3			0.7667589529490966,0.572385426393493,0.462005...

Figure 2:7-2 Calibration values of selected parameters generated during the optimization of auto-calibration of the Youwuzhen watershed model

The results of the auto-calibration can be found in `SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model\CALI_NSGA2_Gen_2_Pop_4`. The near-optimal Pareto solutions plotted as a scatter plot can give a simple and direct interpretation of the calibration optimization processes. For example, Figure 2:7-3 showed the near optimal Pareto solutions of the 1st and 2nd generations with the objectives of maximize the *NSE* and minimize the *RSR* of the simulated streamflow at the watershed outlet, which indicated an evolutionary directory to maximum *NSE* and minimum *RSR*.

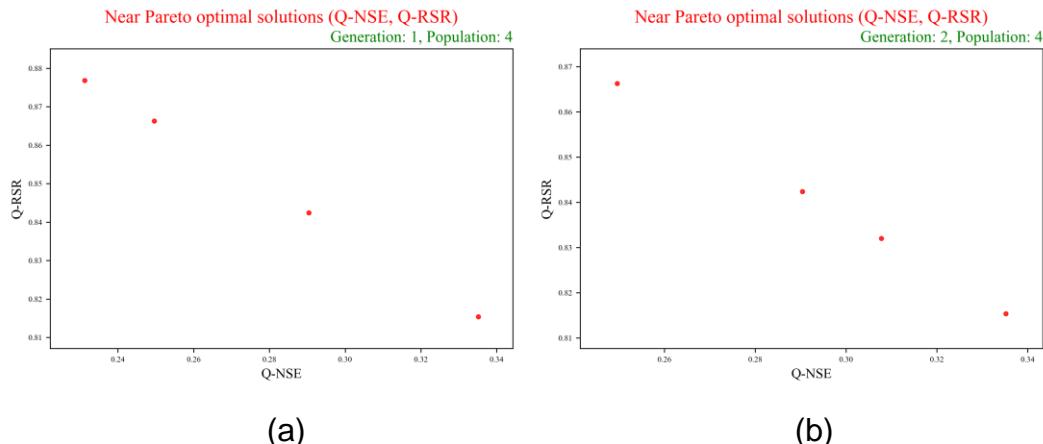


Figure 2:7-3 Near optimal Pareto solutions of the 1st (a) and 2nd (b) generations with the objectives of maximize the *NSE* and minimize the *RSR* of the simulated streamflow at the watershed outlet

In the meantime, to have a better aware of the calibration results, one best solution with the maximum *NSE* of the calibration period was plotted for each generation (red dash line in Figure 2:7-4) together with the 2.5% and 97.5% levels of the cumulative distribution of the output variables (gray band in Figure 2:7-4), i.e., streamflow at the watershed outlet in this demo.

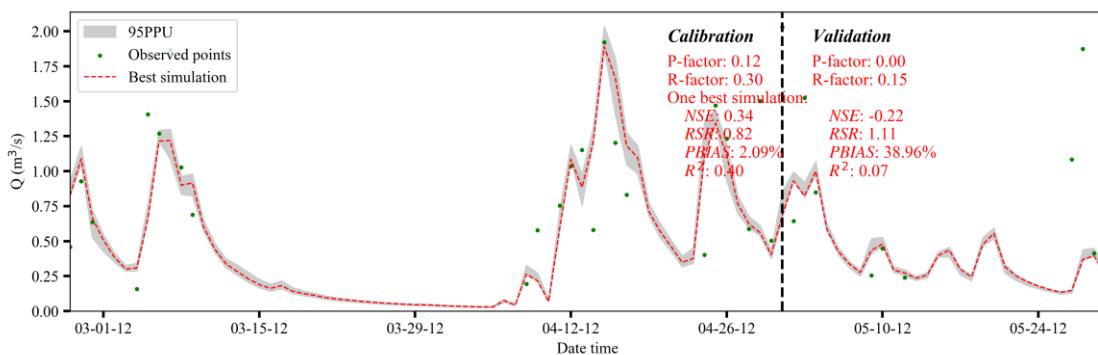


Figure 2:7-4 Auto-calibration result of streamflow (Q , m^3/s) simulation of the 2nd generation at the calibration and validation periods

2:7.2 Configuration file of auto-calibration

The configuration file of auto-calibration, such as that of the Youwuzhen watershed model shown in Figure 2:7-5, includes four sections, i.e., `SEIMS_Model`, `CALI_Settings`, `NSGA2`, and `OPTIONAL_MATPLOT_SETTING`. Among them, `OPTIONAL_MATPLOT_SETTING` is optional. The names of sections and options should not be changed.

- `SEIMS_Model`: Basic settings of the SEIMS-based watershed model, see Section 2:4.3.2 for details. Note that the OpenMP version of SEIMS main program is used by default. However, the options of `version`, `MPI_BIN`, and `processNum` are supported if the MPI&OpenMP version is wanted.
- `CALI_Settings`: Basic settings for auto-calibration.
 1. `paramRngDef`: Filename of the definition of parameters' range for calibration. The format is the same with `paramRngDef` of parameter sensitivity analysis (Section 2:6.2).
 2. `Cali_Time_start`: Starting date time of the calibration period for model calibration with the format of `YYYY-MM-DD HH:MM:SS`.
 3. `Cali_Time_end`: Ending date time of the calibration period for model calibration with the format of `YYYY-MM-DD HH:MM:SS`.
 4. `Vali_Time_start`: (Optional) Starting date time of the validation period with the format of `YYYY-MM-DD HH:MM:SS` for visualization only to explore the effective of the calibrated parameters, e.g., Figure 2:7-4.
 5. `Vali_Time_end`: (Optional) Ending date time of the validation period with the format of `YYYY-MM-DD HH:MM:SS`. The `Vali_Time_start` and `Vali_Time_end` should be specified simultaneously.
- Sections of specific optimization methods for auto-calibration. Currently, the NSGA-II algorithm has been integrated and tested. In the future version of SEIMS, more methods should be integrated.
 1. `NSGA2`: Parameter settings of NSGA-II algorithm.

`GenerationsNum`: Maximum number of generations.
`PopulationSize`: Initial population size, i.e., the number of individuals.
`CrossoverRate`: Crossover probability, ranges from 0 to 1. A larger crossover rate indicates a higher possibility to take place the crossover operation between two parent individuals.
`MutateRate`: Mutate probability.
`MaxMutatePerc`: Maximum percent of genes to be mutated.
`SelectRate`: Selection rate of the evaluated and sorted individuals for each generation which is known as near-optimal Pareto solutions.
- `OPTIONAL_MATPLOT_SETTINGS`: Plot settings for matplotlib, see Section 2:5.2.

```
01 [SEIMS_Model]
02 MODEL_DIR = D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model
03 BIN_DIR = D:\demo\SEIMS\bin
04 HOSTNAME = 127.0.0.1
05 PORT = 27017
06 threadsNum = 2
07 layeringMethod = 1
08 scenarioID = 0
09 # Simulation period (UTCTIME)
10 Sim_Time_start = 2012-01-01 00:00:00
11 Sim_Time_end = 2012-05-30 23:59:59
12
13 [CALI_Settings]
14 # Parameters and ranges
15 paramRngDef = cali_param_rng-Q.def
16 # Calibration period (UTCTIME)
17 Cali_Time_start = 2012-02-27 00:00:00
18 Cali_Time_end = 2012-04-30 23:59:59
19 # Validation period (UTCTIME)
20 Vali_Time_start = 2012-05-01 00:00:00
21 Vali_Time_end = 2012-05-30 23:59:59
22
23 [NSGA2]
24 GenerationsNum = 2
25 PopulationSize = 4
26 CrossoverRate = 0.8
27 MaxMutatePerc = 0.2
28 MutateRate = 0.1
29 SelectRate = 1.0
30
31 [OPTIONAL_MATPLOT_SETTINGS]
32 FIGURE_FORMATS = PDF,PNG
33 LANG_CN = False
34 FONT_TITLE = Times New Roman
35 TITLE_FONTSIZE = 16
36 LEGEND_FONTSIZE = 12
37 TICKLABEL_FONTSIZE = 12
38 AXISLABEL_FONTSIZE = 12
39 LABEL_FONTSIZE = 14
40 DPI = 300
```

