Spatially Explicit Integrated Modeling System (SEIMS)

User Manual

Version 1.0

Liang-Jun Zhu, Junzhi Liu, Cheng-Zhi Qin, A-Xing Zhu

01/06/2019

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

Table of Contents

List of Figures

List of Tables

Section 1. Introduction

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 BMPs scenarios.

SEIMS can be used to build a specific watershed model of a single watershed for long-term or storm event simulation. Multiple watershed process 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 or fields with hydrological connection are used as basic simulation units.
- SEIMS is a multiple watershed processes integrated modular system, to which 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 way.
- SEIMS is a configurable watershed modeling system, in which users can easily specify their objects, inputs 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 common used C++ compiler (e.g. MSVC 2010+, GCC4.6+, Intel C++ 12+, and Clang 8.0+) and run on multi-core computer and high-performance clusters.
- SEIMS, which uses several open source technologies (e.g., GDAL, MongoDB), 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 preprocessing scripts*
- *Set up a SEIMS-based model according to model objectives and run model*
- *Postprocessing, e.g., analyze, plot and graph outputs*
- *(Optional) Parameters 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., **2018**. A modular and parallelized watershed modeling framework. *Environmental Modelling & Software* (*under review*).

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., **2018**. Effects of different spatial configuration units for spatial optimization of watershed best management practices scenarios. *Water* (*under review*).

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 we need SEIMS?

1:3. Terminology

SEIMS-based watershed model – A SEIMS-based watershed model is a folder which consists of several SEIMS configuration files, e.g., `demo_youwuzhen30m_longterm_model`.

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 API specification and could be loaded by SEIMS main program. A SEIMS module is corresponding to one or several hydrological, soil erosion, plant growth, nutrient cycling, and BMPs management process, etc.

1:4. References

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 preprocess, postprocess, parameters sensitivity, calibration, and scenario 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. MSVC 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, Windows 10 64bit, MSVC 2013, 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 Section step by step to get started with SEIMS, including download and install, understanding the data preparation of the demo watershed, preprocessing and running the user-configured SEIMS-based watershed model, postprocessing, parameters sensitivity analysis, auto-calibration, and BMPs (Best Management Practices) 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.

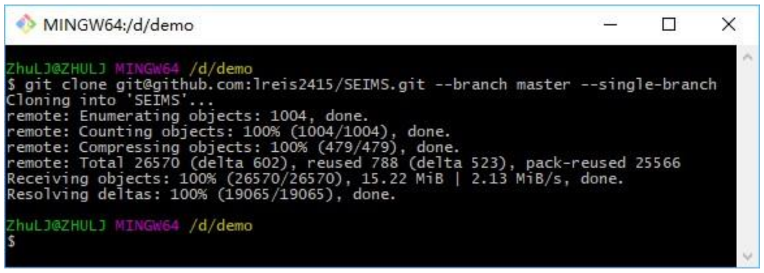
Users are encouraged to download and install (means compile from the source code) SEIMS manually. Generally, the `master` branch of SEIMS repository (https://github.com/lreis2415/SEIMS/tree/master) is the relative stable version, while the `dev` branch reflects the latest development changes.

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.*

     ```
     $ git config --global user.name "Your Name"
     $ git config --global user.email "email@example.com"
     e.g.,
     $ git config --global user.name "crazyzlj"
     $ git config --global user.email "crazyzlj@gmail.com"
     ```
  3. *Enter the following command to clone the single `master` branch (Figure 2:1-1).*

     ```
     $ git clone git@github.com:lreis2415/SEIMS.git --branch master --single-branch
     ```

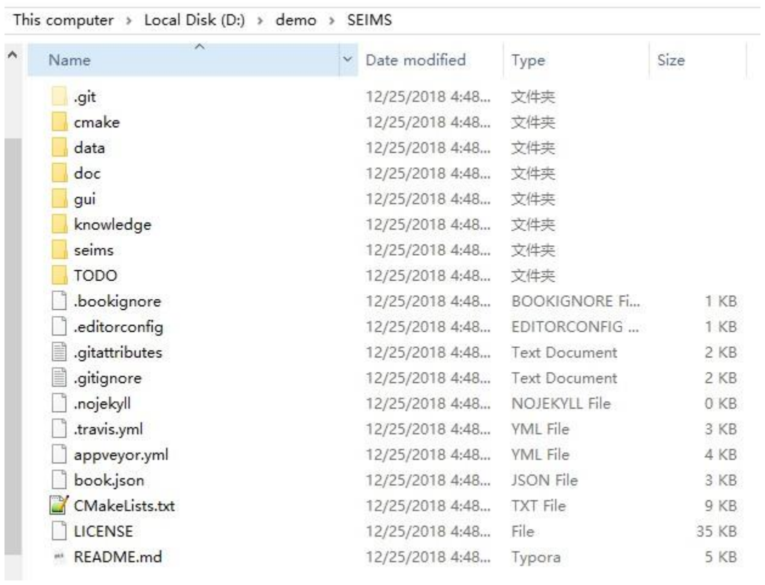


```
ZhuLJ@ZHULJ MINGW64 /d/demo
$ git clone git@github.com:lreis2415/SEIMS.git --branch master --single-branch
Cloning into 'SEIMS'...
remote: Enumerating objects: 1004, done.
remote: Counting objects: 100% (1004/1004), done.
remote: Compressing objects: 100% (479/479), done.
remote: Total 26570 (delta 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 (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 located in `D:/demo/SEIMS` such as Figure 2:1-2.


This computer › Local Disk (D:) › demo › SEIMS

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 Studio (MSVC)

SEIMS uses several features of C++11 such as `nullptr` and `auto` keywords. Therefore, the minimum support version is MSVC 2010. 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). 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 the C/C++ compiler.

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`, 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, e.g., `C:\Program Files (x86)\CMake\cmake.exe`.

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 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 (community) 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.

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\mongod.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:*

  ```
  use test
  db.foo.save({hello:0})
  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.

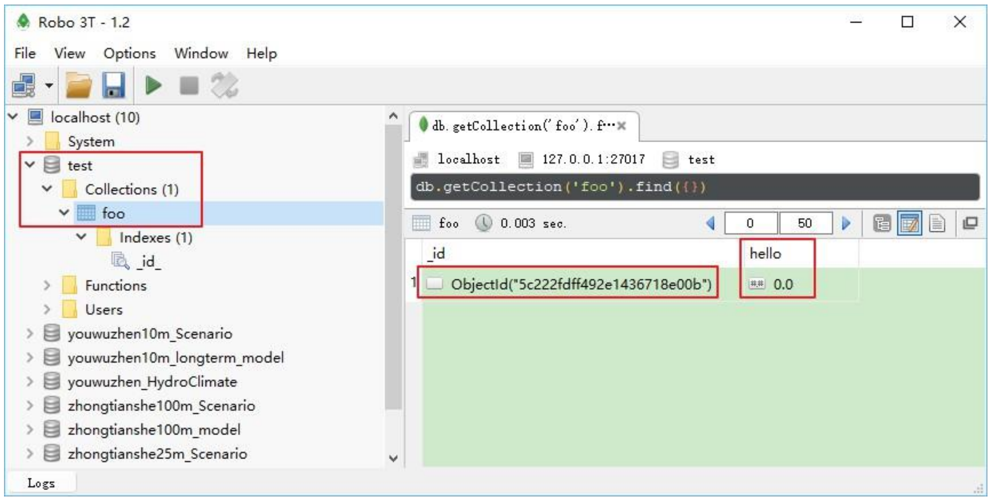
- *Open a CMD as administrator and enter the following commands:*
  ```
  cd D:\MongoDB\bin\mongo.exe --dbpath=D:\MongoDB\db --
  logpath=D:\MongoDB\logs\mongodb.log --install --serviceName "MongoDB"
  ```
- *If something like "*`Tue Oct 09 12:05:15 Service can be started from the command line with 'net start MongoDB'`*" showed in the console window, the service has been successfully registered.*
- *Now, a simple command* `net start MongoDB` *is enough to start MongoDB server.*
- *Next, we need to 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 occurs, 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).



```
db.getCollection('foo').find({})
```

_id	hello
ObjectId("5c222fdff492e1436718e00b")	0.0

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 location. 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\
```

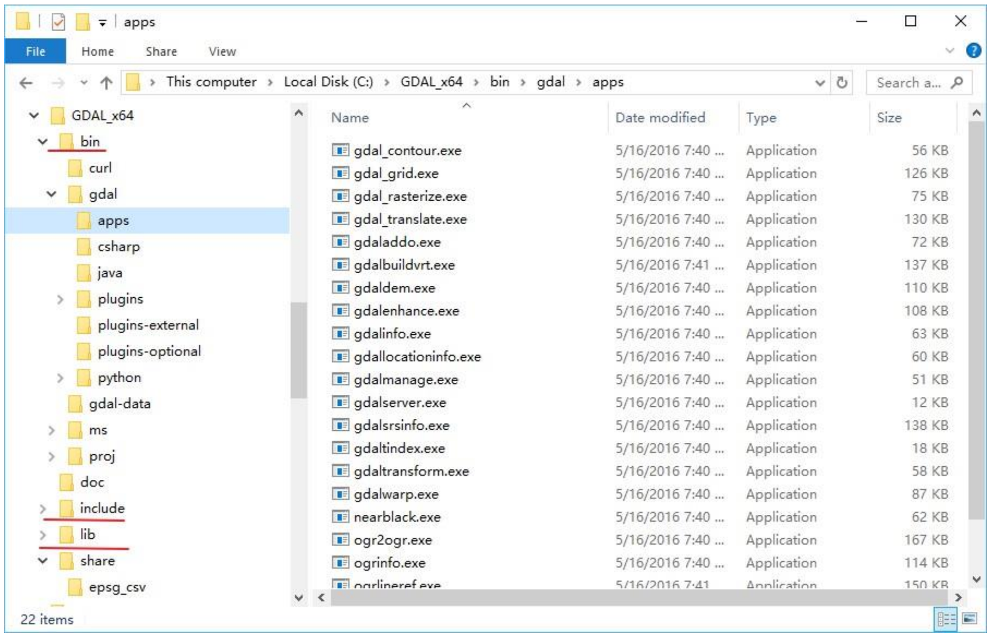
2:1.2.6 GDAL library for C/C++ and Python

Download the pre-compiled library is the most convenient and successful way to install GDAL library for 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.*


apps
File
Home
Share
View
This computer › Local Disk (C:) › GDAL_x64 › bin › gdal › apps
Search a...

