https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences
Discussions

# Improving computational efficiency of GLUE method for hydrological model uncertainty and parameter estimation using CPU-GPU hybrid high performance computer cluster

Depeng Zuo [a], Guangyuan Kan [a, b, *], Hongquan Sun [c], Hongbin Zhang [b], Ke Liang [b, *]

[a] Beijing Key Laboratory of Urban Water Cycle and Sponge City Technology, College of Water Science, Beijing Normal University, Beijing 100875, P. R. China

[b] State Key Laboratory of Simulation and Regulation of Water Cycle in River Basin, Research Center on Flood & Drought Disaster Reduction of the Ministry of Water Resources, China Institute of Water Resources and Hydropower Research, Beijing 100038, P. R. China

[c] National Institute of Natural Hazards, Beijing 100085 P.R. China

* Corresponding authors:
kangy@iwhr.com (Guangyuan Kan), liangkepapers@126.com (Ke Liang)

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences
Discussions

**Abstract:** The Generalized Likelihood Uncertainty Estimation (GLUE) method has been thrived for decades, huge number of applications in the field of hydrological model have proved its effectiveness in uncertainty and parameter estimation. However, for many years, the poor computational efficiency of GLUE hampers its further applications. A feasible way to solve this problem is the integration of modern CPU-GPU hybrid high performance computer cluster technology to accelerate the traditional GLUE method. In this study, we developed a CPU-GPU hybrid computer cluster-based highly parallel large-scale GLUE method to improve its computational efficiency. The Intel Xeon multi-core CPU and NVIDIA Tesla many-core GPU were adopted in this study. The source code was developed by using the MPICH2, C++ with OpenMP 2.0, and CUDA 6.5. The parallel GLUE method was tested by a widely-used hydrological model (the Xinanjiang model) to conduct performance and scalability investigation. Comparison results indicated that the parallel GLUE method outperformed the traditional serial method and have good application prospect on super computer clusters such as the ORNL Summit and Sierra of the TOP500 super computers around the world.

**Key words:** GLUE; Xinanjiang model; MPI; OpenMP; CUDA

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences
Discussions

## 1. Introduction

### *1.1 GLUE-based hydrological model uncertainty and parameter estimation*

The uncertainty in hydrological processes affects the prediction accuracy of the hydrological models (Dong et al., 2015; Kan et al., 2015; Kan et al., 2016; Kan et al., 2017; Kan et al., 2018; Lei et al., 2016; Li et al., 2016; Li et al., 2014; Zuo et al., 2016). The uncertainties can be categorized into three types, including model parameter uncertainty, input data uncertainty, and model structure uncertainty (Lindensehmidt et al., 2007). Among them, the model parameter uncertainty analysis and parameter estimation play important roles to the model simulation and flood forecasting (Dong et al. 2019; Duan et al. 2018, 2019; Gao et al. 2018, 2019). Uncertainties in hydrological simulation brings difficulties in water resource management and hydrological risk reduction (Zhu et al., 2018; Yu et al., 2019; Shao et al., 2016,2018a,b; Wang et al. 2018). Therefore, uncertainty and parameter estimation have become a necessary procedure in the application of hydrological models (Li, 2005).

Until recently, hydrologists have carried on comprehensive studies on uncertainty analysis and parameter estimation on hydrological model and have obtained significant achievements. The Generalized Likelihood Uncertainty Estimation, GLUE, has become the most widely applied model uncertainty analysis and parameter estimation method. The GLUE method was proposed by Beven in 1992 (Beven and Binley, 1992) for studying the equifinality phenomenon of hydrological model, and then it has been widely applied in hydrology studies. Li (Li and Liang, 2006; Shu et al., 2008) used three different typical watersheds as examples and adopted the GLUE method to study the parameter uncertainty issue of the Xinanjiang model, which assessed and conformed the applicability of the GLUE method. Huang (Huang and Xie, 2007) applied the GLUE method in the Xingfeng watershed of the Dong River and studied the TOPMODEL uncertainty in hydrological forecasting. Muleta (Muleta M and Nicklow, 2005) studied the discharge and sediment simulation by using the SWAT model in the Creek catchment in Illinois state of the U.S.A. and considered that the sediment simulation results have higher uncertainty than the discharge simulation.

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences
Discussions

Xu (Xu and Li, 2009) applied the GLUE method for the SWAT model uncertainty assessment and parameter estimation in headwaters of the Hei River.

When utilizing the GLUE method, many hydrologists improved the algorithm for the purpose of studying features of the watersheds or satisfying various requirements. Blasone (Blasone and Vrugt, 2008) replaced the Monte Carlo random sampling of the GLUE by the SCEM-UA sampling. They selected the feasible parameter sets according to the coverage of the estimated confidential intervals and therefore significantly improved the reliability of the confidential intervals. Wei (Wei and Xiong, 2008) applied the above-mentioned method in upper stream of the Han River and obtained better estimation results. Liu (Liu, 2008. Liu et al., 2009) improved the likelihood criterion of the GLUE method and added multiple criteria likelihood functions based on China's Specification for Hydrological Forecasting. The multiple criteria likelihood functions promoted the uncertainty analysis capability and accuracy of the GLUE method. Lin and Chen (Lin et al., 2009) described the model parameter relativity by using Copula function and proposed Copula-Glue based parameter uncertainty estimation method for hydrological models. They also compared the simulation result with the original GLUE. Deng (Deng et al., 2008) generated the synthetic time series data to eliminate the influences of the data and model structure and analyzed the effects of likelihood function of the pollutant attenuation model. They found that the likelihood function has great impact on the analysis results.

*1.2 Previous studies on the acceleration of the GLUE method*

The computational cost of the GLUE method is expensive. It is due to the extensive feasible parameter searching space of the Monte Carlo experiments combined with the huge number of hydrological model objective function evaluations. The traditional GLUE method consumes too much time which prevent the researchers and engineers from applying it to highly complex real-world applications. This is a huge impedance to the development of GLUE-based uncertainty analysis and parameter estimation methods.

Binley (Binley and Beven, 1991) attempt to assess the uncertainty associated with predictions of distributed models using a distributed rainfall-runoff model. However,

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences
Discussions
Open Access
EGU

he recognized that the computational constraints of the Monte Carlo simulations and examined the method of Rosenblueth (1975) which requires 2N+1 simulations with N as the number of model parameters, to make an approximate estimate of the prediction uncertainty. Binley constrained their Monte Carlo sampling to 500 realizations even though they adopted a relatively simple distributed model, the Institute of Hydrology Distributed Model version 4 (IHDM4). However, at that time，performing this level of computation required significant code enhancement in order to exploit a 80 node transputer parallel computer (Beven and Binley, 2014).

Later on, Beven (Beven and Binley, 1992) noticed that the GLUE procedure is computationally intensive and improved the computational efficiency by implementing it on a local parallel processing computer. The transputer utilized by Binley and Beven was a 1980s parallel computer which was designed by David May at the University of Bristol and produced by Inmos, with chips designed to support pipes to other processors. The first floating point transputer, the T800 was produced in 1987. It was used by Binley as TRAM daughter boards for PCs and programmed in a language called Occam. Later on, the simulations run on a parallel computer of Lancaster University Meiko Computing Surface, which contains some 80 transputers, each of which provides about 1 Mflops average performance. The GLUE procedure is adapted to a distributed memory parallel processor, particularly where each realization may be loaded onto a single processor. In general, 500 realizations of the IHDM were run for each storm using a 50 transputer array. Each set of runs for each storm took between 30 and 60 hours of computing time on the 50 processors. In that era, these pioneer studies were trying to employ hardware and software that was new to hydrological sciences and relevant disciplines.