Figure 2:7-5 Configuration content for auto-calibration of the Youwuzhen watershed model based on NSGA-II

2:7.3 Advanced usage

The Python scripts of auto-calibration are located in `SEIMS/seims/calibration`. The `main_nsga2.py` is the entrance for the auto-calibration based on NSGA-II algorithm, which can be executed through the unified format of running SEIMS Python scripts, e.g.,

```
01 >cd D:\demo\SEIMS\seims\calibration
02 >python main_nsga2.py -ini
D:\demo\SEIMS\data\youwuzhen\workspace\calibration.ini .
```

2:8. BMP scenarios analysis

Different best management practices (or beneficial management practices, BMPs for short) scenarios (i.e., spatial configurations of multiple BMPs) at the watershed scale may have significantly different environmental effectiveness, economic cost-benefit, and practicality. It is valuable for decision-making of integrated watershed management to assess the environmental effectiveness and economic cost-benefit of watershed BMP scenarios and then propose optimal ones. During the BMP scenarios analysis, each individual BMP scenario is created by automatically selecting and allocating BMPs on spatial units (known as BMP configuration units). Then the effects of the scenario on watershed behavior are simulated by watershed models. The simulation result is the basis of the automatic spatial optimization of BMP scenarios.

As a demo, one of the experiments conducted in Zhu *et al.* (2019b) is used in this section. The hydrologically connected fields were selected as the BMPs configuration units (see Section 2:3.2 for the preparation). Four spatially explicit BMPs were considered and the BMP configuration strategy based on expert knowledge of upstream–downstream relationships (Wu *et al.*, 2018) was adopted. The NSGA-II algorithm was integrated to achieve the spatial optimization of BMP scenarios. More information please refers to Zhu *et al.* (2019b).

2:8.1 Simple usage

For simple usage, open a CMD window, enter the following commands to execute the predefined BMP scenarios analysis of the Youwuzhen watershed model.

```
01 >cd D:\demo\SEIMS\seims\test
02 >D:
03 >python -m scoop -n 2 demo_scenario_analysis.py -name youwuzhen
```

The runtime logs of scenario analysis including commands of each model run, the average objective values (i.e., environmental effectiveness and net-cost) of each generation, and the time-consuming were showed in Figure 2:8-1.

```
Command Prompt
D:\demo\SEIMS\seims\test>d:\demo\python27\python -m scoop -n 2 demo_scenario_analysis.py -name youwuzhen
[2019-01-12 21:30:59,657] launcher INFO    SCOOP 0.7 1.1 on win32 using Python 2.7.15 (v2.7.15:ca079a
Sea3, Apr 30 2018, 16:30:26) [MSC v.1500 64 bit (AMD64)], API: 1013
[2019-01-12 21:30:59,657] launcher INFO    Deploying 2 worker(s) over 1 host(s).
[2019-01-12 21:30:59,658] launcher INFO    Worker distribution:
[2019-01-12 21:30:59,660] launcher INFO        127.0.0.1:      1 + origin
SEIMS binary location: D:\demo\SEIMS\bin
Demo data location: D:\demo\SEIMS\data\youwuzhen
Data preprocessing location: D:\demo\SEIMS\data\youwuzhen\workspace
### START TO SCENARIOS OPTIMIZING ###
Scenario ID: 110242856, running SEIMS model...
D:\demo\SEIMS\bin\seims_omp.exe -wp D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model -thr
ead 2 -lyr 1 -host 127.0.0.1 -port 27017 -sce 110242856 -id 0
Delete scenario: 110242856 in MongoDB completed!
The environment effectiveness value of the base scenario is 3236159.86
Population: 4, Generation: 2
BMPs configure unit: CONNFIELD, configuration method: UPDOWN
Scenario ID: 129054233, running SEIMS model...
D:\demo\SEIMS\bin\seims_omp.exe -wp D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model -thr
ead 2 -lyr 1 -host 127.0.0.1 -port 27017 -sce 129054233 -id 0
Scenario ID: 121309691, running SEIMS model...
D:\demo\SEIMS\bin\seims_omp.exe -wp D:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model -thr
ead 2 -lyr 1 -host 127.0.0.1 -port 27017 -sce 121309691 -id 0
Delete scenario: 129054233 in MongoDB completed!
Delete scenario: 121309691 in MongoDB completed!
Scenario ID: 939034100, running SEIMS model...
```

(a)

```

Command Prompt
Gen: 2, New model runs: 2, Execute timespan: 12.3440, Sum of model run timespan: 12.0170, Hypervolume: 31.1922
2      2      [3.05757000e+01 1.89988303e-02] [71.83413     0.13134967]      [54.2391975   0.061615
48] [14.90253584  0.04197265]
Running time of all SEIMS models:
IO    COMP   SIMU   RUNTIME
MAX   2.080  0.000  8.979  8.979
MIN   1.553  0.000  5.851  5.851
AVG   1.914  0.000  7.843  7.843
SUM   19.142 0.000  78.427 78.427

Initialization timespan: 0.3190
Model execution timespan: 46.6630
Sum of model runs timespan: 78.4270
Plot Pareto graphs timespan: 1.7810
gen   evals   min                           max                           avg
std
0      4      [3.84449400e+01 2.48242644e-02] [63.96489     0.12609996]      [51.44814   0.059293
36] [9.73100338  0.03942842]
1      4      [3.05757000e+01 1.89988303e-02] [71.83413     0.13134967]      [55.80225   0.081071]
[15.50075626  0.04876756]
2      2      [3.05757000e+01 1.89988303e-02] [71.83413     0.13134967]      [54.2391975   0.061615
48] [14.90253584  0.04197265]
Running time: 56.70s
[2019-01-12 21:32:00,526] launcher (127.0.0.1:29364) INFO Root process is done.
[2019-01-12 21:32:00,530] launcher (127.0.0.1:29364) INFO Finished cleaning spawned subprocesses.

```

(b)

Figure 2:8-1 Runtime logs of BMP scenarios analysis of the Youwuzhen watershed model

Similar to the auto-calibration based on NSGA-II algorithm, the automatically generated BMP scenarios were imported into the `BMP_SCENARIOS` collection of the scenario database for each generation during the BMP scenarios optimization (Figure 2:8-2).