```
GDAL_x64
  bin
    curl
    gdal
      apps
      csharp
      java
      plugins
      plugins-external
      plugins-optional
      python
    gdal-data
    ms
    proj
  doc
  include
  lib
  share
    epsg_csv
```

Name	Date modified	Type	Size
gdal_contour.exe	5/16/2016 7:40 ...	Application	56 KB
gdal_grid.exe	5/16/2016 7:40 ...	Application	126 KB
gdal_rasterize.exe	5/16/2016 7:40 ...	Application	75 KB
gdal_translate.exe	5/16/2016 7:40 ...	Application	130 KB
gdaladdo.exe	5/16/2016 7:40 ...	Application	72 KB
gdalbuildvrt.exe	5/16/2016 7:41 ...	Application	137 KB
gdaldem.exe	5/16/2016 7:40 ...	Application	110 KB
gdalenhance.exe	5/16/2016 7:40 ...	Application	108 KB
gdalinfo.exe	5/16/2016 7:40 ...	Application	63 KB
gdallocationinfo.exe	5/16/2016 7:40 ...	Application	60 KB
gdalmanage.exe	5/16/2016 7:40 ...	Application	51 KB
gdalserver.exe	5/16/2016 7:40 ...	Application	12 KB
gdalsrsinfo.exe	5/16/2016 7:40 ...	Application	138 KB
gdaltindex.exe	5/16/2016 7:40 ...	Application	18 KB
gdaltransform.exe	5/16/2016 7:40 ...	Application	58 KB
gdalwarp.exe	5/16/2016 7:40 ...	Application	87 KB
nearblack.exe	5/16/2016 7:40 ...	Application	62 KB
ogr2ogr.exe	5/16/2016 7:40 ...	Application	167 KB
ogrinfo.exe	5/16/2016 7:40 ...	Application	114 KB
ogrlineref.exe	5/16/2016 7:41	Application	150 KB

22 items

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\a
  pps;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" maintained by Christoph Gohlke (https://www.lfd.uci.edu/~gohlke/pythonlibs/) is highly recommended to download the compiled binaries of packages (i.e., `*.whl` files), not only the GDAL but also almost the commonly used Python packages.

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

- *Select 32- or 64-bit binaries of packages according to the version of your 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.*
- *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.*

  ```
  d:\demo>d:\demo\python27\scripts\pip install numpy-1.15.4+mkl-cp27-
  cp27m-win_amd64.whl
  Processing .\numpy-1.15.4+mkl-cp27-cp27m-win_amd64.whl
  Installing collected packages: numpy
  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.*

  ```
  d:\demo>d:\demo\python27\scripts\pip install GDAL-2.2.4-cp27-
  cp27m-win_amd64.whl
  Processing .\gdal-2.2.4-cp27-cp27m-win_amd64.whl
  Installing collected packages: GDAL
  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.*

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

As we can see, the version of GDAL library for Python in my computer is 2.2.4. This 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. Since mongo-c-driver is still under active and continuous development, the build method and API (Application Program Interface) functions may different with versions. So, for the most compatibility, the version of 1.6.1 is recommended, because 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`*.*

```
>cd D:\demo\mongo-c-driver-1.6.1
# 1. Configure, build, and install libbson first.
# Note that, in the latest version, libbson is no need to be built separately.
#  Please visit http://mongoc.org/libmongoc/current/installing.html for more information.
>cd src\libbson
>cmake -DCMAKE_INSTALL_PREFIX=C:\mongo-c-driver -G "Visual Studio 12 2013 Win64"
>msbuild.exe ALL_BUILD.vcxproj /p:Configuration=RelWithDebInfo
>msbuild.exe INSTALL.vcxproj /p:Configuration=RelWithDebInfo
# 2. Go back to the root folder, configure, build, and install mongoc
>cd ..\..
>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
# Use Windows Native TLS, rather then OpenSSL in case of strange link errors
# Use Windows Native SSPI, rather then Cyrus SASL
>msbuild.exe ALL_BUILD.vcxproj /p:Configuration=RelWithDebInfo
>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 **Py**thon for **GeoC**omputation, 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`.
  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 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*

  ```
  python 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
  ```

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

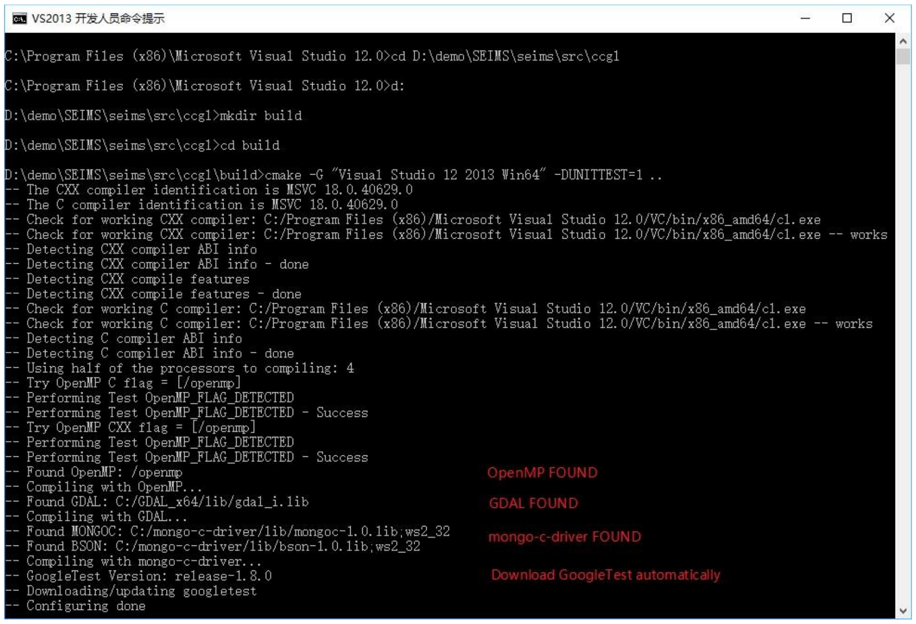
2:1.3 Test of the C/C++ building environment

Now, we have set up the C/C++ building environment for SEIMS such as MSVC 2013, 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 **C**ommon **C**ross-platform **G**eographic-computing **L**ibrary (`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.*

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



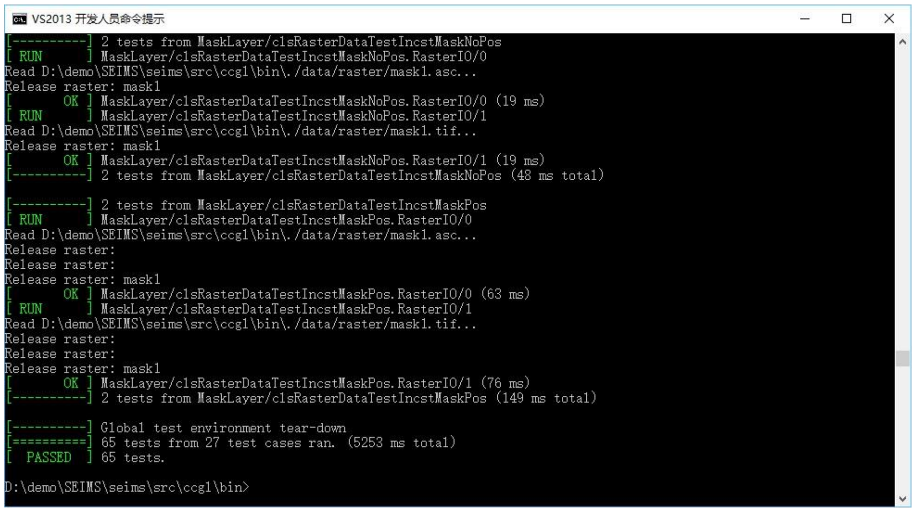
```
C:\Program Files (x86)\Microsoft Visual Studio 12.0>cd D:\demo\SEIMS\seims\src\ccgl

C:\Program Files (x86)\Microsoft Visual Studio 12.0>d:

D:\demo\SEIMS\seims\src\ccgl>mkdir build

D:\demo\SEIMS\seims\src\ccgl>cd build

D:\demo\SEIMS\seims\src\ccgl\build>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 = [/openmp]
-- Performing Test OpenMP_FLAG_DETECTED
-- Performing Test OpenMP_FLAG_DETECTED - Success
-- Try OpenMP CXX flag = [/openmp]
-- Performing Test OpenMP_FLAG_DETECTED
-- Performing Test OpenMP_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.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
```