With the development of computational performance, constraints of computer on the application of GLUE have been relaxed. However, it remains an issue, either because of a model that is particularly slow to run so that it is still not possible to sample sufficient realizations or because of high number of parameter dimensions (Beven and Binley, 2014). The largest number of runs used in a GLUE application that we know of are the two billion runs application (Iorgulescu, 2005; Iorgulescu,

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences
Discussions

2007), of which 216 were accepted as behavioral using a limits of acceptability approach. This was for a model written by just a few lines of code but including 17 parameters for calibration. Two billion runs are still a small sample compared with a discrete sampling strategy with ten values for each parameter. They were constrained to 500 realizations for each (relatively short) event and that was only possible in a reasonable time because they utilized an 80 node transputer system. More recently, GLUE calculations have been speeded up for certain models using parallel graphics processor cards (Beven et al., 2012). Even though a number of researchers have studied on the acceleration of GLUE, little research fully utilized the huge computational horsepower of the new generation CPU (Central Processing Unit)-GPU (Graphics Processing Unit) hybrid high performance computer cluster and its software ecosystems. With the development of modern heterogeneous parallel computing technology, new generation hardware integrated with their versatile software development tools can provide tremendous computing horsepower and much better energy efficiency than ever before. The acceleration of GLUE method should catchup with the state-of-the-art of modern high-performance computing technology. The acceleration of GLUE by using new generation CPU-GPU hybrid high performance computer cluster is of great significance and is in great need.

*1.3 Proceedings of the modern parallel computing technology*

With the development of modern microelectronic technology, multi-core and many-core hybrid heterogeneous parallel computing platform has become the upstart in recent high-performance computing field, owe to its stupendous floating-point compute capability compared with traditional and old generation computers. Until recently, several heterogeneous super computers, such as Summit and Sierra, have shown excellent performance on the TOP500 test. CPU-GPU heterogeneous platform successes owe to its better cost performance and energy consumption ratio. The modern CPU-GPU hybrid computing platform has become the best choice for researchers and engineers who need high-performance computing (Fang et al., 2016).

On the other hand, the GPU has been widely equipped in modern PCs, and the CPU-GPU hybrid heterogeneous computer systems is easily available for scientific

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


computing. Although the available GPU on PCs mainly focuses on the gaming and entertainment tasks other than floating point computation, these devices still perform very well in many applications that do not require double precision computations. Besides, the software development toolkits of the CPU-GPU hybrid platform can be acquired for free. Therefore, the CPU-GPU heterogeneous parallel computing platform can be established easily at relatively low cost. The popularization of the graphics processing cards enables the CPU-GPU hybrid parallel program to execute on almost all PCs (Fang et al., 2016).

The utilization of computational power of hybrid CPU-GPU system is codification on heterogeneous systems. Nevertheless, heterogeneous parallel software development faces great challenges such as heterogeneous data communication, programming and optimization based on GPU architecture, joint compilation of multiple compilers, etc. Owing to the Intel, NVIDIA, and other hardware and software producers' hard works, many powerful and easy-to-use compilers and libraries have emerged which include GCC, ICC, PGI, VC, NVCC, MPI, OpenMP, and OpenCL, etc. With these useful tools, source code development for heterogeneous parallel computing is easier to accomplish than before.

*1.4 Problems needed to be resolved and content of this paper*

With the arrival of the big data era, hydrological model uncertainty and parameter estimation require unprecedented large amount of computing horsepower. This research focused on the GLUE method and the newly emerged modern CPU-GPU hybrid high performance computer cluster acceleration technology. In order to further improve the computational efficiency of the GLUE-based Xinanjiang hydrological model uncertainty and parameter estimation, the CPU-GPU hybrid computer cluster-based parallel GLUE-Xinanjiang uncertainty and parameter estimation method was proposed. The parallel method was implemented on CPU cluster and GPU cluster, respectively. We utilized totally 5 CPUs and 5 GPUs to achieve good acceleration results. At last, the scalability issue was also investigated to prove the satisfactory robustness and portability of the parallel GLUE method.

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences
Discussions

## 2. Methodology

### *2.1 Traditional GLUE-based Xinanjiang model uncertainty and parameter estimation*

#### *2.1.1 Traditional GLUE method*

The requirements of the traditional GLUE procedure for model uncertainty and parameter estimation are listed below:

1) A formal definition of a likelihood measure or set of likelihood measures is required. For hydrological model applications, the Nash-Sutcliffe coefficient of efficiency (*NSCE*) is usually adopted as the likelihood measure or objective function. It can be calculated as follows:

$$NSCE = 1 - \frac{\sum_{i=1}^{n}\left(q_{\mathrm{obs},i} - q_{\mathrm{sim},i}\right)^2}{\sum_{i=1}^{n}\left(q_{\mathrm{obs},i} - \bar{q}_{\mathrm{obs}}\right)^2} \quad (1)$$

where $q_{\mathrm{sim},i}$ denotes simulated discharge at time step $i$; $q_{\mathrm{obs},i}$ denotes observed discharge at time step $i$; $\bar{q}_{\mathrm{obs}}$ denotes mean of the observed discharge values; $n$ denotes the number of discharge data.

2) An appropriate definition of the initial range or distribution of parameter values is necessary for a particular model structure. Generally speaking, the ranges of parameters are predefined according to the parameter physical meanings of the specific hydrological model and the uniform distribution is adopted in most cases since the distribution of parameters are usually unknown.

3) Sampling the parameter sets in the feasible space can be achieved by utilizing the Monte Carlo approach, and their likelihood values should be evaluated with the objective function after obtaining the simulation results of the hydrological model.

4) Uncertainty band or optimality of different parameter sets can be evaluated based on their likelihood value.

#### *2.1.2 The Xinanjiang hydrological model*

The Xinanjiang model was developed in 1973 and published in 1980 (Zhao et al., 1980). Its main feature is the concept of runoff formation on repletion of storage, which means that runoff is not generated until the soil moisture content of the vadose

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences
Discussions

zone reaches field capacity, and thereafter runoff equals the excess rainfall without further loss. This hypothesis was first proposed in the 1960s, and much subsequent experience supports its validity for humid and semi-humid regions. According to the original formulation, runoff so generated was separated into two components using Horton's concept of a final, constant, infiltration rate. Infiltrated water was assumed to go to the groundwater storage and the remainder to surface, or storm runoff. However, evidence of variability in the final infiltration rate, and in the unit hydrograph assumed to connect the storm runoff to the discharge from each sub-basin, suggested the necessity of a third component. Guided by the work of Kirkby, an additional component, interflow, was provided in the model in 1980. The modified model is now successfully and widely applied in China (Singh, 1995). The model structure is demonstrated in figure 1. Detailed descriptions of principles of the Xinanjiang model can be found in relevant literatures (Zhao, 1983; Zhao, 1992; Zhao, 1993; Zhao, 1994).