_id	NAME	COLLECTION	BMPID	LOCATION	DISTRIBUTION	SUBSCENARIO
1	BASE	AREAL_S...	12	33	RASTER LANDUSE	0
2	S240086030	AREAL_S...	17	45-52-55...	RASTER FIELDS_15	1.0
3	S240086030	AREAL_S...	17	9-15-16-2...	RASTER FIELDS_15	2.0
4	S240086030	AREAL_S...	17	6-8-22-43...	RASTER FIELDS_15	3.0
5	S240086030	AREAL_S...	17	4-18-39-5...	RASTER FIELDS_15	4.0
6	S240086030	PLANT...	12	33	RASTER LANDUSE	0
7	S158895241	AREAL_S...	17	4-18-24-5...	RASTER FIELDS_15	1.0
8	S158895241	AREAL_S...	17	9-20-22-2...	RASTER FIELDS_15	2.0
9	S158895241	AREAL_S...	17	8-42-45-4...	RASTER FIELDS_15	3.0
10	S158895241	AREAL_S...	17	16-25-27-...	RASTER FIELDS_15	4.0

Figure 2:8-2 BMP scenarios generated during the spatial optimization of BMP scenarios

The results of the BMP scenarios analysis can be found in
SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model\SA_NSGA2_CONNFILE_UPDOWN_Gen_2_Pop_4 including:

- **Pareto_Economy-Environment** folder: Near optimal Pareto plots, e.g., Figure 2:8-3 showed the near optimal Pareto solutions of the 1st and 2nd generations with the objectives of maximizing the environmental effectiveness and minimizing the economic net-cost.
 - **Scenarios** folder: The BMP scenario information in plain text file and the corresponding spatial distribution raster file. For example, one BMP scenario information of the 2nd generation is as follows.

Scenario ID: 125931440

Gene number: 72

Scenario items:

_id	NAME	COLLECTION	BMPID	LOCATION	DISTRIBUTION	SUBSCENARIO	ID
481	S125931440	AREAL_STRUCT_MANAGEMENT	17	44-45-64	RASTER FIELDS_15	1	125931440
482	S125931440	AREAL_STRUCT_MANAGEMENT	17	21-35-40-53-70	RASTER FIELDS_15	2	125931440
483	S125931440	AREAL_STRUCT_MANAGEMENT	17	60-67	RASTER FIELDS_15	3	125931440
484	S125931440	AREAL_STRUCT_MANAGEMENT	17	4-6-12-16-34-38-41-42-50-68	RASTER FIELDS_15	4	125931440

125931440 8 125931440 PLANT MANAGEMENT 12 33 RASTER LANDUSE

483 3123931446
Effectiveness:

Effectiveness:
economy: E7 712680

economy: 57.712680

The gene number is 72 which equals to the spatial units number of FIELDS_11 types that configured on BMP configurations. Each BMP type has one scenario. BMP scenario was shown in Figure 2:8-4.

- `runtime.log` file: Recoding the near optimal Pareto solutions of each generation.
 - `hypervolume.txt` file: The hypervolume index of each generation.

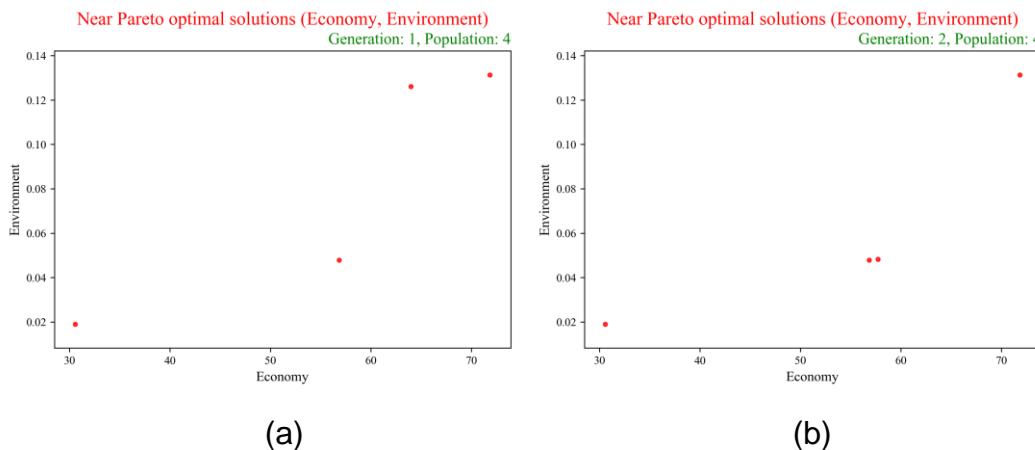


Figure 2:8-3 Near optimal Pareto solutions of the 1st (a) and 2nd (b) generations with the objectives of maximizing the environmental effectiveness and minimizing the economic net-cost

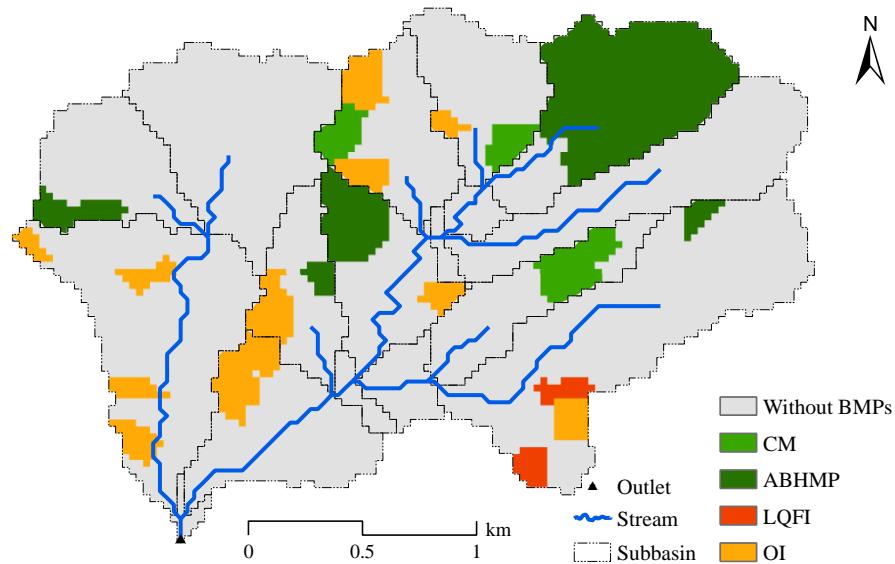


Figure 2:8-4 Spatial distribution of the selected BMP scenario of the 2nd generation of the BMP scenarios optimization

2:8.2 Configuration file of scenario analysis

The configuration file of BMP scenarios analysis, such as that of the Youwuzhen watershed model shown in Figure 2:8-5, includes five sections, i.e., `SEIMS_Model`, `Scenario_Common`, `BMPs`, `NSGA2`, and `OPTIONAL_MATPLOT_SETTING`. Among them, `OPTIONAL_MATPLOT_SETTING` is optional. The names of sections and options should not be changed.

- `SEIMS_Model`: Basic settings of the SEIMS-based watershed model, see Section 2:4.3.2 for details. Note that the OpenMP version of SEIMS main program is used by default. However, the options of `version`, `MPI_BIN`, and `processNum` are supported if the MPI&OpenMP version is wanted.
- `Scenario_Common`: Common settings for scenarios analysis.
 1. `Eval_Time_start`: Starting date time of model evaluation with the format of `YYYY-MM-DD HH:MM:SS`.
 2. `Eval_Time_end`: Ending date time of model evaluation with the format of `YYYY-MM-DD HH:MM:SS`.
 3. `worst_economy`: Economic net-cost under the circumstance of worst scenario.
 4. `worst_environment`: Environmental effectiveness under the circumstance of worst scenario. The `worst_economy` and `worst_environment` are set for the calculation of hypervolume index.
 5. `runtime_years`: For the BMP scenarios analysis based on the evaluation of long-term environmental effectiveness of BMPs, SEIMS assumed that BMPs can reach relative stability after several years of maintenance from their first establishment.