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

- *Then, navigate to the binary directory of* `CCGL` *and run its unit test.*

```
>cd ../bin
>UnitTests_CCGL.exe
```

- *The final results should be like Figure 2:1-6.*



```
[----------] 2 tests from MaskLayer/clsRasterDataTestIncstMaskNoPos
[ RUN      ] MaskLayer/clsRasterDataTestIncstMaskNoPos.RasterIO/0
Read D:\demo\SEIMS\seims\src\ccgl\bin\./data/raster/mask1.asc...
Release raster: mask1
[       OK ] MaskLayer/clsRasterDataTestIncstMaskNoPos.RasterIO/0 (19 ms)
[ RUN      ] MaskLayer/clsRasterDataTestIncstMaskNoPos.RasterIO/1
Read D:\demo\SEIMS\seims\src\ccgl\bin\./data/raster/mask1.tif...
Release raster: mask1
[       OK ] MaskLayer/clsRasterDataTestIncstMaskNoPos.RasterIO/1 (19 ms)
[----------] 2 tests from MaskLayer/clsRasterDataTestIncstMaskNoPos (48 ms total)

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

2:1.4 Installation of SEIMS

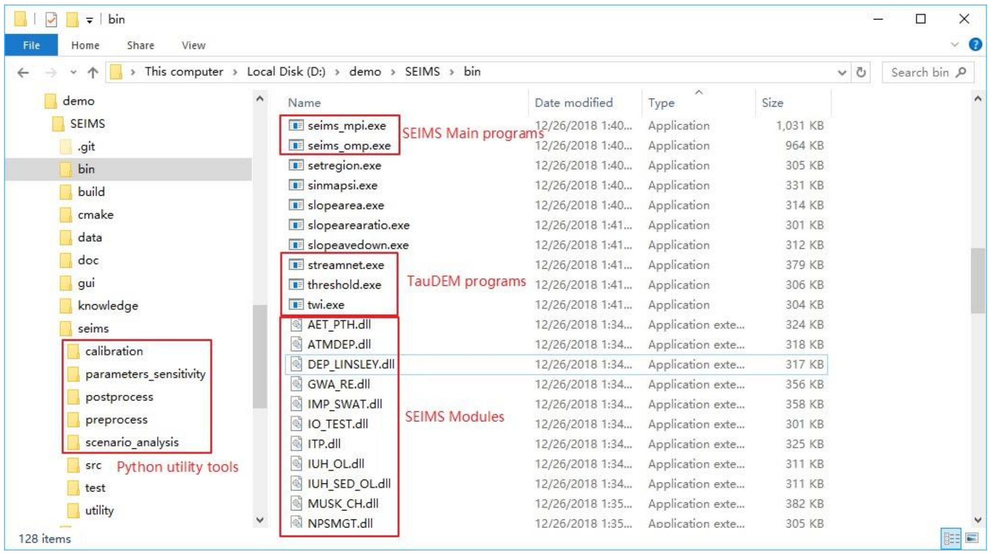
As stated earlier, SEIMS is mainly written by C++ and Python. Python is an interpreted language which means the source code can be executed directly under the Python environment without any manual compilation. So, the installation of SEIMS is the compilation and installation of C++ applications.

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`*.*

```
>cd D:\demo\SEIMS
>d:
>mkdir build
>cd build
>cmake -G "Visual Studio 12 2013 Win64" ..
>msbuild.exe ALL_BUILD.vcxproj /p:Configuration=Release
>msbuild.exe INSTALL.vcxproj /p:Configuration=Release
```

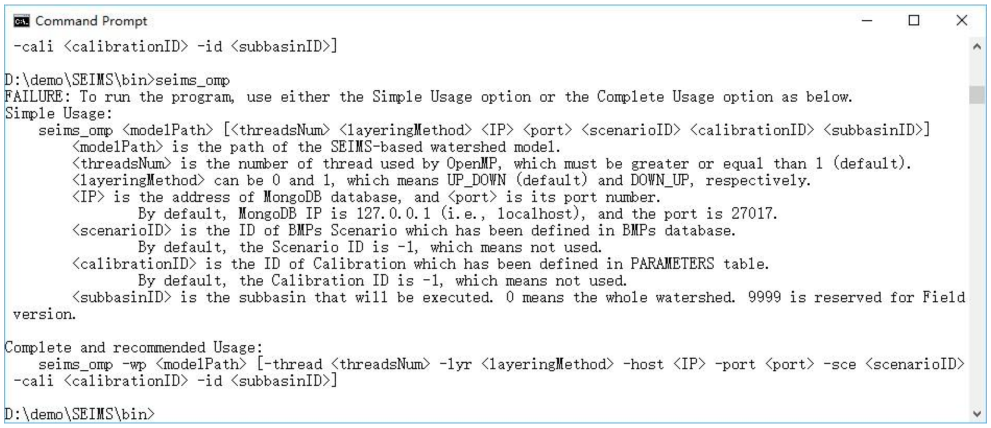
- *After the compilation and installation, all executables and libraries are installed at the default location, i.e.,* `D:\demo\SEIMS\bin`*.*
- *The complete SEIMS includes C++ applications located in* `D:\demo\SEIMS\bin` *and Python utility tools located in* `D:\demo\SEIMS\seims` *such as* `preprocess`*,* `postprocess`*,* `parameters_sensitivity`*,* `calibration`*, and* `scenario_analysis` *(Figure 2:1-7).*



Name	Date modified	Type	Size
seims_mpi.exe	12/26/2018 1:40...	Application	1,031 KB
seims_omp.exe	12/26/2018 1:40...	Application	964 KB
setregion.exe	12/26/2018 1:40...	Application	305 KB
sinmapsi.exe	12/26/2018 1:40...	Application	331 KB
slopearea.exe	12/26/2018 1:40...	Application	314 KB
slopearearatio.exe	12/26/2018 1:41...	Application	301 KB
slopeavedown.exe	12/26/2018 1:41...	Application	312 KB
streamnet.exe	12/26/2018 1:41...	Application	379 KB
threshold.exe	12/26/2018 1:41...	Application	306 KB
twi.exe	12/26/2018 1:41...	Application	304 KB
AET_PTH.dll	12/26/2018 1:34...	Application exte...	324 KB
ATMDEP.dll	12/26/2018 1:34...	Application exte...	318 KB
DEP_LINSLEY.dll	12/26/2018 1:34...	Application exte...	317 KB
GWA_RE.dll	12/26/2018 1:34...	Application exte...	356 KB
IMP_SWAT.dll	12/26/2018 1:34...	Application exte...	358 KB
IO_TEST.dll	12/26/2018 1:34...	Application exte...	301 KB
ITP.dll	12/26/2018 1:34...	Application exte...	325 KB
IUH_OL.dll	12/26/2018 1:34...	Application exte...	311 KB
IUH_SED_OL.dll	12/26/2018 1:34...	Application exte...	311 KB
MUSK_CH.dll	12/26/2018 1:35...	Application exte...	382 KB
NPSMGT.dll	12/26/2018 1:35...	Application exte...	305 KB

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".*



```
 -cali <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 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:
    seims_omp -wp <modelPath> [-thread <threadsNum> -lyr <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 types of file formats, i.e., 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 allowed by SEIMS. Even though, the GeoTIFF and ESRI Shapefile are highly recommended for raster and vector, 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 '`#`' symbol 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

```
# This line is comment.
SUBSCENARIO,NAME,LANDUSE,PARAMETERS
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:

