iEMSs 2019 1st Regional Conference (Asian Region), May 18-20, 2019, Nanjing, China

Development of a modular and parallelized watershed modeling framework

Liang-Jun ZHU, Junzhi LIU, Cheng-Zhi QIN, A-Xing ZHU

(State Key Laboratory of Resources and Environment Information System,
Institute of Geographic Sciences and Natural Resources Research,
Chinese Academy of Sciences)

2019-5-20

Contents

1 Background & study issue

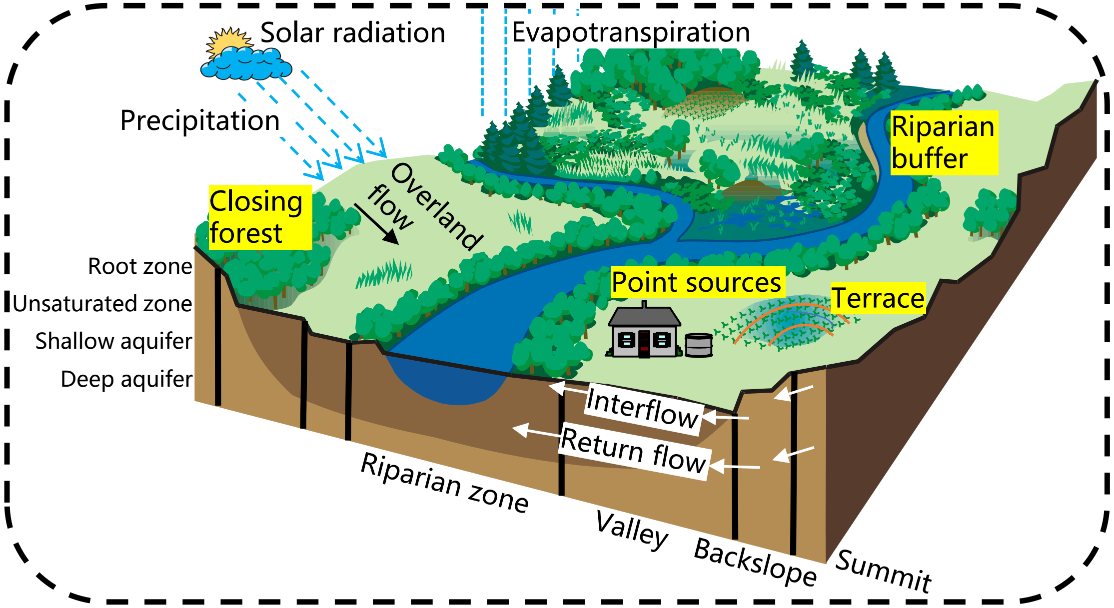
Watershed modeling has been widely used in hydrology studies and watershed management.

Step 1: **Select or customize a watershed model**

Step 2: **Collect and preprocess input data**

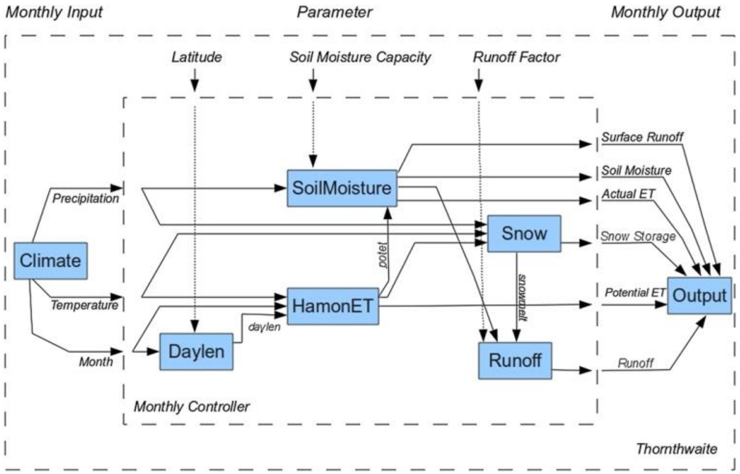
Step 3: **Perform parameters sensitivity analysis and auto-calibration**