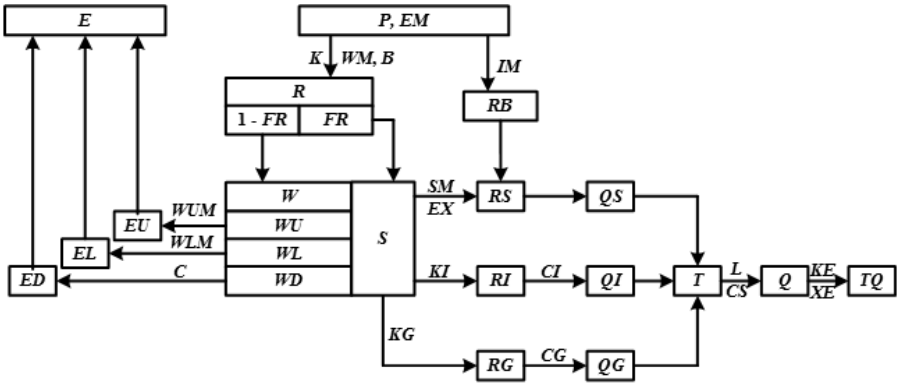


**Fig. 1.** The structure of the Xinanjiang model.

*2.1.3 Traditional uncertainty and parameter estimation*

The traditional GLUE-based Xinanjiang model uncertainty and parameter estimation involves two aspects: a) parameters upper and lower boundaries and their constraints; b) the objective function or likelihood measurement. Additionally, for hydrological simulation and optimization with the Xinanjiang model both the computation time step and hydro-meteorological data time interval were set to 1 day.

For model parameters, the specification of lower and upper boundaries is listed in

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences Discussions

Table 1. For the Xinanjiang model $n$ = 15 ($n$ is the number of parameters need to be optimized). Parameters *KG* and *KI* of the linear reservoir flow concentration module has a structural constraint that *KG*+*KI*=0.7. Therefore, we optimize *KG* in this research and *KI* is calculated by 0.7-*KG*.

**Table 1** Parameters and their boundaries of the Xinanjiang model

| Parameters | Physical meaning | Range and unit |
|---|---|---|
| *K* | Ratio of potential evapotranspiration v.s. pan evaporation | 0.1-1.5 |
| *B* | Distribution of tension water capacity coefficient | 0.1-0.4 |
| *C* | Deeper layer evapotranspiration coefficient | 0.08-0.2 |
| *WUM* | Upper soil layer water capacity | 5-30 (mm) |
| *WLM* | Lower soil layer water capacity | 50-100 (mm) |
| *WDM* | Deep soil layer water capacity | 15-70 (mm) |
| *IM* | Impervious area ratio | 0.01-0.02 |
| *SM* | Free water capacity | 10-50 (mm) |
| *EX* | Distribution of free water capacity coefficient | 1-2 |
| *KG* | Free water storage to groundwater outflow coefficient | 0.1-0.6 |
| *CG* | Groundwater storage recession constant | 0.98-0.998 |
| *CI* | Interflow storage recession constant | 0-0.9 |
| *CS* | Lag and route method recession constant | 0-1 |
| *L* | Lag time | 0-5 (d) |
| *XE* | Muskingum method parameter | -0.5-0.5 |

In this study, we carried on the uncertainty and parameter estimation of the Xinanjiang daily model by using the GLUE method. The Xinanjiang daily model focuses on the water balancing and the hydrograph simulation. Therefore, the objective function (*OBJ*) adopted herein is calculated as follows:

$$OBJ = |RDRE| + 1 - NSCE \tag{2}$$

where *RDRE* and *NSCE* represent the Runoff Depth Relative Error and the Nash-Sutcliffe Coefficient of Efficiency, respectively. The computation of the *NSCE* has been listed in Eq. (1) and the *RDRE* is calculated as follows:

$$RDRE = \frac{\sum_{i=1}^{n} q_{\mathrm{sim},i} - \sum_{i=1}^{n} q_{\mathrm{obs},i}}{\sum_{i=1}^{n} q_{\mathrm{obs},i}} \tag{3}$$

where $q_{\mathrm{sim},i}$ denotes simulated discharge at time step $i$; $q_{\mathrm{obs},i}$ denotes observed

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences Discussions

discharge at time step $i$; $\bar{q}_{\text{obs}}$ denotes mean of the observed discharges; $n$ denotes the number of discharge data.

The parameters and state variables of the Xinanjiang model require two additional constraints to ensure the correctness of the model physical meaning. The constraints are applied on parameters *CG*, *CI*, and *CS* (*CG* must be larger than *CI* and *CI* must be larger than *CS*) and soil moisture *W* (*W* must be non-negative). In order to consider the first constraint in the procedure of the GLUE method, before calculating the *OBJ*, we test the *CG*, *CI*, and *CS* values to verify whether the constraint relationship is satisfied. If it is satisfied, we continue the calculation of the *OBJ*; otherwise, we set the *OBJ* equal to a penalty term which is computed by

$$OBJ = lambda + lambda\left[\left|\min\left(0, CG - CI\right)\right| + \left|\min\left(0, CI - CS\right)\right|\right] \quad (4)$$

where *lambda* is a penalty coefficient which was set to 1000 in this research. If the first constraint is satisfied, then we can run the Xinanjiang daily model by using the hydro-meteorological data to generate the simulated discharge time series. After the model simulation is finished, a "flag" will be returned to indicate whether the simulation is success. If the simulation is early stopped and returned a "flag" indicating that the state variable *W* has negative values, we set the *OBJ* equal to another penalty term expressed as

$$OBJ = lambda + lambda\left(WM_{\text{UB}} - WM\right) \quad (5)$$

where $WM_{\text{UB}}$ denotes the upper boundary of parameter *WM*; *WM* denotes the *WM* value generated by the Monte Carlo sampling of the GLUE method. This penalty term forces the algorithm to search towards larger *WM* to avoid the negative *W* values.

If the above mentioned two constraints are both satisfied, we calculate the *OBJ* according to

$$OBJ = \left|RDRE\right| + 1 - NSCE \quad (6)$$

After the *OBJ* of each parameter set is calculated, we can obtain the uncertainty plot and optimal parameter set based on the *OBJ* values.

### *2.2 Parallel GLUE-based Xinanjiang model uncertainty and parameter estimation*

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences
Discussions

*2.2.1 Parallel GLUE method*

The parallelization of the GLUE method involves two steps which include the parallelization of the Monte-Carlo sampling and the optimal parameter set reduction. We implemented the parallel GLUE method on multi-core CPU computer cluster and many-core GPU computer cluster, respectively. The detailed description of the implementation can be found in the following paragraphs.

*2.2.2 Parallel uncertainty and parameter estimation - CPU computer cluster implementation*

The parallel GLUE method was implemented on a multi-core CPU computer cluster which contains 4 HP Z-series workstations host totally 5 Intel Xeon E5-2630v3 multi-core CPUs. The flow chart of the CPU computer cluster implementation of the parallel GLUE method is demonstrated in figure 2.

The parallel GLUE method starts from the initial settings on the master node. The hydro-meteorological data is loaded from CSV (comma-separated values) files which include daily rainfall, runoff discharge, evapotranspiration, and catchment geographical information. Besides, the program set the total sample number of the Monte Carlo (*NS*), likelihood threshold (*TH*), model parameter boundaries, and *KG* plus *KI* constraint. After initial data loading and settings, the algorithm queries the number of slave nodes (*NN*) and number of CPU cores in each slave node by using MPI and OpenMP, respectively. The work load quantity assigned to each slave node is calculated as follows:

$$NSS_i = NS \times NC_i / NT \quad \text{and} \quad NT = \sum_{i=1}^{NN} NC_i \tag{7}$$

where $NSS_i$ denotes number of samples generated in slave node $i$; $NC_i$ denotes number of CPU cores in slave node $i$; $NT$ denotes total number of CPU cores; $i = 1, 2, \ldots, NN$.