```
demo_dict = {
    'SUBSCENARIO': 1,
    'NAME':'Closing measures'
    'LANDUSE': [7, 16],
    'PARAMETERS': [
        ['Interc_max', 'Maximum Interception Capacity', 'AC', 1],
        ['Conductivity', 'Soil hydraulic conductivity', 'RC', 3.5]
    ]
}
```

.

2:2.2 Spatial data

The Youwuzhen watershed (~5.39 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.

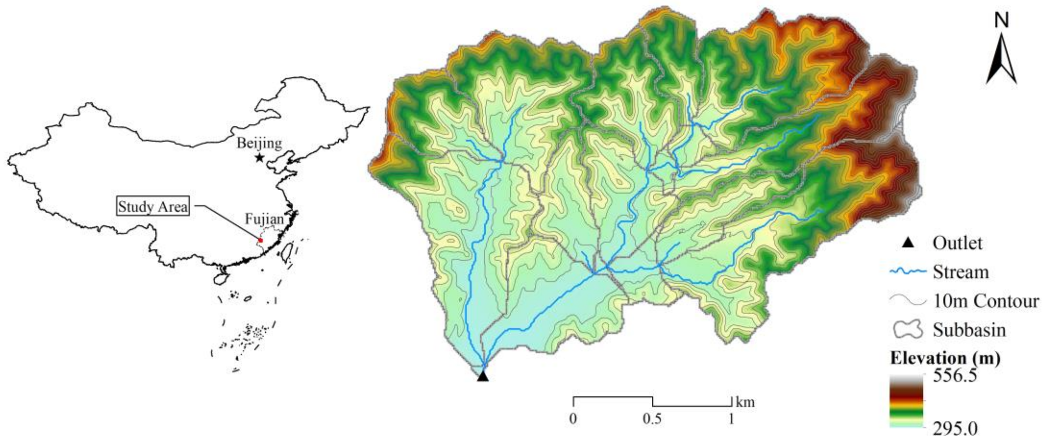


Figure 2:2-1 Location of the demo watershed named Youwuzhen

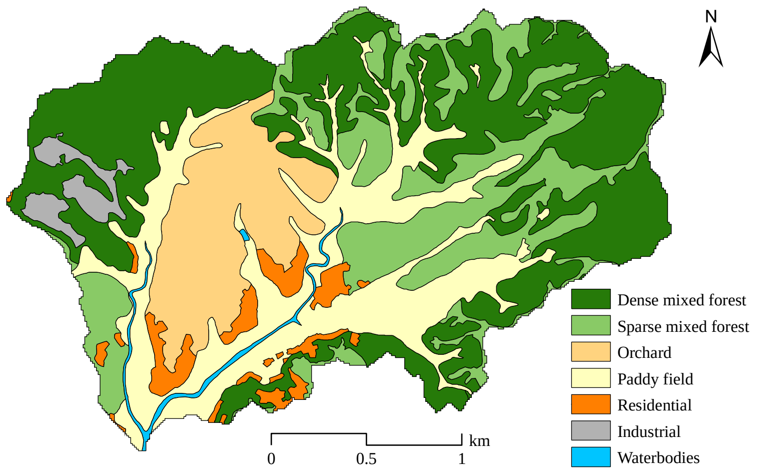


Figure 2:2-2 Landuse map of the Youwuzhen watershed

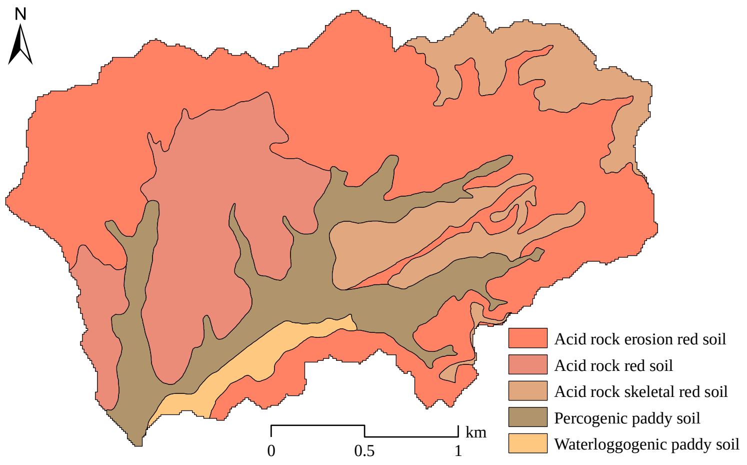

N
0
0.5
1
km
Acid rock erosion red soil
Acid rock red soil
Acid rock skeletal red soil
Percogenic paddy soil
Waterloggogenic paddy soil

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 `D:\demo\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, `D:\demo\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 `YYYY-MM-DD HH:MM:SS`
`StationID`s	Float	Precipitation value for each Station IDs. Multiple

Field name	Datatype	Description
		stations can be `StationID1`,`StationID2`, … `StationIDN`

Thus, the records of precipitation data is something like (`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 ℃.

The format of meteorological station is the same as that of precipitation station (see Section 2:2.3.1), e.g., `D:\demo\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/m2/day) or sunshine duration hour (unit: hr)

Field name	Datatype	Description
`PET`	Float	Potential evapotranspiration (mm)

Thus, the records of meteorological data is something like (`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., `D:\demo\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)

Field name	Datatype	Description
`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,
`D:\demo\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 `D:\demo\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,INT
ERVAL_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 string	Date time with the format of `YYYY-MM-DD HH:MM:SS`
`Type`	String	Monitoring variable
`VALUE`	Float	Monitoring value of current variable at the current date

Field name	Datatype	Description
		time

Thus, the records of observed data is something like (`D:\demo\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 `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
`SEQN` or `SOILCODE`	Integer	-	Unique identifier of soil map
`NAME`	String	-	Soil name
`SOILLAYERS`	Integer	-	Number of soil layers
`SOL_Z`	Float array	mm	Depth from soil surface to bottom of layer
`SOL_OM`	Float array	%	Organic matter content
`SOL_CLAY`	Float array	%	Clay content, d < 0.002 mm
`SOL_SILT`	Float array	%	Silt content, 0.002 mm < d < 0.05 mm
`SOL_SAND`	Float array	%	Sand content, 0.05 mm < d < 2 mm

Field name	Datatype	Unit	Description
`SOL_ROCK`	Float array	%	Rock fragment content, d > 2 mm
`SOL_BD`	Float array	Mg/m3	Moist bulk density, value ranges 1.1 ~ 1.9
`SOL_AWC`	Float array	mm	Available water capacity
`SOL_ZMX`	Float	mm	(Optional) Maximum rooting depth of soil profile
`ANION_EXCL`	Float	-	(optional) Fraction of porosity (void space) from which anions are excluded, default is 0.5
`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_NO3`	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 (`D:\demo\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,SOL_ROCK,SOL_WP,SOL_FC,SOL_POROSITY,SOL_K,SOL_AWC,SOL_NO3,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` or `ICNUM`	Integer	-	ICNUM, Land cover ID number (required when `IGRO` 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 (`D:\demo\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., `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.

Field name	Datatype	Description
`COLLECTION`	String	The name of the plain text file that defines the BMP parameters (e.g., `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 (`D:\demo\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 `scenario\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 `scenario\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 pant 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 Documentation[1] 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`).

[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.

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 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_SUR FACE	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_NA PP	AUTO_NY R	AUTO_EF F	AFRT_SUR FACE	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\slpposunits`) 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,EFFECTIVENESS,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:

```
demo_dict = {'NAME': 'fengjin',
             'DESC': 'CM (closing measures)',
             'REFERENCE': 'Qin et al (2018)',
             'LANDUSE': [8, 6],
             'SLPPOS': [1, 4],
             'PARAMETERS': [
                 {'NAME': 'OM',
                  'DESC': 'ORGANIC MATTER',
                  'CHANGE': 'RC',
                  'IMPACT': 1.22
                  },
                 {'NAME': 'DENSITY',
                  'DESC': 'BULK DENSITY',
                  'CHANGE': 'RC',
                  'IMPACT': 0.98
                  }
             ],
             'EFFECTIVENESS': 3,
             'CAPEX': 15.5,
             'OPEX': 1.5,
             'INCOME': 2,
            }
```
.

2:3. Data preprocessing for watershed modeling

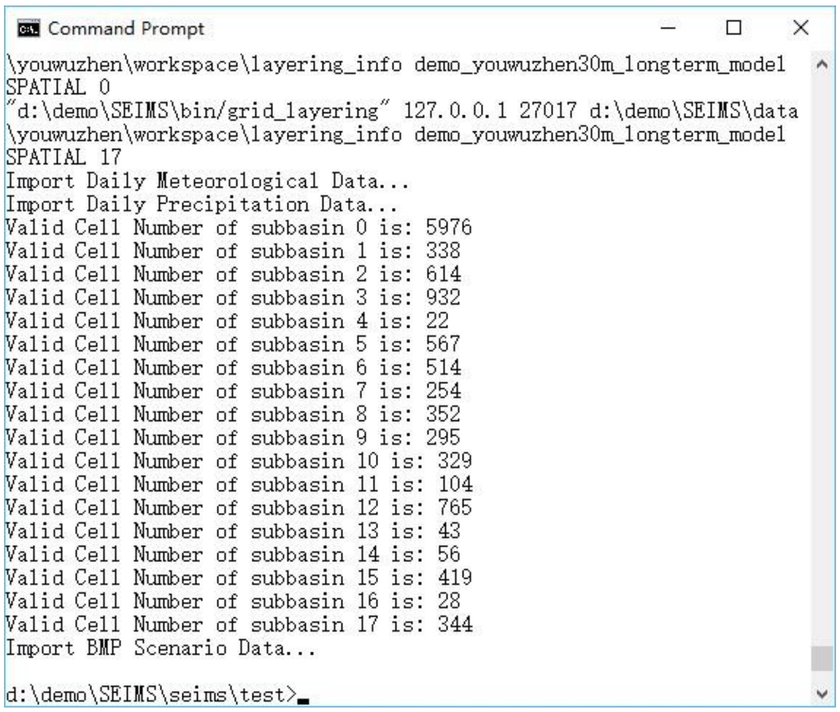
Data preprocessing for watershed modeling is a workflow to extract spatial parameters from various spatial data (e.g., DEM, landuse, and soil map), organize plaintext data (e.g., precipitation and meteorological data, site-monitoring data, and BMP scenarios data), and import these data into the MongoDB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

2:3.1 Simple usage

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

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

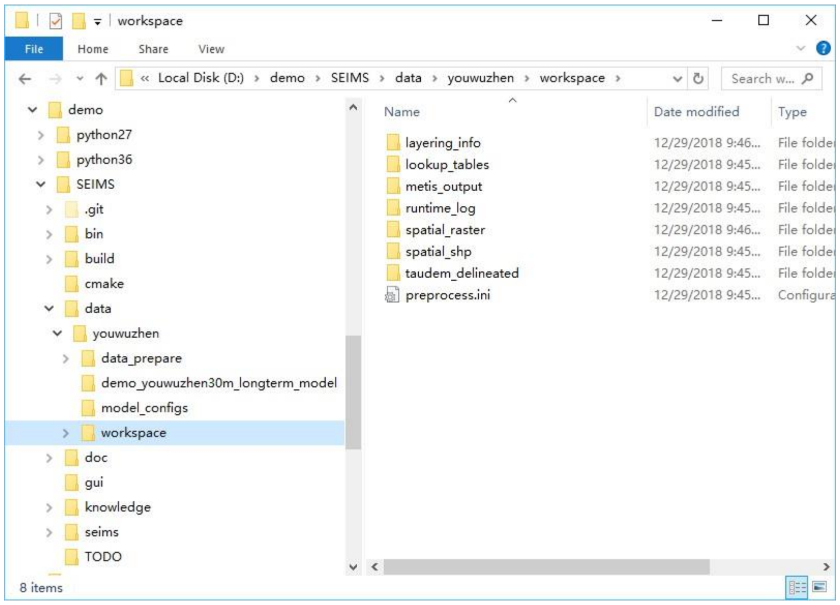
The data preprocessing runs successfully if no errors occurred and 'Import BMP Scenario Data…' showed at last, such as Figure 2:3-1.