Therefore, the `runtime_years` is specified for the evaluation of BMP scenario cost model.

- 6. `Export_scenario_txt`: (Optional) Output each BMP scenario as plain text file (True) or not (False). The default is False.
- 7. `Export_scenario_tif`: (Optional) Output the spatial distribution of each BMP scenario as GeoTiff raster file (True) or not (False). The default is False.
- **BMPs**: Application specific BMPs settings.
 1. `BMPs_info`: Information of selected BMP type for BMP scenarios analysis. The JSON format is used. The key is `BMPID` (see Section 2:2.7.1) and the value is another JSON string of the corresponding field-values (available fields can be found in Table 2:2-8). Note that, the data type of the key MUST be string, although `BMPID` is integer. The `DISTRIBUTION` and `LOCATION` fields can be omitted which will be defined in `BMPs_cfg_units`.

```
01 BMPs_info = {"17":{  
02     "COLLECTION": "AREAL_STRUCT_MANAGEMENT",  
03     "SUBSCENARIO": [1, 2, 3, 4]  
04 }  
05 }
```

2. `BMPs_retain`: Retained BMP types for generated BMP scenarios during the optimization. The format is the same with `BMPs_info`.

```
01 BMPs_retain = {"12":{  
02     "COLLECTION": "PLANT_MANAGEMENT",  
03     "DISTRIBUTION": "RASTER|LANDUSE",  
04     "LOCATION": "33",  
05     "SUBSCENARIO": 0  
06 }  
07 }
```

3. `Eval_info`: Information of model evaluation, including the `OUTPUTID` and the corresponding output filename (`ENVEVAL`) and the base value of environmental variable (`BASE_ENV`), e.g., the annual total amount of soil erosion. If the `BASE_ENV` is set to -9999, the base scenario will be firstly evaluated before the BMP scenarios analysis.

```
01 Eval_info = {  
02     "OUTPUTID": "SED_DL",  
03     "ENVEVAL": "SED_DL_SUM.tif",  
04     "BASE_ENV": -9999  
05 }
```

4. `BMPs_cfg_units`: Information of BMP configuration units which also follows the JSON format. The `UNITJSON` is a file that describes the basic information of each spatial unit and spatial relationships between spatial units, e.g., the upstream and downstream relationships. In the current version, the hydrologic response units (HRUs, HRU), spatially explicit HRUs (EXPLICITHRU), hydrologically connected fields (CONNFIELD), and slope position units (SLPPOS) are supported. More details please refer to Zhu et al. (2019b).

```
01 BMPs_cfg_units = {"CONNFIELD": {  
02     "DISTRIBUTION": "RASTER|FIELDS_15",  
03     "UNITJSON": "connected_field_units_updown_15.json"  
04 }  
05 }
```

5. **BMPs_cfg_method**: BMP configuration strategy used according to the characteristics of BMP configuration units, such as randomly strategy (RAND), strategy with knowledge on the suitable landuse types/slope positions of individual BMPs (SUIT), strategy based on expert knowledge of upstream–downstream relationships (UPDOWN, Wu et al. [2018]), and strategy with domain knowledge on the spatial relationships between BMPs and slope positions along the hillslope (HILLSLP, Qin et al. [2018]). More details please refer to Zhu et al. (2019b).
- Sections of specific optimization methods for scenario analysis. Currently, the NSGA-II algorithm has been integrated and tested. In the future version of SEIMS, more methods should be integrated.
 1. **NSGA2**: Parameter settings of NSGA-II algorithm.
 - GenerationsNum**: Maximum number of generations.
 - PopulationSize**: Initial population size, i.e., the number of individuals.
 - CrossoverRate**: Crossover probability, ranges from 0 to 1. A larger crossover rate indicates a higher possibility to take place the crossover operation between two parent individuals.
 - MutateRate**: Mutate probability.
 - MaxMutatePerc**: Maximum percent of genes to be mutated.
 - SelectRate**: Selection rate of the evaluated and sorted individuals for each generation which is known as near-optimal Pareto solutions.
 - **OPTIONAL_MATPLOT_SETTINGS**: Plot settings for matplotlib, see Section 2:5.2.

```

01 [SEIMS_Model]
02 MODEL_DIR = C:\z_code\Hydro\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model
03 BIN_DIR = C:\z_code\Hydro\SEIMS\bin
04 HOSTNAME = 127.0.0.1
05 PORT = 27017
06 threadsNum = 2
07 layeringMethod = 1
08 scenarioID = 0
09 # Simulation period (UTCTIME)
10 Sim_Time_start = 2012-01-01 00:00:00
11 Sim_Time_end = 2012-05-30 23:59:59
12
13 [Scenario_Common]
14 Eval_Time_start = 2012-02-27 00:00:00
15 Eval_Time_end = 2012-04-30 23:59:59
16 worst_economy = 300.
17 worst_environment = 0.
18 runtime_years = 8
19 export_scenario_txt = True
20 export_scenario_tif = True
21
22 [BMPs]
23 BMPs_info = {"17": {"COLLECTION": "AREAL_STRUCT_MANAGEMENT", "SUBSCENARIO": [1, 2, 3, 4]}}
24 BMPs_retain = {"12": {"COLLECTION": "PLANT_MANAGEMENT", "DISTRIBUTION": "RASTER|LANDUSE",
"LOCATION": "33", "SUBSCENARIO": 0}}
25 Eval_info = {"OUTPUTID": "SED_DL", "ENVEVAL": "SED_DL_SUM.tif", "BASE_ENV": -9999}
26 BMPs_cfg_units = {"CONNFIELD": {"DISTRIBUTION": "RASTER|FIELDS_15", "UNITJSON":
"connected_field_units_updown_15.json"}}
27 BMPs_cfg_method = UPDOWN
28
29 [NSGA2]
30 GenerationsNum = 2
31 PopulationSize = 4
32 CrossoverRate = 1.0
33 MaxMutatePerc = 0.2
34 MutateRate = 1.0
35 SelectRate = 1.0
36
37 [OPTIONAL_MATPLOT_SETTINGS]
38 FIGURE_FORMATS = PDF,PNG
39 LANG_CN = False
40 FONT_TITLE = Times New Roman
41 TITLE_FONTSIZE = 16
42 LEGEND_FONTSIZE = 12
43 TICKLABEL_FONTSIZE = 12
44 AXISLABEL_FONTSIZE = 12
45 LABEL_FONTSIZE = 14
46 DPI = 300

```

Figure 2:8-5 Configuration content for BMP scenario analysis of the Youwuzhen watershed model based on NSGA-II

2:8.3 Advanced usage

The Python scripts of scenarios analysis are located in `SEIMS/seims/scenario_analysis/spatialunits`. The `main_nsga2.py` is the entrance for the scenarios analysis based on NSGA-II algorithm, which can be executed through the unified format of running SEIMS Python scripts, e.g.,

```
01 >cd D:\demo\SEIMS\seims\scenario_analysis\spatialunits  
02 >python main_nsga2.py -ini  
D:\demo\SEIMS\data\youwuzhen\workspace\scenario_analysis.ini .
```


Section 3. Design and implementation

This section is intended to give a detailed introduction of the design and implementation of SEIMS, which will be extended from Zhu et al. (2019a).