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences
Discussions

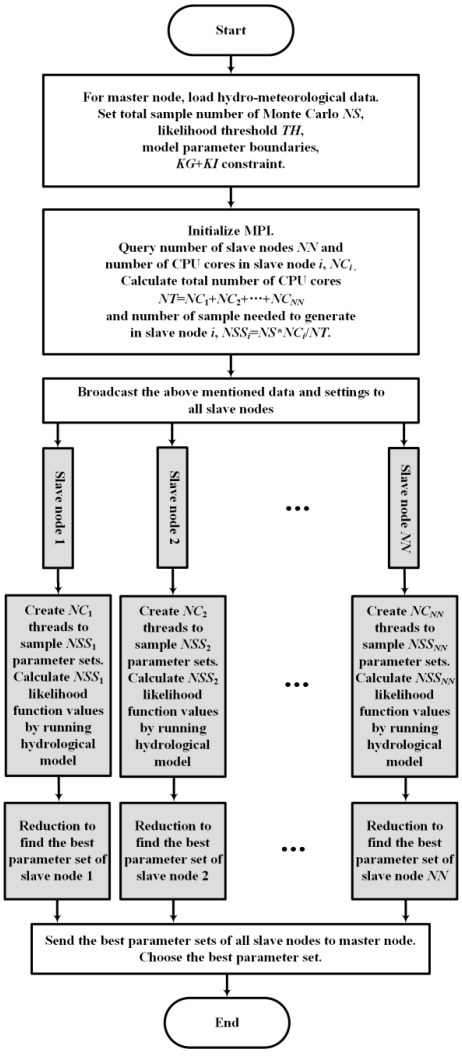


**Fig. 2.** The flow chart of the CPU computer cluster implementation of the parallel GLUE method.

After initial data loading and model settings have been finished, the above mentioned data and settings are broadcasted to all slave nodes by using MPI_Bcast API. For each slave node (let's set slave node $i$ as an example), generate $NC_i$ threads to sample $NSS_i$ parameter sets which fall in the parameter feasible space. For each thread, run the Xinanjiang hydrological model and compute the likelihood function value by using the generated parameter set. Save parameter set with likelihood value

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences
Discussions

higher than *TH*. Model simulations and likelihood function evaluations in the $NC_i$ threads are executed in parallel by using the OpenMP technology. After the $NC_i$ threads of calculations have been finished, OpenMP parallel reduction operation was started to find the optimal (largest) parameter set of each slave node. At last, send the feasible parameter sets and optimal parameter set to the master node by using MPI_Send API.

During the Monte Carlo sampling, likelihood calculations, and parallel reduction of all the slave nodes, the master node waits for all these computations' completion. Once all the above computations are completed, the master node receives feasible and best parameter sets of each slave node by using MPI_Recv API. At last, the master node generates the uncertainty plots for the feasible parameter sets and chooses the parameter set with the largest likelihood function value as the optimal parameter set and finishes the execution by using MPI_Finalize API.

*2.2.3 Parallel uncertainty and parameter estimation - GPU computer cluster implementation*

The parallel GLUE method was implemented on a many-core GPU computer cluster which is constructed by 4 HP Z-series workstations host totally 5 NVIDIA Tesla K40c many-core GPUs. The flow chart of the GPU computer cluster implementation of the parallel GLUE method is demonstrated in figure 3.

The parallel GLUE method starts from the initial settings on the master node. The hydro-meteorological data is loaded from CSV (comma-separated values) files which include daily rainfall, runoff discharge, evapotranspiration, and catchment geographical information of the study catchment. The algorithm also sets the total sample number of Monte Carlo (*NS*), likelihood threshold (*TH*), model parameter boundaries, and *KG* plus *KI* constraint. After initial data loading and settings, the algorithm begins the MPI execution and queries the number of slave nodes (*NN*), number of GPUs in each slave node, and number of GPU cores in each slave node by using MPI and CUDA, respectively. The work load quantity assigned to each slave node is calculated as follows:

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences
Discussions

$$NSS_i = NS \times \sum_{j=1}^{NG_i} NC_{i,j} \Big/ NT \quad \text{and} \quad NT = \sum_{i=1}^{NN} \sum_{j=1}^{NG_i} NC_{i,j} \tag{8}$$

where $NSS_i$ denotes number of samples generated in slave node $i$; $NC_{i,j}$ denotes number of GPU cores of GPU $j$ in slave node $i$; $NG_i$ denotes number of GPUs in slave node $i$; $NT$ denotes total number of GPU cores; $i = 1, 2, \ldots, NN$; $j = 1, 2, \ldots, NG_i$.