```
\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

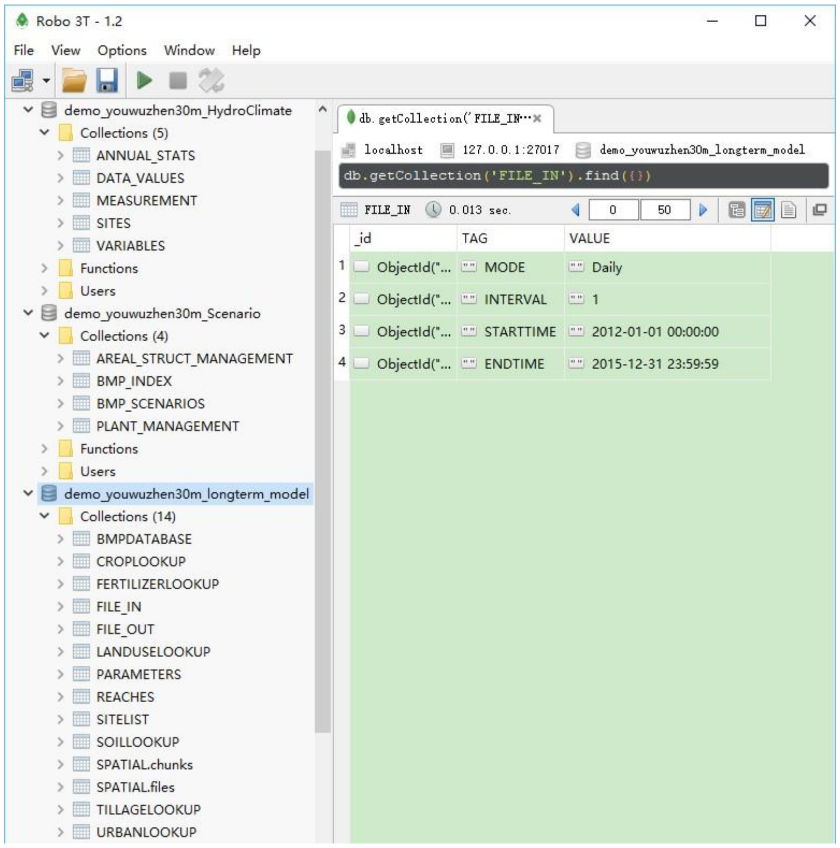
After running the simple usage, the input configuration file for data preprocessing (`preprocess.ini`) is generated in the directory of intermediate data (i.e., `data\youwuzhen\workspace`, Figure 2:3-2) and the data for watershed modeling has been imported into MongoDB (Figure 2:3-3). The details of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


workspace
File
Home
Share
View
Local Disk (D:) › demo › SEIMS › data › youwuzhen › workspace ›
Search w...
8 items

```
demo
  python27
  python36
  SEIMS
    .git
    bin
    build
    cmake
    data
      youwuzhen
        data_prepare
        demo_youwuzhen30m_longterm_model
        model_configs
        workspace
    doc
    gui
    knowledge
    seims
    TODO
```

Name	Date modified	Type
layering_info	12/29/2018 9:46...	File folde
lookup_tables	12/29/2018 9:45...	File folde
metis_output	12/29/2018 9:45...	File folde
runtime_log	12/29/2018 9:45...	File folde
spatial_raster	12/29/2018 9:46...	File folde
spatial_shp	12/29/2018 9:45...	File folde
taudem_delineated	12/29/2018 9:45...	File folde
preprocess.ini	12/29/2018 9:45...	Configura

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



```
demo_youwuzhen30m_HydroClimate
  Collections (5)
    ANNUAL_STATS
    DATA_VALUES
    MEASUREMENT
    SITES
    VARIABLES
  Functions
  Users
demo_youwuzhen30m_Scenario
  Collections (4)
    AREAL_STRUCT_MANAGEMENT
    BMP_INDEX
    BMP_SCENARIOS
    PLANT_MANAGEMENT
  Functions
  Users
demo_youwuzhen30m_longterm_model
  Collections (14)
    BMPDATABASE
    CROPLOOKUP
    FERTILIZERLOOKUP
    FILE_IN
    FILE_OUT
    LANDUSELOOKUP
    PARAMETERS
    REACHES
    SITELIST
    SOILLOOKUP
    SPATIAL.chunks
    SPATIAL.files
    TILLAGELOOKUP
    URBANLOOKUP
```

```
db.getCollection('FILE_IN').find({})
```

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

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

2:3.2 Configuration file for data preprocessing

Actually, the simple usage of the data preprocessing includes two steps, i.e., generate the configuration file according to the local paths of SEIMS and execute data preprocessing by the advanced usage which will be elaborated in the next Section.

SEIMS takes the INI file as the format of configuration files. `INI` files are simple text files with a basic structure composed of sections, properties and values. Semicolons (‘`;`’) or number sign (‘`#`’) at the beginning of the line indicate a comment which will be ignored. Figure 2:3-4 shows the configuration content of the Youwuzhen watershed.

```
[PATH]
PREPROC_SCRIPT_DIR = d:\demo\SEIMS\seims\preprocess
CPP_PROGRAM_DIR = d:\demo\SEIMS\bin
MPIEXEC_DIR = None
BASE_DATA_DIR = d:\demo\SEIMS\data\youwuzhen
CLIMATE_DATA_DIR = d:\demo\SEIMS\data\youwuzhen\data_prepare\climate
SPATIAL_DATA_DIR = d:\demo\SEIMS\data\youwuzhen\data_prepare\spatial
MEASUREMENT_DATA_DIR = d:\demo\SEIMS\data\youwuzhen\data_prepare\observed
BMP_DATA_DIR = d:\demo\SEIMS\data\youwuzhen\data_prepare\scenario
MODEL_DIR = d:\demo\SEIMS\data\youwuzhen\demo_youwuzhen30m_longterm_model
TXT_DB_DIR = d:\demo\SEIMS\data\youwuzhen\data_prepare\lookup
WORKING_DIR = d:\demo\SEIMS\data\youwuzhen\workspace
[MONGODB]
HOSTNAME = 127.0.0.1
PORT = 27017
ClimateDBName = demo_youwuzhen30m_HydroClimate
BMPScenarioDBName = demo_youwuzhen30m_Scenario
SpatialDBName = demo_youwuzhen30m_longterm_model
[CLIMATE]
HydroClimateVarFile = Variables.csv
MeteoSiteFile = Sites_M.csv
PrecSiteFile = Sites_P.csv
MeteoDataFile = meteo_daily.csv
PrecDataFile = pcp_daily.csv
thiessenIdField = ID
[SPATIAL]
dem = ywzdem30m.tif
outlet_file = outlet_beijing1954.shp
PrecSitesThiessen = thiessen_pcp.shp
MeteoSitesThiessen = thiessen_meteo.shp
landuseFile = ywzlanduse30m.tif
landcoverInitFile = landcover_initial_parameters.csv
soilSEQNFile = ywzsoil30m.tif
soilSEQNText = soil_properties_lookup.csv
[OPTIONAL_PARAMETERS]
D8AccThreshold = 35
np = 4
D8DownMethod = Surface
dorm_hr = -1.
T_base = 0.
imperviousPercInUrbanCell = 0.3
defaultLanduse = 33
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 key-value pairs are included within each section. The names of sections and keys should not be changed.

- *`PATH`: Full paths of executables, data, and intermediate workspace.*
  1. `PREPROC_SCRIPT_DIR`*: The path of Python scripts for data preprocessing.*
  2. `CPP_PROGRAM_DIR`*: The install directory of SEIMS C/C++ applications.*
  3. `MPIEXEC_DIR`*: The path of the executable of MPI. If it has been added in environment paths, the value can be* `None`*.*
  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, 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 configuration files for the watershed modeling. For data preprocessing, the* `file.in` *and* `file.out` *are required, the calibrated parameters* `param.cali` *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. `ClimateDBName`*: Name of the hydro-climate database created in MongoDB server (Figure 2:3-3).*
  4. `BMPScenarioDBName`*: Name of the BMP scenarios database created in MongoDB server (Figure 2:3-3).*
  5. `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`*.*
- *`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).*
- *`OPTIONAL_PARAMETERS`: Optional parameters.*
  1. `D8AccThreshold`: *Flow accumulation threshold for streamflow 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`: *Method of calculating 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, ‘`mpi_`’ for the task partitioning of parallel computing at subbasin level, 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:

```
>cd D:\demo\SEIMS\seims\preprocess
>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.,

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

It is worth to note that 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 explore the proper threshold manually according to the flow accumulated raster (i.e., `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 scripts are to support the currently available SEIMS modules of watershed processes. That means, if the input parameters of a newly developed SEIMS module are not included in current preprocessing, some scripts should added for their preparation. For more information, please referred to the “Section 4:1 Write 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 D8 flow direction (*`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.*
- `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` *and* `DOWNUP` *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* `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 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 is 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).