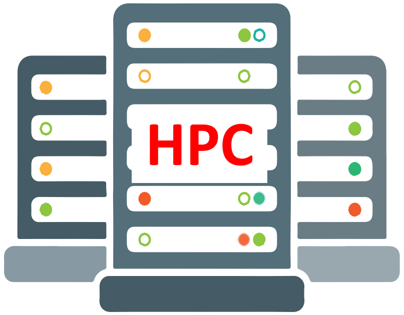
Step 4: **Analyze and apply the calibrated model**



Two major issues

- **A flexible and extensible modeling framework** is needed to meet **various modeling purposes** (Kneis, 2015).
- **Parallel computing** is required since a large amount of computation is needed by both the **model itself** and the **model-level applications** (Clark et al., 2017; David et al., 2013; Zhang et al., 2013).





Existing modeling framework

- **Environmental Modeling Framework (EMF)**
  - ✓ **Standard interfaces** for coupling existing models
  - ✓ **Parallel computing** support for common operations (e.g., regridding)
  - × **May not provide specific support for the parallelization of spatially explicit watershed modeling**

ESMF

OpenMI

CSDMS
community surface
dynamics modeling system

- **Watershed Modeling Framework**
  - ✓ **EMF specifically designed for watershed modeling**, e.g., OMS3 (David et al., 2013) and ECHSE (Kneis, 2015).
  - × **Shared-memory multithreaded programming** (e.g., **OpenMP**), **limits their scalability** on distributed-memory platforms (e.g., **SMP cluster**).

Existing parallelization strategies

- ➢ **Parallelization strategies for specific watershed models**
  - ✓ Effectively utilize both **SMP cluster and shared-memory parallel platforms**
  - ✓ Based on **spatial discretization** or **spatio-temporal discretization** and implemented by **MPI or a hybrid of MPI and OpenMP** (Liu et al., 2016; Vivoni et al., 2011; Wang et al., 2013; Yalew et al., 2013)
  - × **Require high parallel programming skills**
  - × **Often lack standard and concise module interfaces**