In the current pre-released version, this section is not yet finished, please refers to Zhu et al. (2019a) to get the necessary information.

3:1. Overall design of SEIMS

The overall architecture of SEIMS was designed as shown in Figure 3:1-1, and consists mainly of the SEIMS module library based on the modular structure, the SEIMS main programs (i.e., OpenMP version and MPI&OpenMP version), the watershed database, and utility tools for watershed model applications (such as parameter sensitivity analysis tool, and auto-calibration tool). The parallel computing middleware at multiple levels is implemented at the basic-unit level in SEIMS modules, the subbasin level in the MPI&OpenMP version of SEIMS main program, and the model level in watershed model applications, respectively. A SEIMS-based model consists of one SEIMS main program, several customized SEIMS modules, and the watershed database. Each SEIMS module inherits from the base module class (i.e., `SimulationModule`) with standard and concise interfaces (e.g., `SetData`, `GetData`, and `Execute` functions) and depends on base modules such as `I/O` module (Figure 3:1-1). The basic-unit level parallelization is achieved using OpenMP in the execution function of each SEIMS module. The basic version of SEIMS main program, which is the OpenMP version, is responsible for loading a set of user-configured modules to build and execute a simulation workflow. The MPI&OpenMP version of SEIMS main program uses the subbasin level domain decomposition and task scheduling to create instances of the OpenMP versioned SEIMS main program for each individual subbasins and distribute them among different computing nodes with MPI-based communication, so to achieve subbasin level parallel computing. Watershed model applications that requires numerous model runs (e.g., sensitivity analysis of a watershed model) are parallelized at model level based on job management.

With the support of parallel computing, SEIMS is compatible with common operating systems (such as Windows® and Linux®) and parallel computing platforms such as personal computers with multi-core CPU (Central Processing Unit) and SMP clusters..

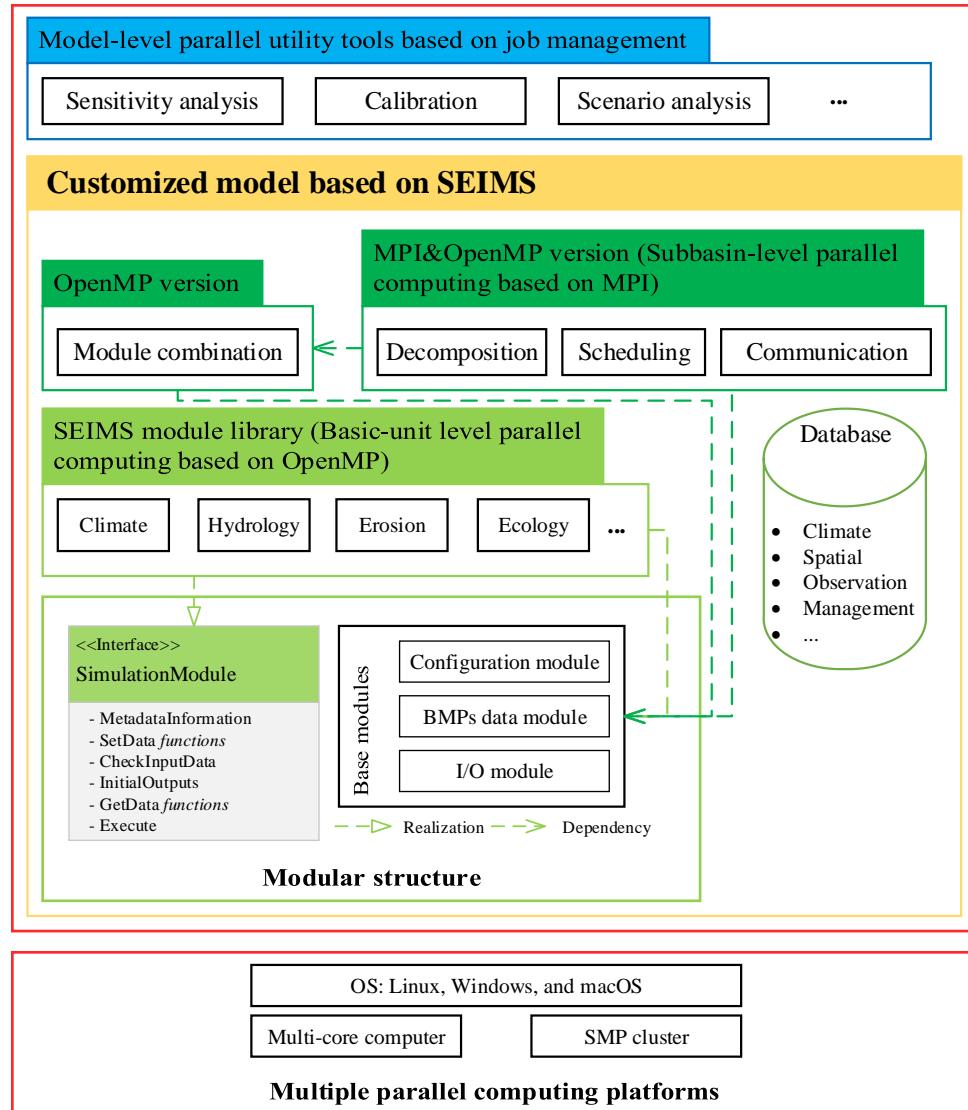


Figure 3:1-1 Architecture of Spatially Explicit Integrated Modeling System (SEIMS) which consists of the SEIMS module library, two versions of SEIMS main programs (i.e., OpenMP version and OpenMP&MPI version), the watershed database, and utility tools for watershed model applications, and can be conducted on multiple parallel computing platforms.

Section 4. Write your own code

The section presents a simple demo of developing a new module of watershed subprocess based on the SEIMS module interfaces.

4:1. Develop a new module of one watershed process

Generally, it requires three main steps to develop a new module for a specific watershed subprocess based on the SEIMS module interfaces.

- *Review the related literature and/or source code (if exists) of the watershed subprocess algorithm and figure out the input and output parameters of the algorithm as well as the specific simulation formula (or the execution code).*
- *Create new SEIMS module based on the module template and CMake structure.*
- *Open the SEIMS solution in Visual Studio or other IDEs and start to code following the recommended order of APIs.*

In this section, the simulation of potential evapotranspiration using Hargreaves method was selected as an example to demonstrate these steps.

4:1.1 Review literature and/or existing model code

Evapotranspiration is a collective term that includes all processes by which water at the earth's surface is converted to water vapor (Neitsch et al., 2011). It includes evaporation from the plant canopy, transpiration, sublimation, and evaporation from the soil. An accurate estimation of evapotranspiration is critical in the assessment of water resources and the impact of climate and land use change on those resources.

The Hargreaves method was originally derived from eight years of cool-season Alta fescue grass lysimeter data from Davis, California (Hargreaves, 1975). The Hargreaves method was implemented in the SWAT model with the improved equation (Hargreaves, 1985):

$$\lambda E_o = \text{petco} \cdot H_0 \cdot (T_{\max} - T_{\min})^{0.5} \cdot (T_{\text{avg}} + 17.8)$$

where λ is the latent heat of vaporization (MJ kg^{-1}), E_o is the potential evapotranspiration (mm d^{-1}), petco is coefficient related to radiation which ranges from 0.0019 to 0.0032, H_0 is the extraterrestrial radiation ($\text{MJ m}^{-2} \text{d}^{-1}$), T_{\max} , T_{\min} , and T_{avg} are the maximum, minimum, and average air temperature for a given day ($^{\circ}\text{C}$), respectively.