- `demo_youwuzhen30m_longterm_model`*: Main spatial database of the study area*
  1. *Model configuration collections (i.e., tables), 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 partitioned 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., `D:\demo\SEIMS\data\youwuzhen\data_prepare\scenario`), such as `PLANT_MANAGEMENT` and `AREAL_STRUCT_MANAGEMENT` in this demo study.*

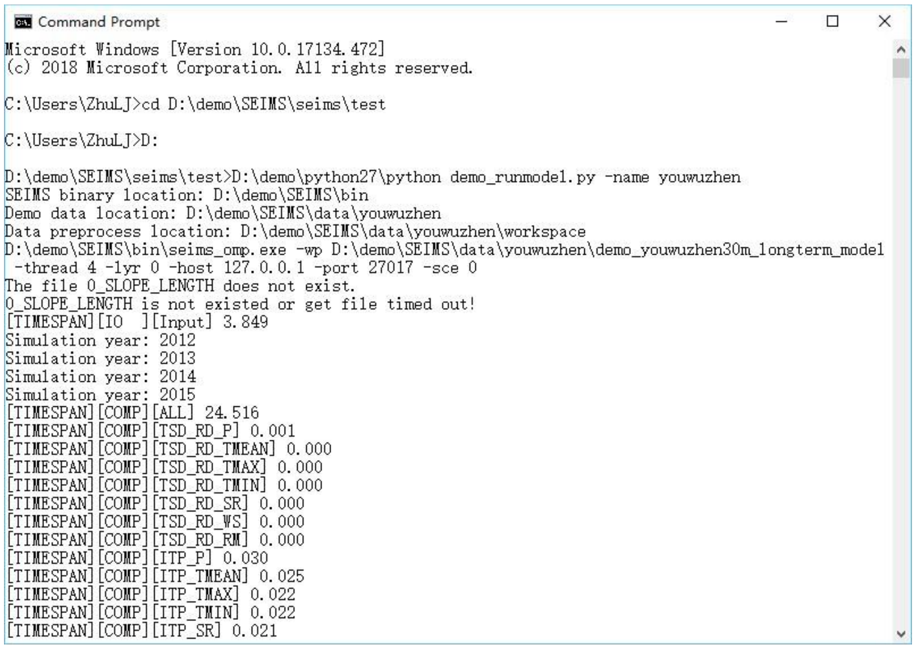
2:4. Running a SEIMS-based watershed model

2:4.1 Simple usage

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

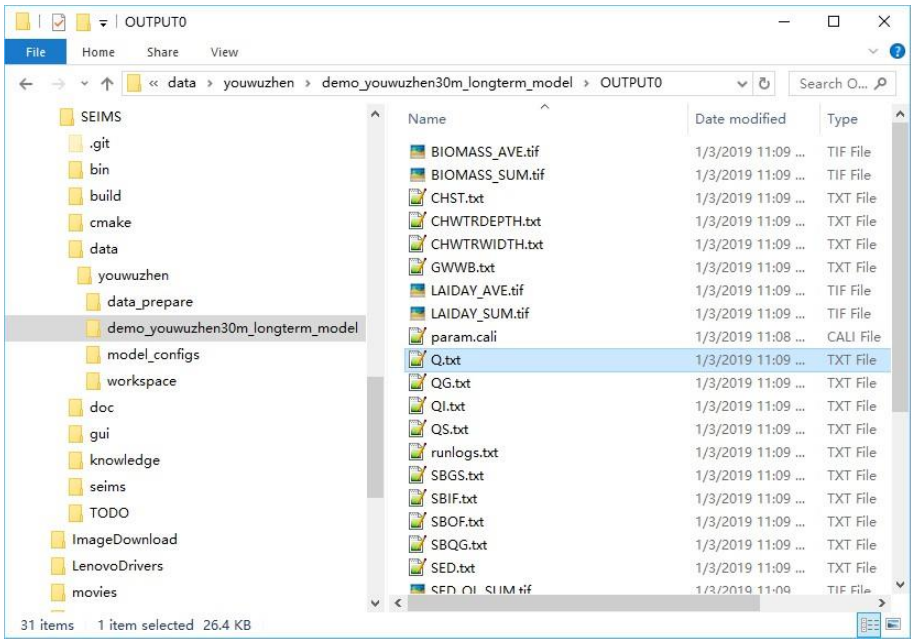
```
>cd D:\demo\SEIMS\seims\test
>D:
>python demo_runmodel.py -name youwuzhen
```

The simulation will be finished in a few seconds (Figure 2:4-1). The predefined output information can be found in the model folder (i.e., `MODEL_DIR` defined in the configuration file of data preprocessing, Section 2:3.2), such as Figure 2:4-2.



```
Command Prompt
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_longterm_model
 -thread 4 -lyr 0 -host 127.0.0.1 -port 27017 -sce 0
The file 0_SLOPE_LENGTH does not exist.
0_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


OUTPUT0
File
Home
Share
View
« data › youwuzhen › demo_youwuzhen30m_longterm_model › OUTPUT0
Search O...

```
SEIMS
  .git
  bin
  build
  cmake
  data
    youwuzhen
      data_prepare
      demo_youwuzhen30m_longterm_model
      model_configs
      workspace
  doc
  gui
  knowledge
  seims
  TODO
ImageDownload
LenovoDrivers
movies
```

Name	Date modified	Type
BIOMASS_AVE.tif	1/3/2019 11:09 ...	TIF File
BIOMASS_SUM.tif	1/3/2019 11:09 ...	TIF File
CHST.txt	1/3/2019 11:09 ...	TXT File
CHWTRDEPTH.txt	1/3/2019 11:09 ...	TXT File
CHWTRWIDTH.txt	1/3/2019 11:09 ...	TXT File
GWWB.txt	1/3/2019 11:09 ...	TXT File
LAIDAY_AVE.tif	1/3/2019 11:09 ...	TIF File
LAIDAY_SUM.tif	1/3/2019 11:09 ...	TIF File
param.cali	1/3/2019 11:08 ...	CALI File
Q.txt	1/3/2019 11:09 ...	TXT File
QG.txt	1/3/2019 11:09 ...	TXT File
QI.txt	1/3/2019 11:09 ...	TXT File
QS.txt	1/3/2019 11:09 ...	TXT File
runlogs.txt	1/3/2019 11:09 ...	TXT File
SBGS.txt	1/3/2019 11:09 ...	TXT File
SBIF.txt	1/3/2019 11:09 ...	TXT File
SBOF.txt	1/3/2019 11:09 ...	TXT File
SBQG.txt	1/3/2019 11:09 ...	TXT File
SED.txt	1/3/2019 11:09 ...	TXT File
SED_OL_SUM.tif	1/3/2019 11:09	TIF File

31 items 1 item selected 26.4 KB

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., `data\youwuzhen\demo_youwuzhen30m_longterm_model`) which consists of several SEIMS configuration files such as `file.in`, `file.out`, `param.cali`, and `config.fig`. The `file.in` and `file.out` for basic model configuration are required, while `param.cali` for calibrated model parameters is optional. These three plain text files will be read and imported into MongoDB during data preprocessing (Section 2:3.2). The MongoDB also can be updated by only running the single Python script when these files are changed such as during the manual calibration, i.e., `preprocess/db_import_model_parameters.py`.

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 `TAG`s which must appear in this file.

- `MODE`: *A string that indicates it is 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 specified for storm event simulation considering the differences between hillslope scale processes and channel scale process. For example,* `INTERVAL|3600,86400` *means that the time-step for hillslope scale processes is one hour and one day 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.

```
# Long-term simulation with a time-step of one day
MODE|Daily
INTERVAL|1
# Model run from Jan 1, 2012 to Dec 31, 2015
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 watershed/subbasin 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`	Float	Time-step for output. The default is `-9999`, which means

Field name	Datatype	Description
		the same time-step with the simulation will be used.
`INTERVAL_UNIT`	String	Unit for `INTERVAL`, can be `DAY`, `HR`, 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., `C:\z_code\Hydro\SEIMS\seims\preprocess\database\AvailableOutputs.csv`. So, users can specify the outputs of `QRECH` (Streamflow at reach outlet of each time step, m^3/s) and `PET` (Potential evapotranspiration, mm) by simply specifying the output IDs, e.g.,

```
OUTPUTID
QRECH
PET
```

, or specifying more detail information except for the output IDs, 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.cali

The `param.cali` file is designed to define calibrated parameters. The absent of this file or a blank file is allowed and indicates the model will be simulated with default parameters. 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., `C:\z_code\Hydro\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.cali` 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. 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`. 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.*

As a demo, the `config.fig` file of the Youwuzhen watershed model is shown in Figure 2:4-3. More details about the selected modules can be found in Qin et al. (2018).

```
### Driver factors, including climate and precipitation
0 | TimeSeries | | TSD_RD
0 | Interpolation_0 | Thiessen | ITP
### Surface processes
0 | Soil temperature | Finn Plauborg | STP_FP
0 | PET | PenmanMonteith | PET_PM
0 | Interception | Maximum Canopy Storage | PI_MCS
0 | Snow melt | Snowpeak Daily | SNO_SP
0 | Infiltration | Modified rational | SUR_MR
0 | Depression and Surface Runoff | Linsley | DEP_LINSLEY
0 | Hillslope erosion | MUSLE | SERO_MUSLE
0 | Plant Management Operation | SWAT | PLTMGT_SWAT
0 | Percolation | Storage routing | PER_STR
0 | Subsurface | Darcy and Kinematic | SSR_DA
0 | SET | Linearly Method from WetSpa | SET_LM
0 | PG | Simplified EPIC | PG_EPIC
0 | ATMDEP | Atomosphere deposition | ATMDEP
0 | NUTR_TF | Nutrient Transformation of C, N, and P | NUTR_TF
0 | Water overland routing | IUH | IUH_OL
0 | Sediment overland routing | IUH | IUH_SED_OL
0 | Nutrient | Attached nutrient loss | NUTRSED
0 | Nutrient | Soluble nutrient loss | NUTRMV
0 | Pothole | SWAT cone shape | IMP_SWAT
0 | Soil water | Water balance | SOL_WB
### Route Modules, including water, sediment, and nutrient
0 | Groundwater | Linear reservoir | GWA_RE
0 | Nutrient | groundwater nutrient transport | NUTRGW
0 | Water channel routing | MUSK | MUSK_CH
0 | Sediment channel routing | Simplified Bagnold equation | SEDR_SBAGNOLD
0 | Nutrient | Channel routing | NutrCH_QUAL2E
```

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

2:4.3 Advanced usage

The SEIMS main programs include an OpenMP version and a version which is a hybrid of OpenMP and MPI (hereafter referred to as MPI&OpenMP version). 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.