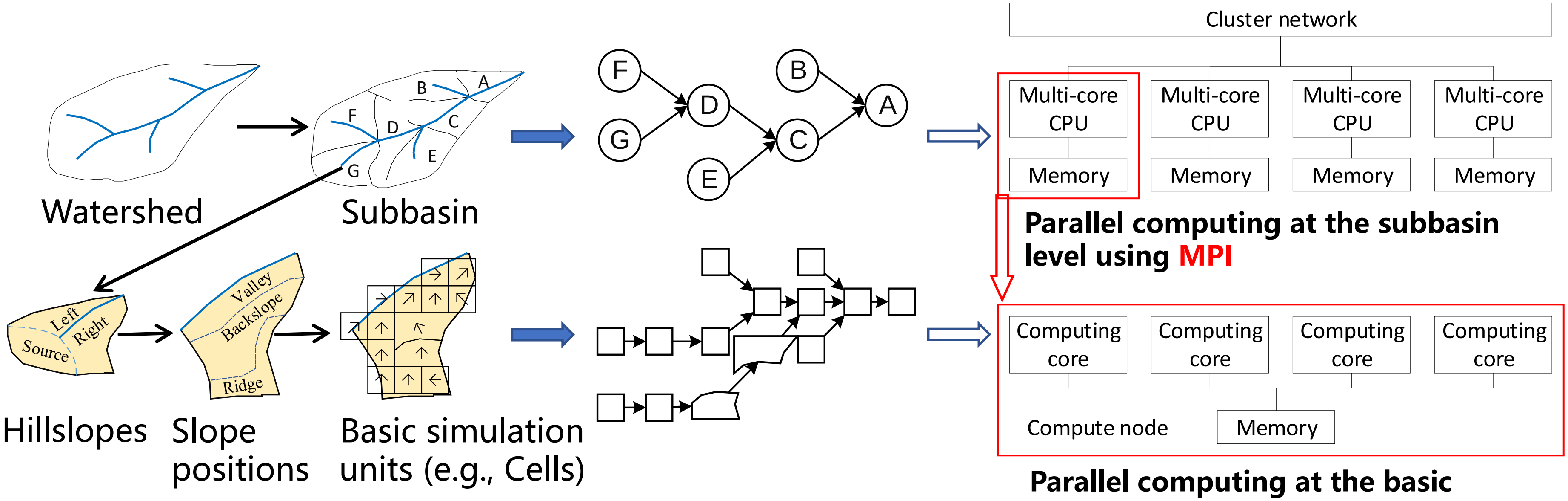
Study issue

How to develop a **flexible and efficient watershed modeling framework** to facilitate rapid development of parallelized watershed models on multiple parallel platforms?

- **flexible and extensible modular structure**
- **efficient and easy-to-use parallel computing middleware**

2 Basic idea & overall design

- A watershed can be partitioned into **spatial hierarchical units** (Band, 1999)
- Upstream-downstream orders based on **flow direction** (Liu et al., 2014; Wang et al., 2011)
- “subbasin-basic simulation unit” two-level parallelization strategy (Liu et al., 2014, 2016)



Watershed

Subbasin

Hillslopes

Slope positions

Basic simulation units (e.g., Cells)

Cluster network

Multi-core CPU

Memory

Multi-core CPU

Memory

Multi-core CPU

Memory

Multi-core CPU

Memory

Parallel computing at the subbasin level using MPI

Computing core

Computing core

Computing core

Computing core

Compute node

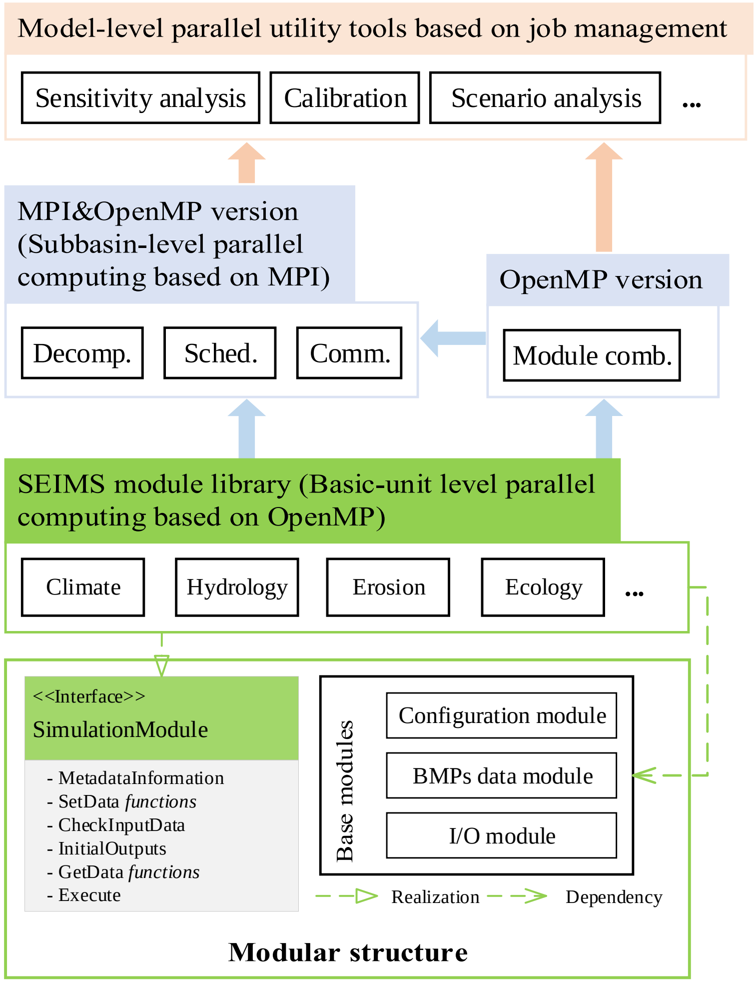
Memory

Parallel computing at the basic simulation unit level using OpenMP

2 Basic idea & overall design

SEIMS, short for **Spatially Explicit Integrated Modeling System**

- **Flexible modular structure**
  - Standard and concise interface
  - Nearly serial programming based on **OpenMP for basic simulation unit level**
  - **Without taking care of parallel programming details for subbasin level based on MPI**
- **Parallel computing middleware**
  - Basic simulation unit level (**OpenMP**)
  - Subbasin level (**MPI**)
  - **Model level** (**job management**)