The source code of the Hargreaves method for potential evapotranspiration can be found in `etpot.f`. The following code snippet showed the core simulation code.

```
01      case (2)    !! HARGREAVES POTENTIAL EVAPOTRANSPIRATION METHOD
02
03      !! extraterrestrial radiation
04      !! 37.59 is coefficient in equation 2.2.6 !!extraterrestrial
05      !! 30.00 is coefficient in equation 2.2.7 !!max at surface
06      ramm = 0.
07      ramm = hru_rmx(j) * 37.59 / 30.
08
09      if (tmx(j) > tmn(j)) then
10          pet_day = harg_petco(hru_sub(j))*(ramm / xl)*(tmpav(j) + 17.8)*
11          &                                         (tmx(j) - tmn(j))**0.5
12          pet_day = Max(0., pet_day)
13      else
14          pet_day = 0.
15      endif
```

where `ramm` is the extraterrestrial radiation which is calculated by maximum possible radiation for the day (`hru_rmx`), `hru_rmx` is a function of day of year and latitude, `xl` is the latent heat of vaporization which is calculated by the mean air temperature (`tmpav`).

The source code of the calculation of `hru_rmx` can be found in `clgen.f`.

To sum up, the simulation of potential evapotranspiration using the Hargreaves method requires the coefficient related to radiation, latitude, maximum temperature, minimum temperature, and average temperature. The simulation code can be rewritten from the FORTRAN source code of SWAT (i.e., `etpot.f` and `clgen.f`).

4:1.2 Create new SEIMS module

As introduced in previous sections, the source code of SEIMS library is located in `SEIMS/seims/src/seims_main/modules`. For the convenience of management, SEIMS modules are divided into several categories. There is one `CMakeLists.txt` file to indicate which categories are involved in the SEIMS project, i.e., `SEIMS/seims/src/seims_main/modules/CMakeLists.txt`.

```
01 MESSAGE(STATUS "      Compiling SEIMS_subdir: modules...")
02 ADD_SUBDIRECTORY("./climate")
03 ADD_SUBDIRECTORY("./hydrology")
04 ADD_SUBDIRECTORY("./hydrology_longterm")
05 ADD_SUBDIRECTORY("./erosion")
06 ADD_SUBDIRECTORY("./nutrient")
07 ADD_SUBDIRECTORY("./ecology")
08 ADD_SUBDIRECTORY("./management")
09 ADD_SUBDIRECTORY("./test")
```

It is worth to note that a corresponding line with the `ADD_SUBDIRECTORY()` command should be appended if a new category is added.

The template of the SEIMS module is the `template` folder located in `SEIMS/seims/src/seims_main/modules/test`, which contains four files, i.e., `CMakeLists.txt`, `api.cpp`, `template.h`, and `template.cpp`.

1. `CMakeLists.txt`: file used to create the project for building the module executable library.
2. `api.cpp`: implementation file of `MetadataInformation` function.

3. **template.h**: header file that defines the module class which is inherited from the basic class of SEIMS module (i.e., `SimulationModule`). The header file should be renamed with a meaningful name, such as `PETHargreaves.h`.
4. **template.cpp**: implementation file that includes `SetData` and `GetData` functions `CheckInputData` and `InitialOutputs` functions, and `Execute` function. The implementation file should be renamed with the same filename of the header file, such as `PETHargreaves.cpp`.

Please follow these steps to create a new SEIMS module based on the module template.

1. Copy the folder of the SEIMS module template to an existing or newly created directory of SEIMS module library, e.g., `modules/hydrology_longterm`, and rename with the new module name followed the format of `<SubprocessNameAbbr>_<AlgorithmNameAbbr>`, e.g., `PET_H` that means the simulation of potential evapotranspiration using Hargreaves method.
2. Open the `CMakeLists.txt` file in the module folder of `PET_H` and specify the name of the executable library (the 2nd line) and the project folder for Visual Studio or Xcode (the 10th line, which is Optional). For example, the content of the `CMakeLists.txt` file in the module template is as follows,

```

01 MESSAGE(STATUS "      Compiling test_subdir: template...")
02 SET(MODNAME template)
03 PROJECT(SEIMS_MODULE_${MODNAME})
04 FILE(GLOB SRC_LIST *.cpp *.h)
05 ADD_LIBRARY(${MODNAME} SHARED ${SRC_LIST})
06 SET(LIBRARY_OUTPUT_PATH ${SEIMS_BINARY_OUTPUT_PATH})
07 TARGET_LINK_LIBRARIES(${MODNAME} common_algorithm module_setting bmps data)
08 INSTALL(TARGETS ${MODNAME} DESTINATION ${INSTALL_DIR})
09 IF (MSVC OR XCODE)
10   SET_PROPERTY(TARGET ${MODNAME} PROPERTY FOLDER "modules/test")
11 ENDIF ()

```

After modification for the `PET_H` module,

```

01 MESSAGE(STATUS "      Compiling hydrology_longterm_subdir: PET_H...")
02 SET(MODNAME PET_H)
03 PROJECT(SEIMS_MODULE_${MODNAME})
04 FILE(GLOB SRC_LIST *.cpp *.h)
05 ADD_LIBRARY(${MODNAME} SHARED ${SRC_LIST})
06 SET(LIBRARY_OUTPUT_PATH ${SEIMS_BINARY_OUTPUT_PATH})
07 TARGET_LINK_LIBRARIES(${MODNAME} module_setting common_algorithm)
08 INSTALL(TARGETS ${MODNAME} DESTINATION ${INSTALL_DIR})
09 IF (MSVC OR XCODE)
10   SET_PROPERTY(TARGET ${MODNAME} PROPERTY FOLDER "modules/hydrology_longterm")
11 ENDIF ()

```

Note that, the 7th line is for specifying libraries to use when linking the current SEIMS module. If users do not sure how to determine the exact libraries, just keep the default settings in the module template.

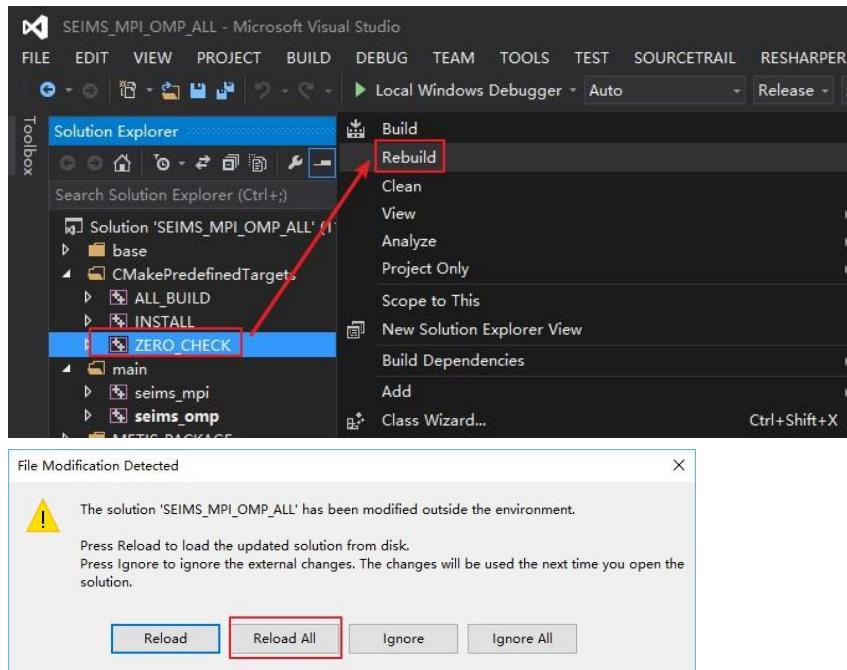
3. Append the `ADD_SUBDIRECTORY()` command of the relative directory path of the new module to the `CMakeLists.txt` file in its parent directory, such as `modules/hydrology_longterm/CMakeLists.txt`, such as,

```

01 MESSAGE STATUS "      Compiling SEIMS_subdir: hydrology_longterm..." )
02 ADD_SUBDIRECTORY(./DEP_LINSLEY)
03 ADD_SUBDIRECTORY(./GWA_RE)
04 ADD_SUBDIRECTORY(./IUH_DL)
05 ADD_SUBDIRECTORY(./MUSK_CH)
06 ADD_SUBDIRECTORY(./PER_PI)
07 ADD_SUBDIRECTORY(./PER_STR)
08 ADD_SUBDIRECTORY(./PET_PM)
09 ADD_SUBDIRECTORY(./PET_PT)
10 ADD_SUBDIRECTORY(./PET_H)