2:4.3.1 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 <IP> -port <port> -sce <scenarioID>
-cali <calibrationID> -id <subbasinID>]
```

In which,

- `modelPath` *is the path of the SEIMS-based watershed model (see* `MODEL_DIR` *in the configuration file of data preprocessing, Section 2:3.2).*
- `threadsNum` *(Optional) is the number of threads used for OpenMP-based parallel computing, which must be greater or equal than 1 (default).*
- `layeringMethod` *(Optional) can be 0 and 1, which means the routing layering method based on flow direction algorithms of* `UP_DOWN` *(default, layering from source) and* `DOWN_UP` *(layering from outlet), respectively.*
- `IP` *is the IP address of MongoDB server. The default is 127.0.0.1 (i.e., localhost).*
- `port` *is the port number of MongoDB server, and the default is 27017.*
- `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.*
- `calibrationID` *is the ID of Calibration which has been defined in* `PARAMETERS` *collection of the main database. By default, the* `calibrationID` *is -1, which means no calibration will be applied.*
- `subbasinID` *is the subbasin that will be executed. 0 means the whole watershed. 9999 is reserved for Field version.*

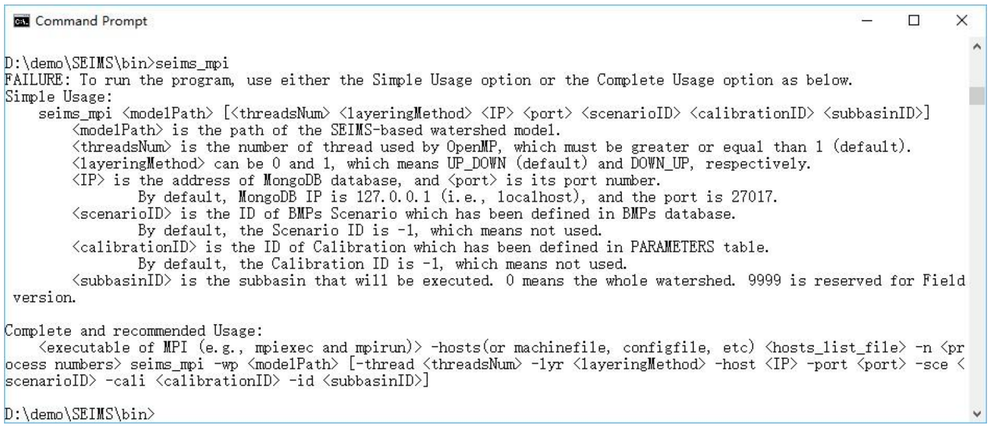
For the Youwuzhen watershed, one of the complete usages should be:

```
>cd D:\demo\SEIMS\bin
>D:
>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 is shown in Figure 2:4-1, in which the time-consuming of data Input/Output (IO), computation of each module, and entire simulation are shown.

2:4.3.2 MPI&OpenMP version

The MPI&OpenMP version of SEIMS main program has the same arguments with the OpenMP version but the different way of calling it (Figure 3:1-1).



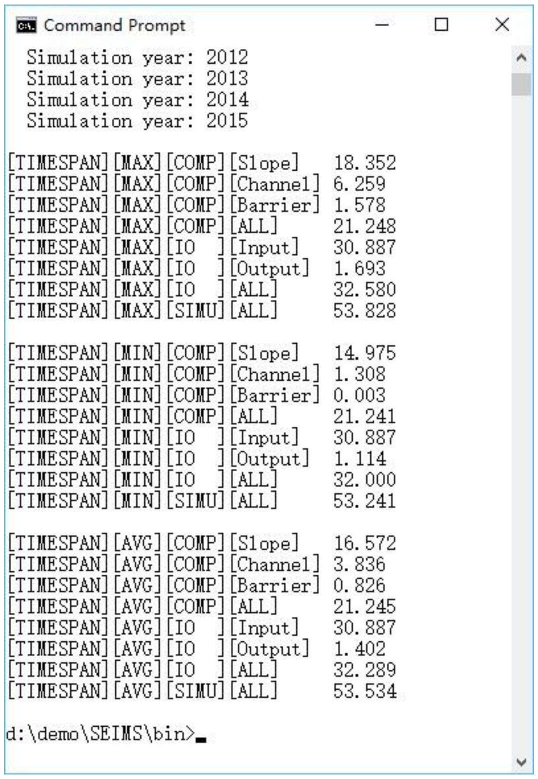
```
D:\demo\SEIMS\bin>seims_mpi
FAILURE: To run the program, use either the Simple Usage option or the Complete Usage option as below.
Simple Usage:
    seims_mpi <modelPath> [<threadsNum> <layeringMethod> <IP> <port> <scenarioID> <calibrationID> <subbasinID>]
        <modelPath> is the path of the SEIMS-based 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 <pr
ocess numbers> seims_mpi -wp <modelPath> [-thread <threadsNum> -lyr <layeringMethod> -host <IP> -port <port> -sce <
scenarioID> -cali <calibrationID> -id <subbasinID>]

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

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 should be:

```
>cd D:\demo\SEIMS\bin
>D:
>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 output information is like 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, module computing, 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.

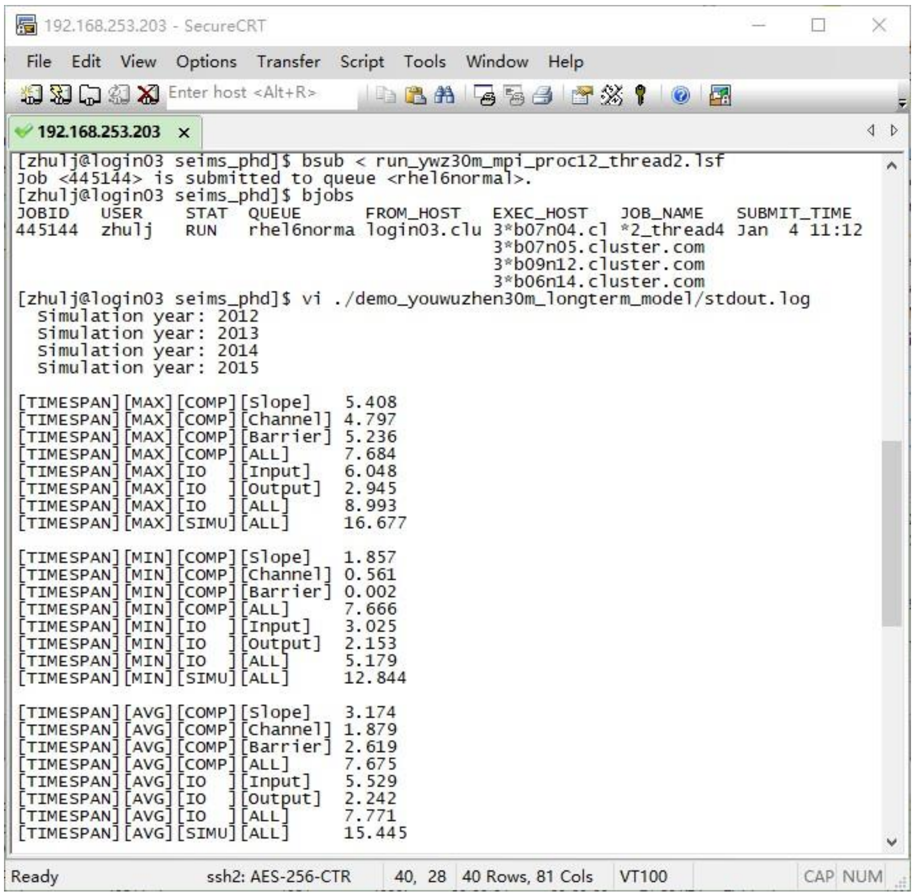


```
  Simulation year: 2012
  Simulation year: 2013
  Simulation year: 2014
  Simulation year: 2015

[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.lsf` 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.

More information about the parallel performance tests of a single run of the SEIMS-based watershed model can be referred to Zhu et al. (*Environ. Model. Softw.*, *under review*).