Model-level parallel utility tools based on job management

Sensitivity analysis | Calibration | Scenario analysis | ...

MPI&OpenMP version (Subbasin-level parallel computing based on MPI)

Decomp. | Sched. | Comm.

OpenMP version

Module comb.

SEIMS module library (Basic-unit level parallel computing based on OpenMP)

Climate | Hydrology | Erosion | Ecology | ...

<<Interface>>
SimulationModule
- MetadataInformation
- SetData *functions*
- CheckInputData
- InitialOutputs
- GetData *functions*
- Execute

Base modules

Configuration module

BMPs data module

I/O module

Realization

Dependency

Modular structure

3 Implementation

SEIMS: A **modular and parallelized** watershed modeling framework

- **C++:** SEIMS main programs and modules
- **Python:** Utility tools with **model-level parallel computing**, e.g., sensitivity analysis, auto-calibration, and scenario analysis.
- **Current SEIMS module library:** contains hydrology, erosion, nutrient cycling, and plant growth processes from **WepSpa, SWAT, LISEM**, etc.
- **MongoDB database:** flexible data management
- **Open-source** (https://github.com/lreis2415/SEIMS)

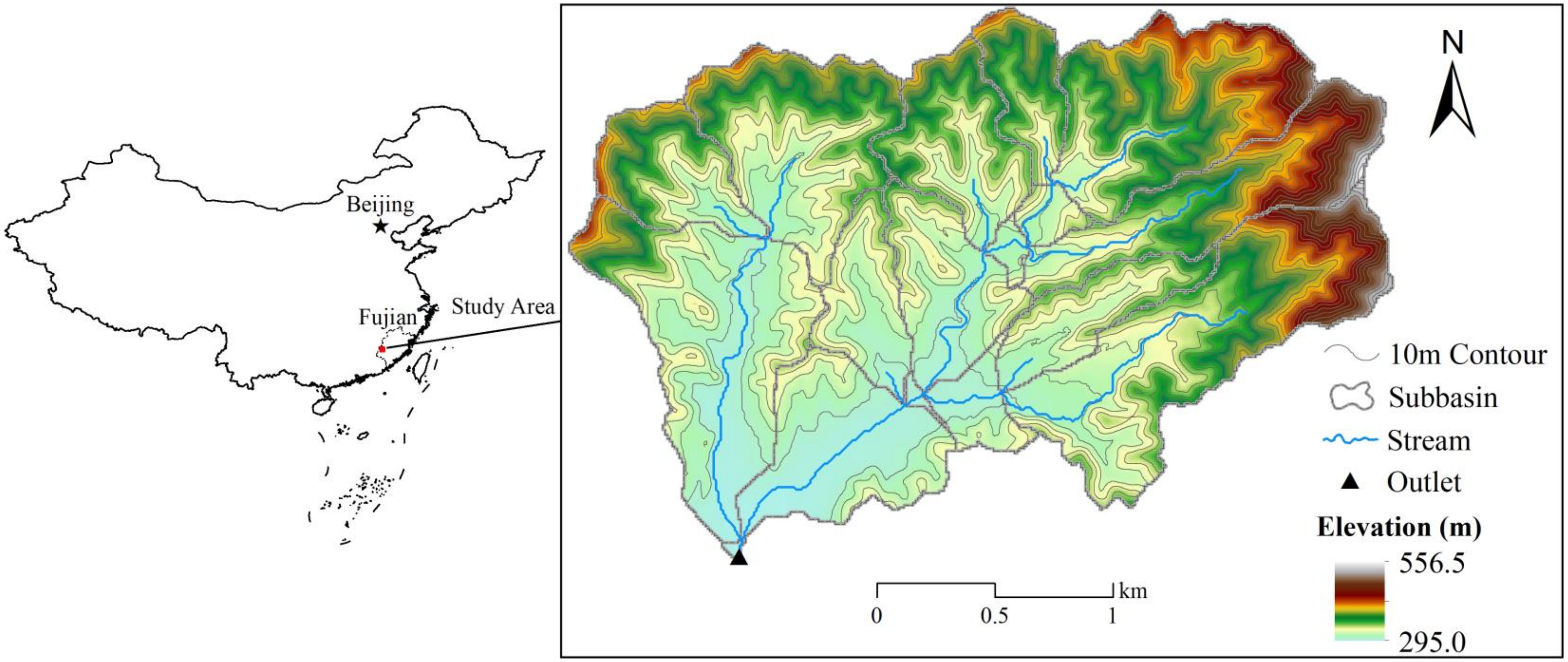
How to contribute modules/algorithms?