```

4. Rerun the CMake command as introduced in Section 2:1.4 to build the SEIMS project in which the newly created module which will be included.
5. Alternative to step 4, open the SEIMS solution in Visual Studio (i.e., SEIMS/build/SEIMS_MPI_OMP_ALL.sln), and right-click on the ZERO_CHECK project in CMakePredefinedTargets and select Rebuild. Then, click on Reload All to reload all projects of SEIMS including the newly created module.



4:1.3 Open SEIMS solution in Visual Studio and start to code

After (re)building the SEIMS solution which includes the newly created module, the following steps are recommended to start to code.

1. Open the header file (i.e., PETHargreaves.h in this demo) and rename the class name from ModuleTemplate to PETHargreaves. Also remember to change the include guard from SEIMS_MODULE_TEMPLATE_H to SEIMS_MODULE_PET_H_H or any other unique macro name in the whole SEIMS solution.
2. Open the api.cpp file and write the MetadataInformation function. To avoid inconsistent typing of the name, unit, and description of the same parameter/variable, the definition of all these strings are defined as macro strings in the text.h header file (SEIMS/seims/src/seims_main/base/text.h) in the current implementation of

SEIMS. Thus, developers should search the planned variable first in the `text.h` to make sure its definition is unique.

According to the review of theory literature and source code of SWAT, eight inputs should be defined and four of them are from the database while the others are outputs of other modules.

```
01 // Parameters from Database (non-time series)
02 mdi.AddParameter(VAR_PET_HCOEF, UNIT_NON_DIM, DESC_PET_HCOEF,
Source_ParameterDB, DT_Single);
03 mdi.AddParameter(VAR_CELL_LAT, UNIT_LONLAT_DEG, DESC_CELL_LAT,
Source_ParameterDB, DT_Raster1D);
04 // Inputs from the output of other modules
05 mdi.AddInput(VAR_TMEAN, UNIT_TEMP_DEG, DESC_TMEAN, Source_Module, DT_Raster1D);
06 mdi.AddInput(VAR_TMAX, UNIT_TEMP_DEG, DESC_TMAX, Source_Module, DT_Raster1D);
07 mdi.AddInput(VAR_TMIN, UNIT_TEMP_DEG, DESC_TMIN, Source_Module, DT_Raster1D);
08 mdi.AddInput(DataType_RelativeAirMoisture, UNIT_PERCENT, DESC_RM,
Source_Module, DT_Raster1D);
```

As to the output variables of the current module, except for the potential evapotranspiration on the current day (VAR_PET), there are also an auxiliary output, i.e., the day length (VAR_DAYLEN) when calculating the maximum possible radiation for the day.

```
01 mdi.AddOutput(VAR_DAYLEN, UNIT_HOUR, DESC_DAYLEN, DT_Raster1D);
02 mdi.AddOutput(VAR_PET, UNIT_WTRDLT_MMD, DESC_PET, DT_Raster1D);
```

3. Define the input parameters, input variables from other modules, and output variables of the current module in the header file.

```
01 // Parameters from Database
02 int m_nCells; ///< valid units number
03 float m_HCoef_pet; ///< coefficient related to radiation used in Hargreaves method
04 float* m_cellLat; ///< latitude of each valid units
05 // Inputs from the output of other modules
06 float* m_meanTemp; ///< mean air temperature for a given day (deg C)
07 float* m_maxTemp; ///< maximum air temperature for a given day (deg C)
08 float* m_minTemp; ///< minimum air temperature for a given day (deg C)
09 float* m_rhd; ///< relative humidity (%)
10 // Output variables
11 float* m_dayLen; ///< VAR_DAYLEN, day length (hr)
12 float* m_pet; ///< VAR_PET, potential evapotranspiration on the day
```

4. Complete the initialization (i.e., the constructor function that sets default values for all variables defined in header file) and finalization (i.e., the destructor function that releases the newly allocated memory of the current module such as array-based output variables).

```
01 PETHargreaves::PETHargreaves() :
02     m_nCells(-1), m_HCoef_pet(0.0023f), m_cellLat(nullptr),
03     m_meanTemp(nullptr), m_maxTemp(nullptr), m_minTemp(nullptr),
04     m_dayLen(nullptr), m_pet(nullptr) {
05 }
06
07 PETHargreaves::~PETHargreaves() {
08     if (m_dayLen != nullptr) Release1DArray(m_dayLen);
09     if (m_pet != nullptr) Release1DArray(m_pet);
10 }
```

5. Assign values to input parameters and input variables from other modules in **SetData** series functions such as **SetValue**, **Set1DData**, and **Set2DData**.

```
01 void PETHargreaves::SetValue(const char* key, const float value) {
02     string sk(key);
03     if (StringMatch(sk, VAR_PET_HCOEF)) m_Hcoef_pet = value;
04     else {
05         throw ModelException(MID_PET_H, "SetValue", "Parameter " + sk +
06                             " does not exist in current module.");
07     }
08 }

01 void PETHargreaves::Set1DData(const char* key, const int n, float* value) {
02     // Check the first dimension (m_nCells) of array-based data
03     CheckInputSize(key, n);
04     string sk(key);
05     if (StringMatch(sk, VAR_TMEAN)) m_meanTemp = value;
06     else if (StringMatch(sk, VAR_TMAX)) m_maxTemp = value;
07     else if (StringMatch(sk, VAR_TMIN)) m_minTemp = value;
08     else if (StringMatch(sk, DataType_RelativeAirMoisture)) m_rhd = value;
09     else if (StringMatch(sk, VAR_CELL_LAT)) m_cellLat = value;
10     else {
11         throw ModelException(MID_PET_H, "Set1DData", "Parameter " + sk +
12                             " does not exist in current module.");
13     }
14 }
```