```
[zhulj@login03 seims_phd]$ bsub < run_ywz30m_mpi_proc12_thread2.lsf
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

2:4.3.3 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., replace the Penman-Monteith method (`PET_PM`) to Priestley-Taylor method (`PET_PT`) for the simulation of potential evapotranspiration. Note that the `PET_PM` module also simulates the potential plant transpiration while the `PET_PT` not, thus the `AET_PTH` module is added which is used to simulate the potential plant transpiration and potential and actual soil evaporation. Therefore, the soil evaporation module (`SET_LM`) is no longer need. The newly customized Youwuzhen watershed model (i.e., the `config.fig` file) is shown in Figure 2:4-7.

```
### Driver factors, including climate and precipitation
0 | TimeSeries | | TSD_RD
0 | Interpolation_0 | Thiessen | ITP
### Surface processes
0 | Soil temperature | Finn Plauborg | STP_FP
0 | PET | PriestleyTaylor | PET_PT
#0 | PET | PenmanMonteith | PET_PM
0 | Interception | Maximum Canopy Storage | PI_MCS
0 | Snow melt | Snowpeak Daily | SNO_SP
0 | Infiltration | Modified rational | SUR_MR
0 | Depression and Surface Runoff | Linsley | DEP_LINSLEY
0 | Hillslope erosion | MUSLE | SERO_MUSLE
0 | Plant Management Operation | SWAT | PLTMGT_SWAT
0 | Percolation | Storage routing | PER_STR
0 | Subsurface | Darcy and Kinematic | SSR_DA
0 | AET | Hargreaves and PriestleyTaylor | AET_PTH
#0 | SET | Linearly Method from WetSpa | SET_LM
0 | PG | Simplified EPIC | PG_EPIC
0 | ATMDEP | Atomosphere deposition | ATMDEP
0 | NUTR_TF | Nutrient Transformation of C, N, and P | NUTR_TF
0 | Water overland routing | IUH | IUH_OL
0 | Sediment overland routing | IUH | IUH_SED_OL
0 | Nutrient | Attached nutrient loss | NUTRSED
0 | Nutrient | Soluble nutrient loss | NUTRMV
0 | Pothole | SWAT cone shape | IMP_SWAT
0 | Soil water | Water balance | SOL_WB
### Route Modules, including water, sediment, and nutrient
0 | Groundwater | Linear reservoir | GWA_RE
0 | Nutrient | groundwater nutrient transport | NUTRGW
0 | Water channel routing | MUSK | MUSK_CH
0 | Sediment channel routing | Simplified Bagnold equation | SEDR_SBAGNOLD
0 | Nutrient | Channel routing | NutrCH_QUAL2E
```

Figure 2:4-7 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 2:4-8 shows the spatial distributions of average potential evapotranspiration simulated by the Penman-Monteith method (a) and Priestley-Taylor method (b), which have a very similar spatial pattern but different values. The differences of potential evapotranspiration can also be reflected in the streamflow at watershed outlet (Figure 2:4-9).

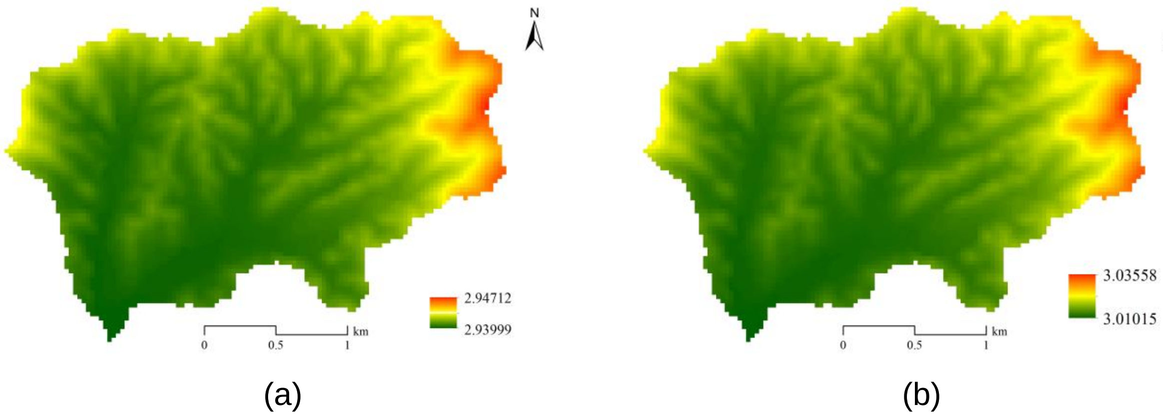


Figure 2:4-8 Spatial distributions of average potential evapotranspiration simulated by the Penman-Monteith method (a) and Priestley-Taylor method (b) based on the Youwuzhen watershed model

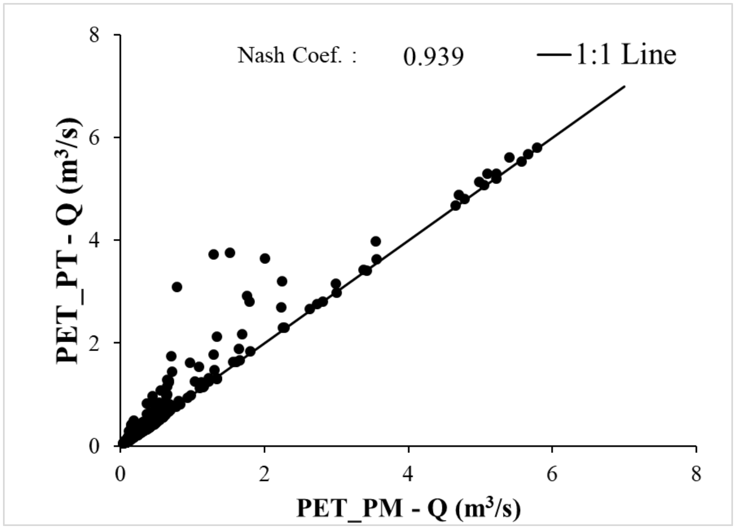


Figure 2:4-9 Differences of the streamflow (Q, m^3/s) at the outlet of Youwuzhen watershed derived from the Youwuzhen watershed models using the Penman-Monteith method (PET_PM) and Priestley-Taylor method (PET_PT) for the simulation of potential evapotranspiration, respectively.

2:5. Postprocessing

2:6. Parameters sensitivity analysis

2:7. Auto-Calibration

2:8. Scenarios analysis

Section 3.   Design and implementation

3:1. Overall design of SEIMS

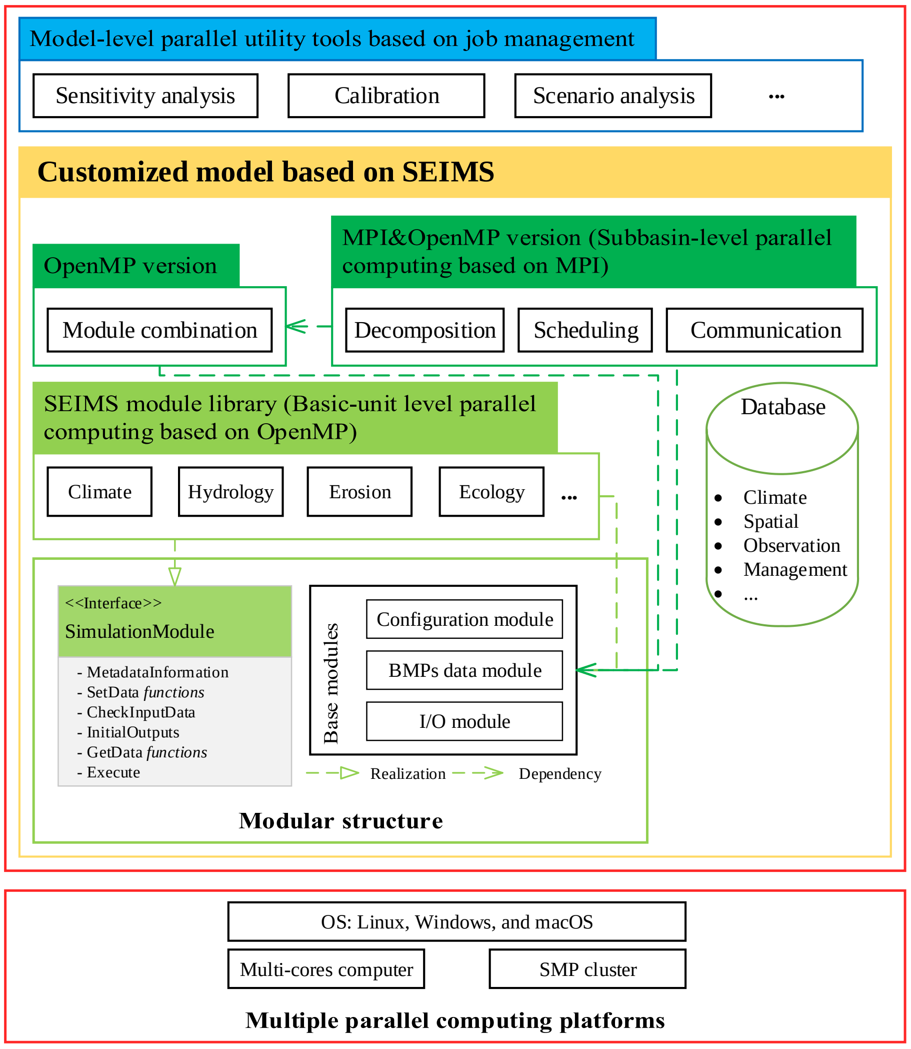


Figure 3:1-1 Architecture of Spatially Explicit Integrated Modeling System (SEIMS)

3:2. Flow charts of simulation

3:3. Modular structure

3:4. Parallel computing middleware

3:5. Watershed database

3:6. Coding protocol

Section 4. Write your own code

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

4:2. Rewrite existing watershed models into SEIMS