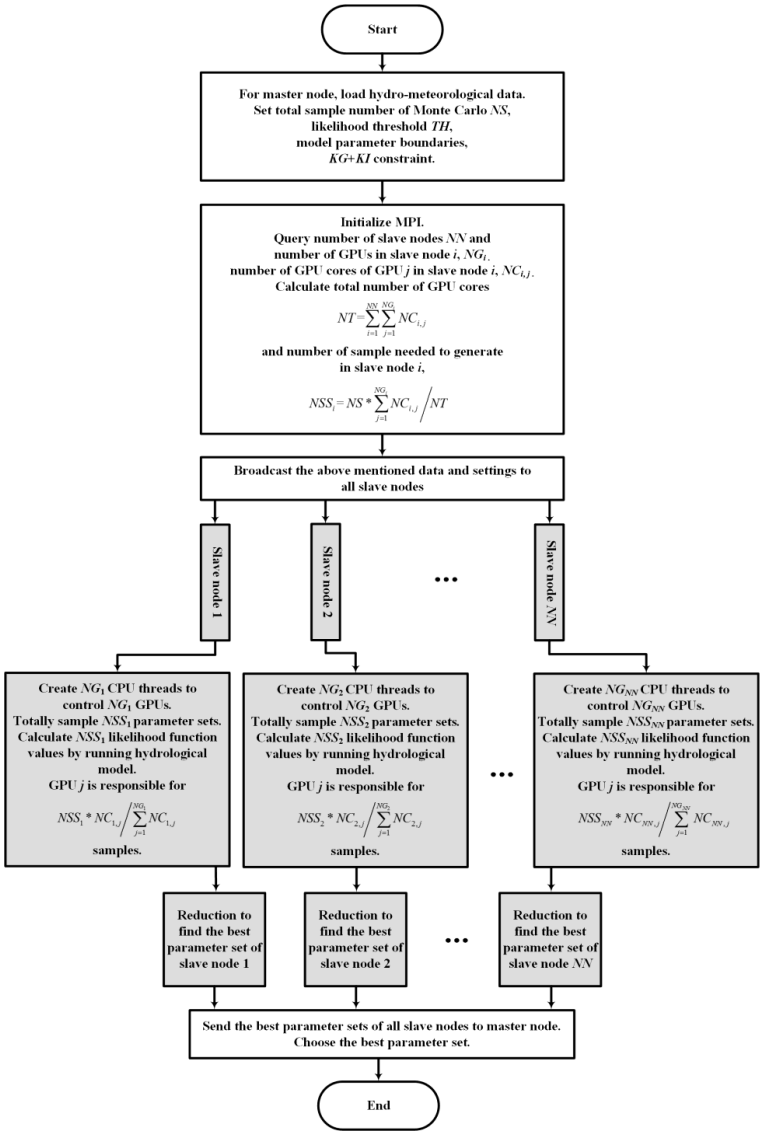


**Fig. 3.** The flow chart of the GPU computer cluster implementation of the parallel GLUE method.

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences Discussions

After initial data loading and settings have been finished, the above-mentioned data and settings are broadcasted to all slave nodes by using MPI_Bcast API. For each slave node (let's set slave node $i$ as an example), create $NG_i$ CPU threads to control $NG_i$ GPUs by using OpenMP and offload the data and settings on the GPUs by using CUDA. Totally sample $NSS_i$ parameter sets fall in the parameter feasible space on the GPUs. Each GPU thread run the Xinanjiang hydrological model and compute the likelihood function value by using the generated parameter set, parameter sets with likelihood value higher than $TH$ were saved. The model and likelihood function calculations are executed in parallel by using the OpenMP and CUDA technology on the GPUs. The GPU $j$ is responsible for parameter set generation, model run, and likelihood function evaluations of $NSS_i \times NC_{i,j} \Big/ \sum_{j=1}^{NG_i} NC_{i,j}$ samples. After the calculations of likelihood, the CUDA parallel reduction was started to find the feasible parameter sets and the best (largest) parameter set of each slave node. At last, the uncertainty plot of the feasible parameter sets and the best parameter set was send to the master node by using MPI_Send API.

During the Monte Carlo sampling, likelihood calculations, and parallel reduction of all the slave nodes, the master node waits for all these computations' completion. Once all the above computations are completed, the master node receives the feasible parameter sets and the best parameter set of each slave node by using MPI_Recv API. At last, the master node plots the uncertainty plot and chooses the parameter set with the largest likelihood function value as the best parameter set and finishes the execution by using MPI_Finalize API.

*2.3 Hardware and software adopted in this study*

The hardware utilized in this research is a HP Z-series workstation computer cluster constituted by one HP Z820 and three HP Z840 workstations. The screen and four workstations are demonstrated in figure 4. The USB KVM switch which is used for the controlling and switching of four workstations by one set of screen, keyboard, and mouse is shown in figure 5. This computer cluster has 5 Intel Xeon E5-2630v3 multi-core CPUs and 5 NVIDIA Tesla K40c many-core GPUs.

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences
Discussions

**Fig. 4.** The HP Z-series computer cluster.

**Fig. 5.** The USB KVM switch.

The Intel Xeon E5-2630v3 CPU is a high-end server level OEM/tray microprocessor. It's a Haswell-EP architecture CPU with 0.022nm manufacturing process. It has 8 CPU cores and supports hyper-threading technology with up to 16 parallel threads. The base frequency of the CPU core is 2.4GHz and the turbo frequency is 3.2GHz. The level 1 cache size is 8x32KB 8-way set associative instruction and data caches. The level 2 cache size is 8x256KB 8-way set associative caches. The level 3 cache size is 20MB shared cache. It supports many new features such as MMX instructions, SSE/streaming SIMD extensions, AVX/advanced vector extensions, and TBT2.0/turbo boost technology 2.0, etc. The V core is 0.65V-1.3V. The maximum operating temperature is 72.1℃. The minimum power dissipation is 32 Watt for C1E state and 12 Watt for C6 state.

The Tesla K40c is a high-end professional graphics card by NVIDIA, launched in October 2013. Built on the 28nm process, and based on the GK110B graphics processor, the card supports DirectX 12.0. The GK110B graphics processor is a large

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences
Discussions

chip with a die area of 561mm$^2$ and 7080 million transistors. It features 2880 shading units, 240 texture mapping units and 48 ROPs. NVIDIA has placed 12288MB GDDR5 memory on the card, which are connected using a 384-bit memory interface. The GPU is operating at a frequency of 745MHz, memory is running at 1502MHz. Being a dual-slot card, the NVIDIA Tesla K40c draws power from 1x6-pin+1x8-pin power connectors, with power draw rated at 245W maximum. Tesla K40c is connected to the rest of the system using a PCIe 3.0x16 interface. The card measures 267mm in length, and features a dual-slot cooling solution. Its price at launch was 7699 US Dollars.

The software is developed based on the Microsoft Windows 7 64-bit operating system in this research. The software ecosystem applied in this study is constituted by the MPICH2, Microsoft VC++2010 with OpenMP, and NVIDIA CUDA 6.5.

MPICH is a high-performance and widely portable implementation of the Message Passing Interface (MPI) standard (MPI-1, MPI-2, and MPI-3). The goals of MPICH are: (1) to provide an MPI implementation that efficiently supports different computation and communication platforms including commodity clusters (desktop systems, shared-memory systems, multi-core architectures), high-speed networks (10 Gigabit Ethernet, InfiniBand, Myrinet, Quadrics) and proprietary high-end computing systems (Blue Gene, Cray) and (2) to enable cutting-edge research in MPI through an easy-to-extend modular framework for other derived implementations. MPICH is distributed as source (with an open-source, freely available license). It has been tested on several platforms, including Linux (on IA32 and x86-64), Mac OS/X (PowerPC and Intel), Solaris (32- and 64-bit), and Windows.

The OpenMP API supports multi-platform shared-memory parallel programming in C/C++ and Fortran. The OpenMP API defines a portable, scalable model with a simple and flexible interface for developing parallel applications on platforms from the desktop to the supercomputer. In this research, we adopted the OpenMP in Visual C++2010 implementation to develop parallel codes. The OpenMP C and C++ application program interface lets us write applications that effectively use multiple processors. Visual C++2010 supports the OpenMP 2.0 standard which includes

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


OpenMP directives, clauses, library reference, and program interface.

CUDA is a parallel computing platform and programming model invented by NVIDIA. It enables dramatic increases in computing performance by harnessing the power of the graphics processing unit (GPU). With millions of CUDA-enabled GPUs sold to date, software developers, scientists, and researchers are using GPU-accelerated computing for broad-ranging applications. The NVIDIA CUDA Toolkit adopted in this study provides a comprehensive development environment for C and C++ developers building GPU-accelerated applications. The CUDA Toolkit includes a compiler for NVIDIA GPUs, math libraries, and tools for debugging and optimizing the performance of applications. We'll also find programming guides, user manuals, API reference, and other documentation to help us get started quickly accelerating our application with GPUs.

*2.4 Hydro-meteorological data utilized in this study*

The study area of this research is the Ba River basin. It originates from the north slope of the Qinling, China. The full length of the Ba River is 92.6km. The elevation difference from the headwater to the outlet of the river is 1142m. The total slope is 12.8%. The area of the research basin is 2577km$^2$. The Ba River basin is an asymmetric watershed. The left bank tributaries are sparse and long, while the right bank tributaries are condensed and short. The Ba River is a mountainous river. The river discharge hydrograph rises and falls steeply. The peak flow usually appears in summer season and the drying season is winter. The average annual precipitation of the studied area is 630.9mm. The average annual evaporation is 949.7mm. The average annual runoff is 493.1 million m$^3$. There are ten rainfall gauges located in this area and the outlet station is the Maduwang station. Observed daily rainfall, evaporation, and daily average discharges range from 2000 to 2010 were utilized as the calibration data set. The Thiessen polygon map of the Maduwang catchment are demonstrated in figure 6.