- **Create a new SEIMS module using module template** (i.e., copy the module template files to a new folder) **and finish the module interface**:
  - **Transplant** existing C++ execution code directly
  - **Rewrite** existing code of other languages
  - **Write** from the beginning based on the principle and formulas
- The degree of difficulty depends on the user’s **understanding of the watershed subprocesses and the module interface, as well as the user’s programming experience**.
- Please refer to the **SEIMS user manual** for more detailed tutorials. (https://github.com/lreis2415/SEIMS/blob/master/SEIMS-UserManual.pdf)

4 Case study

Hydrological modeling of the Youwuzhen watershed, Fujian province, China.

- **53,933 grid cells** with a 10 m resolution
- **17 subbasins** delineated
- **Daily** climate and hydrological observed data from **2012 to 2015**



Experimental environments

- Hardware
  - **A Linux cluster** with 134 computing nodes
  - Each computing node with 2-way Intel® Xeon® E5650 6-cores CPUs (i.e., 12 physical cores in total), 24 GB memory, and one InfiniBand (40 GB/s) network card
- Software
  - Red Hat® Enterprise Linux® Server 6.2
  - **Intel® C++ 12.1 compiler** with the support of OpenMP 3.1, Intel® MPI Library 4.0.3, GDAL-1.11.5, and mongo-c-driver-1.6.1
  - **Python-2.7.13** with several third-party packages such as GDAL-1.11.5, NumPy-1.12.1, matplotlib-1.5.3, pymongo-3.4.0, DEAP-1.2, SALib-1.1.2, and SCOOP-0.7

Construction of the SEIMS-based model

- **Configuration file** (selected SEIMS modules in sequence)