6. Accomplish the validation check of data set by **SetData** functions in **CheckInputData** function and initializing necessary output variables in **InitialOutputs** function.

```
01 bool PETHargreaves::CheckInputData() {
02     CHECK_POSITIVE(MID_PET_H, m_nCells);
03     CHECK_POINTER(MID_PET_H, m_maxTemp);
04     CHECK_POINTER(MID_PET_H, m_meanTemp);
05     CHECK_POINTER(MID_PET_H, m_minTemp);
06     CHECK_POINTER(MID_PET_H, m_cellLat);
07     return true;
08 }

01 void PETHargreaves::InitialOutputs() {
02     CHECK_POSITIVE(MID_PET_H, m_nCells);
03     if (nullptr == m_pet) Initialize1DArray(m_nCells, m_pet, 0.f);
04     if (nullptr == m_dayLen) Initialize1DArray(m_nCells, m_dayLen, 0.f);
05 }
```

7. Implement or transplant the actual simulation code in **Execute** function.

```

01 int PETHargreaves::Execute() {
02     CheckInputData();
03     InitialOutputs();
04 #pragma omp parallel for
05     for (int i = 0; i < m_nCells; i++) {
06         // calculate the max solar radiation
07         float srMax; // maximum solar radiation of current day
08         MaxSolarRadiation(m_dayOfYear, m_cellLat[i], m_dayLen[i], srMax);
09         //calculate latent heat of vaporization(from swat)
10         float latentHeat = 2.501f - 0.002361f * m_meanTemp[i];
11         /// extraterrestrial radiation
12         /// equation 1:1.1.6 in SWAT Theory 2009, p33
13         float h0 = srMax * 37.59f / 30.0f;
14         /// calculate potential evapotranspiration
15         /// equation 2:2.2.24 in SWAT Theory 2009, p133
16         float petValue = m_HCoef_pet * h0
17             * pow(Abs(m_maxTemp[i] - m_minTemp[i]), 0.5f)
18             * (m_meanTemp[i] + 17.8f) / latentHeat;
19         m_pet[i] = m_petFactor * Max(0.0f, petValue);
20     }
21     return 0;
22 }

```

Note that the function of calculating maximum solar radiation of current day, as a universal function, should be separated from the specific module. So the MaxSolarRadiation function is defined in

SEIMS/seims/src/seims_main/base/common_algorithm/ClimateParams.h and implemented in

SEIMS/seims/src/seims_main/base/common_algorithm/ClimateParams.cpp.

```
01 #include "ClimateParams.h"
```

The #pragma omp parallel for is the compiler directive of OpenMP to indicate the following for-loop code region that to be executed concurrently.

8. Complete the `GetData` functions (e.g., `GetValue`, `Get1DData`, and `Get2DData`) for output variables of current module.

```

01 void PETHargreaves::Get1DData(const char* key, int* n, float** data) {
02     InitialOutputs();
03     string sk(key);
04     *n = m_nCells;
05     if (StringMatch(sk, VAR_PET)) *data = m_pet;
06     else if (StringMatch(sk, VAR_DAYLEN)) *data = m_dayLen;
07     else {
08         throw ModelException(MID_PET_H, "Get1DData", "Parameter " + sk + " does
09         not exist.");
10     }

```

9. Build the SEIMS module by right click the project name of the current module and select `Build` command.

4:1.4 Test the new SEIMS module

The newly developed `PET_H` module was tested based on the Youwuzhen watershed model customized in Section 2:4.3.3. The module configuration file is shown in Figure 4:1-1.

```
01 ### Driver factors, including climate and precipitation
02 0 | TimeSeries | | TSD_RD
03 0 | Interpolation_0 | Thiessen | ITP
04 ### Surface processes
05 0 | Soil temperature | Finn Plauborg | STP_FP
06 0 | PET | Hargreaves | PET_H
07 #0 | PET | PenmanMonteith | PET_PM
08 0 | Interception | Maximum Canopy Storage | PI_MCS
09 0 | Snow melt | Snowpeak Daily | SNO_SP
10 0 | Infiltration | Modified rational | SUR_MR
11 0 | Depression and Surface Runoff | Linsley | DEP_LINSLEY
12 0 | Hillslope erosion | MUSLE | SERO_MUSLE
13 0 | Plant Management Operation | SWAT | PLTMGT_SWAT
14 0 | Percolation | Storage routing | PER_STR
15 0 | Subsurface | Darcy and Kinematic | SSR_DA
16 0 | AET | Hargreaves and PriestleyTaylor | AET_PTH
17 #0 | SET | Linearly Method from WetSpa | SET_LM
18 0 | PG | Simplified EPIC | PG_EPIC
19 0 | ATMDEP | Atmosphere deposition | ATMDEP
20 0 | NUTR_TF | Nutrient Transformation of C, N, and P | NUTR_TF
21 0 | Water overland routing | IUH | IUH_OL
22 0 | Sediment overland routing | IUH | IUH_SED_OL
23 0 | Nutrient | Attached nutrient loss | NUTRSED
24 0 | Nutrient | Soluble nutrient loss | NUTRMV
25 0 | Pothole | SWAT cone shape | IMP_SWAT
26 0 | Soil water | Water balance | SOL_WB
27 ### Route Modules, including water, sediment, and nutrient
28 0 | Groundwater | Linear reservoir | GWA_RE
29 0 | Nutrient | groundwater nutrient transport | NUTRGW
30 0 | Water channel routing | MUSK | MUSK_CH
31 0 | Sediment channel routing | Simplified Bagnold equation | SEDR_SBAGNOLD
32 0 | Nutrient | Channel routing | NutrCH_QUAL2E
```

Figure 4:1-1 Test the Hargreaves method for the simulation of potential evapotranspiration based on the Youwuzhen watershed model customized in Section 2:4.3.3.

The Youwuzhen watershed model was executed with all model parameters remain the default values as well. Figure 4:1-2 showed the spatial distributions of average potential evapotranspiration simulated by the Priestley-Taylor method (Figure 4:1-2a) and Hargreaves method (Figure 4:1-2b).

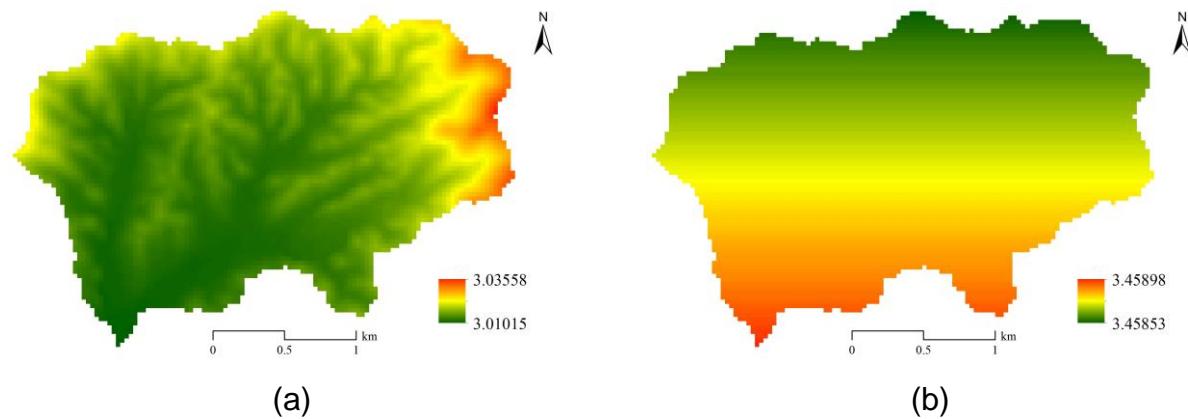


Figure 4:1-2 Spatial distributions of average potential evapotranspiration simulated by the Youwuzhen watershed model using the (a) Priestley-Taylor method and the (b) Hargreaves method