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences
Discussions

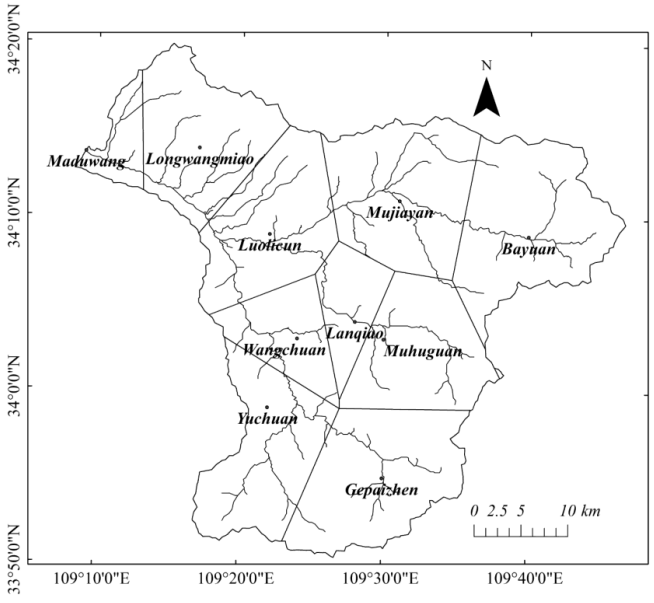


**Fig. 6.** The map of the Maduwang catchment.

## 3. Results and discussion

### *3.1 Total execution time comparison*

In order to test and examine the computational performance of the traditional serial and the improved parallel GLUE methods, we compared the total execution time of these methods. Total execution time of serial and parallel GLUE methods is demonstrated in figure 7. For the purpose of comparing total execution time corresponds to different parallelism degree, we adjusted the number of samples, which reflects the parallelism degree, from 10000 to 1000000 with a multiplier of 10. Figure 7 (A) and (B) show the total execution time of serial, CPU cluster parallel, and GPU cluster parallel GLUE methods, respectively. The serial GLUE is executed by 1 CPU with 1 core. The CPU cluster parallel GLUE is executed by 5 CPUs with 80 cores. The GPU cluster parallel GLUE is executed by 5 GPUs with 14400 cores and the GPU boost and ECC are set to off.

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences
Discussions
Open Access
EGU

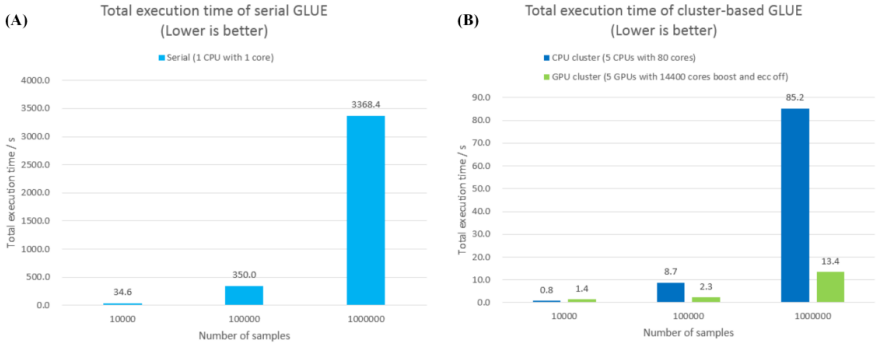


**Fig. 7.** Total execution time of serial and parallel GLUE methods; (A) serial GLUE; (B) parallel GLUE.

As we can see in figure 7, when the number of samples increases, the total execution time of the three versions of GLUE also increases. The execution time of serial, CPU cluster parallel, and GPU cluster parallel GLUE increases from 34.6s to 3368.4s, from 0.8s to 85.2s, and from 1.4s to 13.4s, respectively. For the serial GLUE, the total execution time increases approximately according to a multiplier of 10, which is the same increment multiplier of the number of samples. The CPU cluster parallel GLUE performs similar to the serial GLUE. Its total execution time also increases approximately according to the multiplier of 10. However, the GPU cluster parallel GLUE do not perform as the above two versions. When the number of samples increases from 10000 to 100000, the total execution time increases more than 10 times (from 1.4s to 2.3s). When the number of samples increases from 100000 to 1000000, the total execution time increases less than 10 times (from 2.3s to 13.4s), which indicates that with higher computational burden (larger number of samples), the GPU cluster parallel version's computational efficiency becomes better than the serial and CPU cluster parallel versions. It can be observed from figure 7 that parallel GLUE methods perform much better than their serial counterpart. The GPU cluster parallel GLUE runs very fast for large number of samples and significantly outperforms the other two GLUE versions. With larger number of samples, the parallel GLUE methods run much faster than the serial version. It can be concluded that parallel GLUE implementations can solve much larger scale parameter estimation

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences
Discussions

tasks with much faster speed than the traditional serial GLUE implementation.

*3.2 Speedup ratio comparison*

The speedup ratio of different GLUE implementations are demonstrated in figure 8. The figure 8 (A) and (B) show the speedup ratio of cluster-based parallel GLUE v.s. serial GLUE and GPU cluster parallel GLUE v.s. CPU cluster parallel GLUE, respectively. The hardware utilized here is same as above mentioned, i.e. 5 CPUs with 80 cores and 5 GPUs with 14400 cores with GPU boost and ECC off.

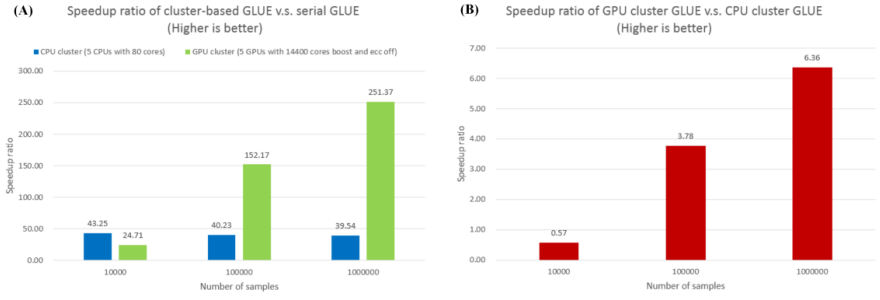


**Fig. 8.** Speedup ratio of different GLUE implementations; (A) parallel v.s. serial GLUE; (B) GPU cluster v.s. CPU cluster GLUE.

It can be seen in figure 8 (A) that with the increase of number of samples, the speedup ratio of the CPU cluster parallel GLUE implementation decreases a bit from 43.25x to 39.54x. On the contrary, with the increase of number of samples, the speedup ratio of the GPU cluster parallel GLUE implementation increases significantly from 24.71x to 251.37x. Under the condition of relatively small number of samples such as 10000, the speedup ratio of GPU cluster (24.71x) is a bit smaller than the CPU cluster (43.25x). While when the number of samples increases, the GPU cluster implementation (152.17x and 251.37x) outperforms the CPU cluster implementation (40.23x and 39.54x) significantly. It can be inferred that the GPU cluster parallel implementation performs much better than the CPU cluster parallel version, especially for large scale hydrological model parameter estimation tasks.