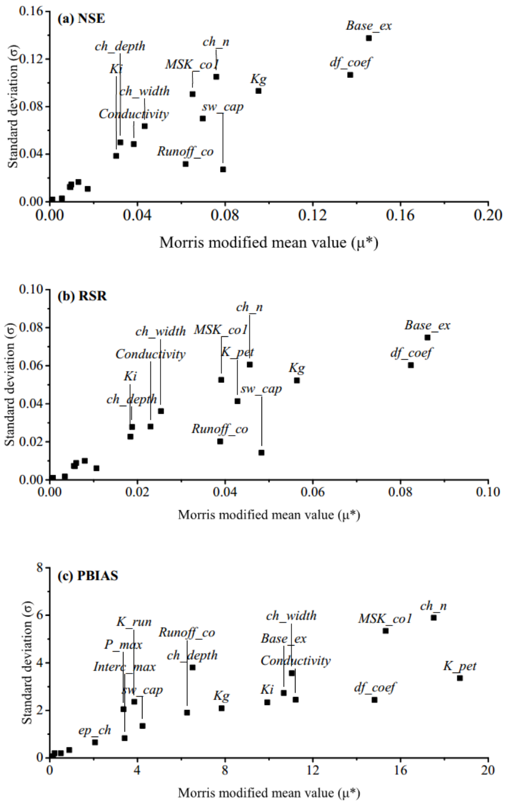
```
### Driver factors, including meteorological data and precipitation
1 | TimeSeries | | TSD_RD
2 | Interpolation_0 | Thiessen | ITP
### Hillslope processes
3 | Soil temperature | Finn Plauborg | STP_FP
4 | PET | PenmanMonteith | PET_PM
5 | Interception | Maximum Canopy Storage | PI_MCS
6 | Snow melt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

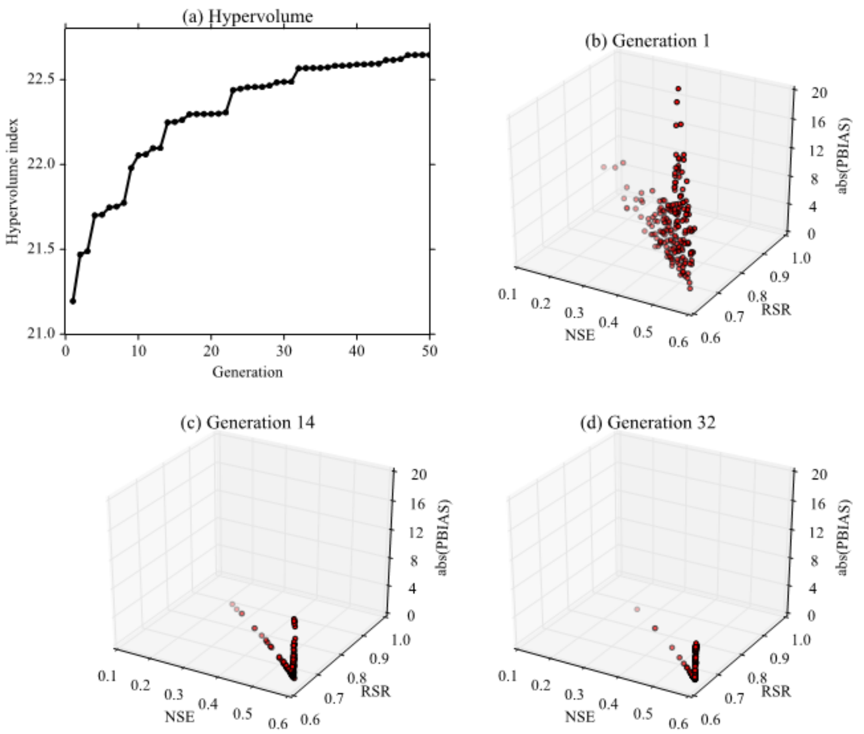
Loading and preprocessing driver factors, e.g., climate data

Hillslope processes, e.g., potential evapotranspiration, canopy interception, depression storage, surface runoff, percolation, interflow.

Channel flow routing processes

Parameters sensitivity analysis and auto-calibration





Morris screening method

- 2410 SEIMS-based model runs
- 2.12 hr of the parallel job
- 149.78 hr of all individual models

→ 70.67 times speedup

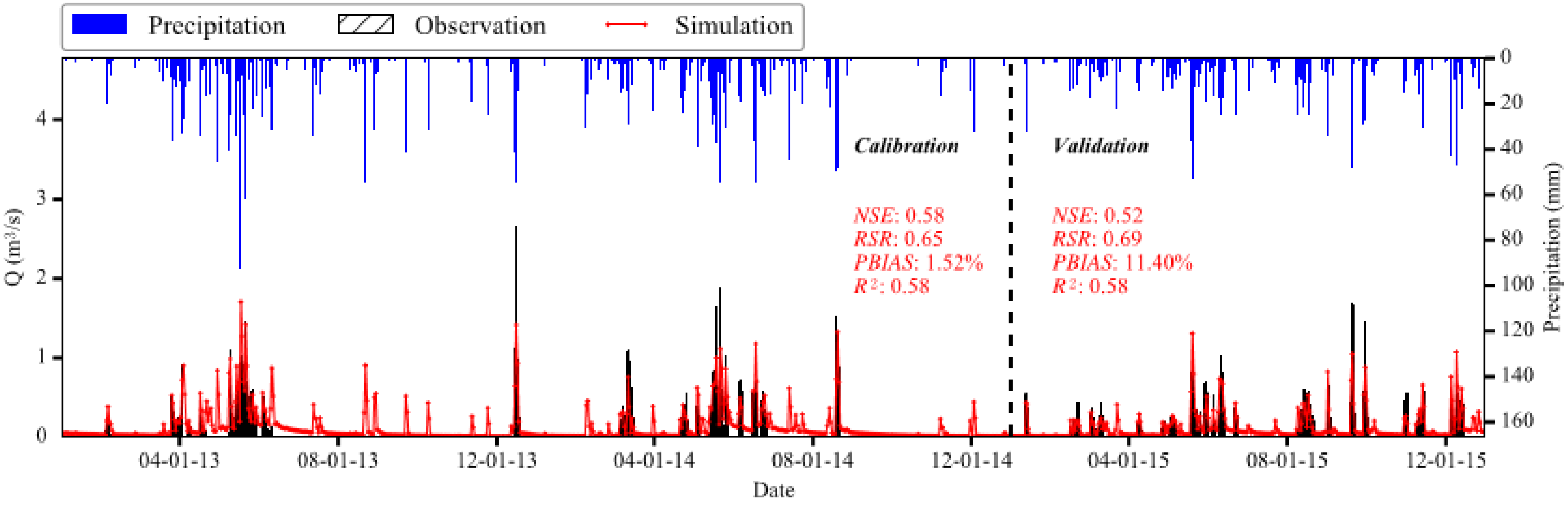
NSGA-II algorithm

- 7916 SEIMS-based model runs
- 10.74 hr of the parallel job
- 695.57 hr of all individual models

→ 64.79 times speedup

One selected calibrated model

Simulated flow discharge (m^3/s) at the watershed outlet



Calibration: NSE: 0.58, RSR: 0.65, PBIAS: 1.52%, R^2: 0.58

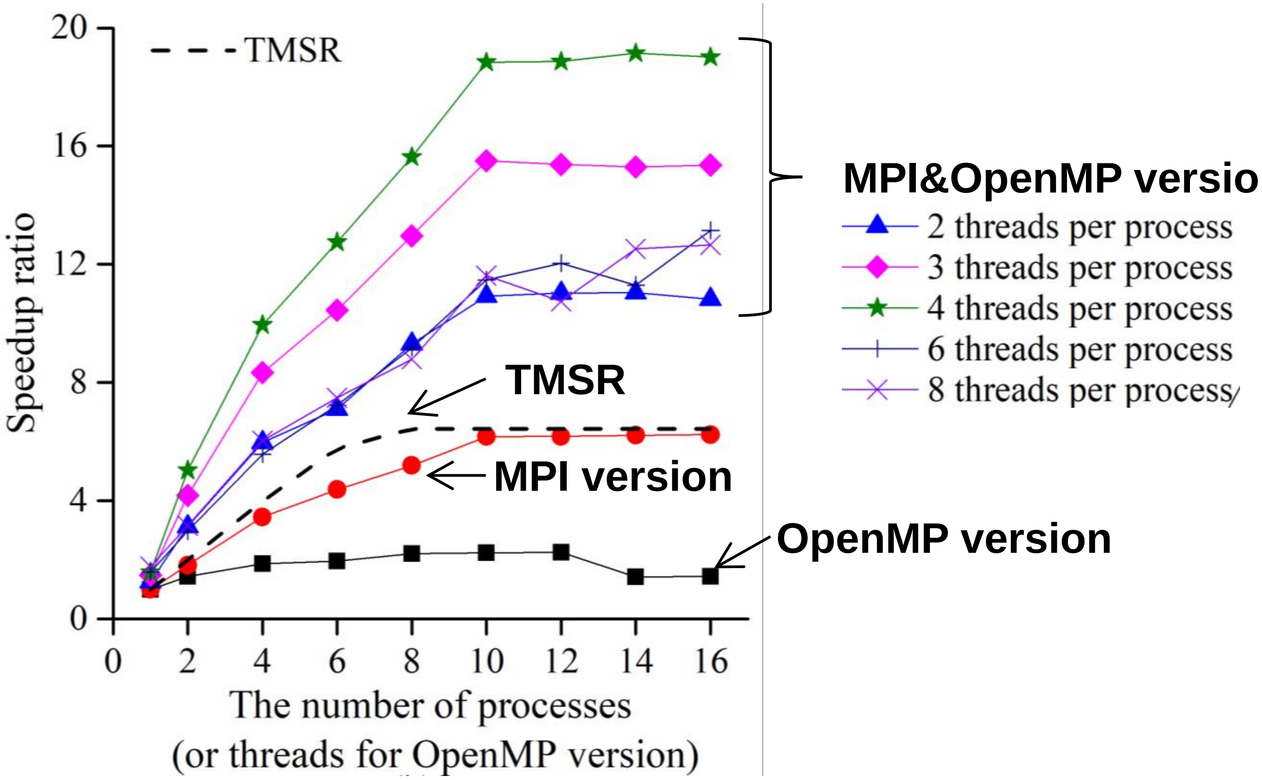