The above paragraph assesses the speedup ratio between parallel GLUE implementations and serial GLUE implementation. This section compares the performances of the GPU clusters parallel GLUE and CPU cluster parallel GLUE

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


implementations. We can see in figure 8 (B) that with the increase of the number of samples, the GPU cluster version runs gradually faster than the CPU cluster version. For relatively small scale problems such as number of samples equals to 10000, the speedup ratio is less than 1x (0.57x), which means the GPU implementation is slower than the CPU implementation. Nevertheless, when faces with the large scale parameter estimation problems such as the number of samples equals to 100000 or 1000000, the GPU cluster parallel implementation run 3.78x or 6.36x faster than the CPU cluster parallel implementation. These facts indicate that the GPU cluster parallel GLUE method is more suitable to large scale model parameter estimation problems and runs much faster than the CPU cluster parallel GLUE method for these kinds of tasks.

### *3.3 Scalability analysis*

#### *3.3.1 Analysis based on total execution time*

In this section, the scalability of the parallel GLUE methods is analyzed based on the total execution time. Here we focused on the scalability analysis, therefore, we fixed the number of samples to 1000000. The GPU cluster turns off the GPU boost and ECC. We varied the number of CPUs or GPUs and tested the total execution time to compare the performances of parallel GLUE methods. The total execution time of CPU cluster parallel GLUE and GPU cluster parallel GLUE is demonstrated in figure 9. It can be inferred from the figure that the total execution time of the parallel GLUE methods decreases when applying more CPUs or GPUs to accelerate the computation. With more CPUs or GPUs, the computational efficiency of the parallel GLUE methods improves significantly. These testing results indicate that the scalability of the parallel GLUE method is good and the parallel method can be applied to highly parallel and heavy computational burden tasks.

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences
Discussions

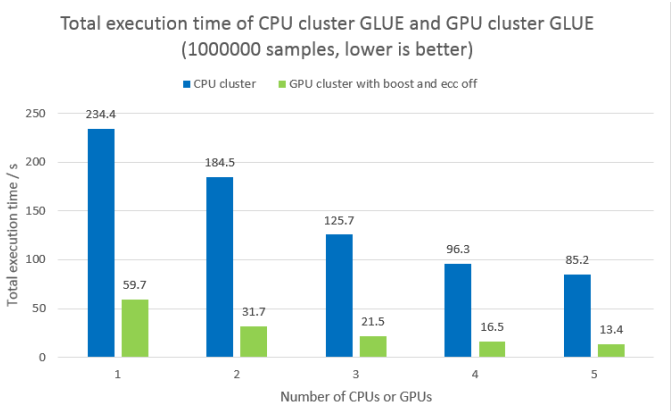


**Fig. 9.** Total execution time of CPU cluster parallel GLUE and GPU cluster parallel GLUE.

*3.3.2 Analysis based on speedup ratio of CPU cluster GLUE and GPU cluster GLUE v.s. serial GLUE*

In this section, the scalability of the parallel GLUE methods is analyzed based on the speedup ratio between parallel GLUE methods and serial GLUE method. Here we focused on the scalability analysis and therefore set the number of samples to 1000000. The GPU cluster turns off the GPU boost and ECC. We varied the number of CPUs or GPUs and tested the speedup ratio between parallel GLUE methods and serial GLUE method to compare the performances of parallel GLUE methods. The speedup ratio of CPU cluster parallel GLUE and GPU cluster parallel GLUE v.s. serial GLUE is demonstrated in figure 10. We can see from figure 10 that the speedup ratios of the parallel GLUE methods increase with the increment of the number of CPUs or GPUs adopted to execute the parallel computation. The GPU cluster parallel GLUE outperforms the CPU cluster parallel GLUE significantly. With all the 5 GPUs run in parallel, the GPU cluster parallel GLUE achieved 251.37x speedup ratio compared with the CPU cluster parallel version's 39.54x speedup ratio. These facts showed that even the CPU cluster parallel GLUE can run much faster than the serial CPU version, the GPU cluster parallel GLUE achieved even much better computational efficiency than the CPU cluster parallel GLUE.

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences
Discussions

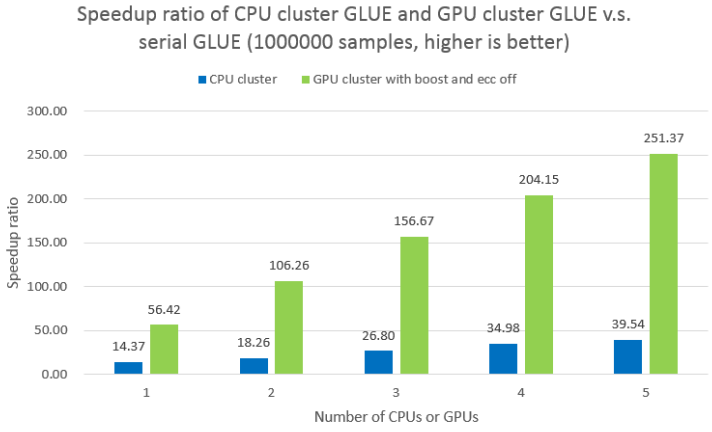


**Fig. 10.** Speedup ratio of CPU cluster parallel GLUE and GPU cluster parallel GLUE v.s. serial GLUE.

*3.3.3 Analysis based on speedup ratio of GPU cluster GLUE v.s. CPU cluster GLUE*

In this section, the scalability of the parallel GLUE methods is analyzed based on the speedup ratio between parallel GLUE methods. Here we focused on the scalability analysis by setting the number of samples as 1000000. The GPU cluster turns off the GPU boost and ECC. We varied the number of CPUs or GPUs and tested the speedup ratio between parallel GLUE methods to compare the performances of parallel GLUE methods. The speedup ratio of GPU cluster parallel GLUE v.s. CPU cluster parallel GLUE is demonstrated in figure 11. It can be observed in figure 11 that the GPU cluster parallel GLUE method run faster than the CPU cluster parallel GLUE when using 1 to 5 CPUs or GPUs. The GPU version runs 3.93x to 6.36x faster than the CPU version which indicates that the GPU cluster parallel GLUE has better scalability than the CPU version. Therefore, adopting more GPUs or CPUs can achieve better computational performance than only using small number of CPUs or GPUs.

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021




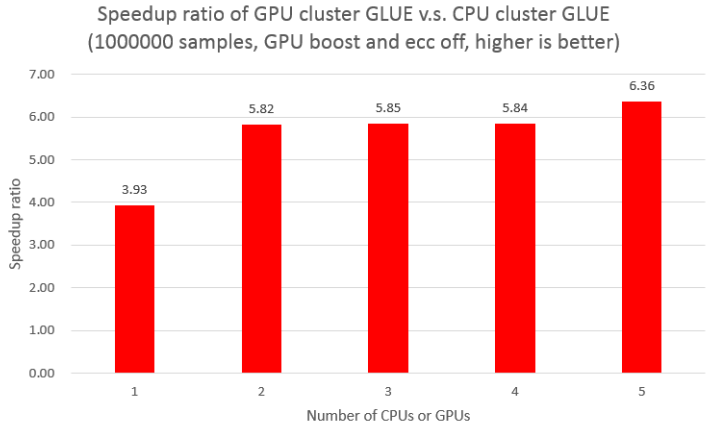


**Fig. 11.** Speedup ratio of GPU cluster parallel GLUE v.s. CPU cluster parallel GLUE.

## 4. Conclusions

In this research, we developed an improved version of GLUE method to accelerate its computational efficiency. The multi-core CPU and many-core GPU hybrid high performance computing hardware system and MPI, OpenMP, and CUDA-based hybrid software development ecosystems were developed. We applied the improved parallel and traditional serial GLUE method to the parameter estimation of a famous hydrological Xinanjiang model real world application. The following conclusions can be drawn here.