Validation : NSE: 0.52, RSR: 0.69, PBIAS: 11.40%, R^2: 0.58

Experiments of parallel performance of single model run

- **The <u>OpenMP version</u>** of the SEIMS-based model was executed with 1, 2, 4, 6, 8, 10, 12, 14, and 16 threads, respectively.
- **The <u>MPI&OpenMP version</u>** was executed with the same counts of processes and also with different thread counts (i.e., 1, 2, 3, 4, 6, and 8 threads). **MPI&OpenMP version with 1 thread per process can be regarded as <u>MPI version</u>.**
- **Speedup ratio:** Ratio between serial and parallel computing times
- **Benchmark**: Theoretical maximum speedup ratio **(*TMSR*) at the subbasin level** (Liu et al., 2013)

Parallel performance of single model run

Speedup ratio



MPI&OpenMP version

TMSR

MPI version

OpenMP version

- **Subbasin level parallelization (MPI version) is greater than that of basic simulation unit level (OpenMP version).**
- **The two-level parallelization is dramatically improved than any single level parallelization and greater than *TMSR*.**

5 Summary

SEIMS: A modular and parallelized watershed modeling framework

- **Flexible and extensible:** Modular structure
- **High performance:** Multi-level parallel computing middleware
- **Easy-to-use:** Transplant/rewrite/write new SEIMS modules in a nearly serial programming manner
- **Cross-platform:** Common OS (Windows, Linux, and macOS) and parallel computing platforms (personal computers with multi-core CPU and SMP cluster)
- **Open-source:** https://github.com/lreis2415/SEIMS

5 Summary

Next move…

- Support of irregularly shaped fields as basic simulation units
- Support of multiple flow direction model
- Support of multiple parallel task scheduling at the subbasin level
- **Transplant/Rewrite other successful watershed models** (e.g., DHSVM)
- **Integrate into a web-service based and user-friendly modeling platform**

Zhu L-J, Liu J*, Qin C-Z*, Zhu A-X. A modular and parallelized watershed modeling framework. ***Environmental Modelling & Software***. (Under review after major revision)

Acknowledgements: National Natural Science Foundation of China (41431177, 41601413, 41871362, and 41871300), the Innovation Project of LREIS (O88RA20CYA), National Basic Research Program of China (2015CB954102), PAPD, and Outstanding Innovation Team in Colleges and Universities in Jiangsu Province.

朱良君 (Liang-Jun Zhu)

SEIMS: *https://github.com/lreis2415/SEIMS*

Email: zlj@lries.ac.cn