(1) The parallel GLUE-Xinanjiang methods run much faster than their serial implementation counterpart and achieved much smaller computational time and much higher speedup ratios. The computational efficiency comparison results indicated the satisfying performance of the parallel methods. Furthermore, the GPU cluster parallel GLUE method achieved the highest speedup ratio up to 251.37x which significantly improves the computational efficiency and makes the highly intensive computation Monte Carlo based GLUE method possible and meaningful to be applied in real word and real time applications.

(2) The parallel GLUE methods have good and satisfactory scalability which allows scientists and engineers to apply more CPU or GPU devices to further improve the computational efficiency and makes them capable to tack the large-scale

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


computation tasks. It is possible to solve problems previously not solvable due to too much memory and computation power requirements. These kinds of parameter estimation problems often come up with the requirements of national or even global hydrological or meteorological modelling and simulation tasks. With the development of the cluster based super computers such as the Summit and Sierra of the TOP500 super computers, the CPU-GPU hybrid high performance computer cluster-based parallel GLUE method can achieve even better performance and real-world application results in the near future.

**Conflict of interest**

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

**Acknowledgements**

This research was funded by Open Fund of Beijing Key Laboratory of Urban Water Cycle and Sponge City Technology (HYD2020OF02); National Key R&D Program of China (2019YFC1510605); IWHR Research & Development Support Program (JZ0199A032021); GHFUND A No.20210701; IWHR-CDR Scientific and Technological Innovation Talent Fund (Basic Research Type); Research and Development Program of Zhongnan Engineering Corporation Limited, Development of Distributed Hydrological Model. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Tesla K40 and TITAN V GPUs used for this research.

**References**

Beven K, Binley A, 1992. Future of distributed models: Model calibration and uncertainty prediction. Hydrological Processes, 6 (3), 279-298.

Beven K, Leedal D, Hunter N, Lamb R, 2012. Communicating uncertainty in flood risk mapping, in Prceedings, FloodRisk2012.

Beven K, Binley A, 2014. GLUE: 20 years on. Hydrological Processes, 28, 5897-5918.

Binley A, Beven K, 1991. Physically based modelling of catchment hydrology: a

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


likelihood approach to reducing predictive uncertainty. In Computer Modelling in the Environmental Sciences, Farmer DG, Rycroft MJ (eds). Clarendon Press: Oxford; 75-88.

Blasone R, Vrugt J, 2008. Generalized likelihood uncertainty estimation (GLUE) using adaptive Markov Chain Monte Carlo sampling. Advances in Water Resources, 31, 630-648.

Deng Y, Zheng B, Su Y, et al., 2008. Impact of likelihood function to GLUE method. Research of Environmental Sciences, 21 (2), 44-48.

Dong J, Zheng C, Kan G, Wen J, Zhao M, Yu J, 2015. Applying the ensemble artificial neural network-based hybrid data-driven model to daily total load forecasting. Neural Computing & Applications, 26 (3), 603-611.

Dong, Jianzhi; Crow, Wade T.; Duan, Zheng; (2019). A double instrumental variable method for geophysical product error estimation, REMOTE SENSING OF ENVIRONMENT, 225, 217-228

Duan, Z., Chen, Q.*, Chen, C., Liu, J., Gao, H., Song, X. & M. Wei (2018) Spatiotemporalanalysis of nonlinear trends in precipitation over Germany during 1951‐2013 from multiple observation‐based gridded products. International Journal of Climatology, doi: 10.1002/joc.5939

Duan, Z., Tuo, Y., Liu, J.*, Gao, H.*, Song, X., Zhang, Z., Yang, L. & D. F. Mekonnen (2019)Hydrological evaluation of open-access precipitation and air temperature datasets using SWAT in a poorly gauged basin in Ethiopia. Journal of Hydrology.569, 612-626, doi: 10.1016/j.jhydrol.2018.12.026

Gao, H., Li, H.*, Duan, Z., Ren, Z., Meng, X. & Pan, X. (2018). Modelling glacier variation and its impact on water resource in the Urumqi No1 Glacier in Central Asia. Science of The Total Environment. 644, 1160–1170, DOI: 10.1016/j.scitotenv.2018.07.004

Gao, H.*, Birkel, C., Hrachowitz, M., Tetzlaff, D., Soulsby, C. & H.H.G. Savenije. (2018) A simple topography-driven and calibration-free runoff generation model. Hydrology and Earth System Sciences 23(2), 787-809

Fang M, Zhang W, Fang J, Zhou H, Gao C, 2016. GPU Programming and Code

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Optimization High Performance Computing for the Masses. Beijing: Tsinghua University Press.

Huang G, Xie H, 2007. Uncertainty analysis of watershed hydrological model based on GLUE method. Journal of South China University of Technology (Natural Science Edition), 35 (3), 137-149.

Iorgulescu I, Beven K, Musy A, 2005. Data-based modelling of runoff and chemical tracer concentrations in the Haute-Menthue (Switzerland) research catchment. Hydrological Processes, 19, 2557-2574.

Iorgulescu I, Beven K, Musy A, 2007. Flow, mixing, and displacement in using a data-based hydrochemical model to predict conservative tracer data. Water Resources Research, 43, W03401, DOI:10.1029/2005WR004019.

Kan Guangyuan, Zhang Mengjie, Liang Ke, Wang Hao, Jiang Yunzhong, Li Jiren, Ding Liuqian, He Xiaoyan, Hong Yang, Zuo Depeng, Bao Zhenxin, Li Chaochao. Improving water quantity simulation & forecasting to solve the energy-water-food nexus issue by using heterogeneous computing accelerated global optimization method. Applied Energy, 210, 420-433, JAN 15 2018.

Kan Guangyuan, He Xiaoyan, Li Jiren, Ding Liuqian, Hong Yang, Zhang Hongbin, Liang Ke, Zhang Mengjie. Computer Aided Numerical Methods for Hydrological Model Calibration: An Overview and Recent Development. Archives of Computational Methods in Engineering, 1-25, April 25, 2017.

Kan Guangyuan, Li Jiren, Zhang Xingnan, Ding Liuqian, He Xiaoyan, Liang Ke, Jiang Xiaoming, Ren Minglei, Li Hui, Wang Fan, Zhang Zhongbo, Hu Youbing. A new hybrid data-driven model for event-based rainfall-runoff simulation. Neural Computing & Applications, 28 (9), SI:2519-2534, SEP 2017.

Kan Guangyuan, He Xiaoyan, Li Jiren, Ding Liuqian, Zhang Dawei, Lei Tianjie, Hong Yang, Liang Ke, Zuo Depeng, Bao Zhenxin, Zhang Mengjie. A novel hybrid data-driven model for multi-input single-output system simulation. Neural Computing & Applications, 29 (7), SI:577-593, APR 2018.

Kan Guangyuan, Lei Tianjie, Liang Ke, Li Jiren, Ding Liuqian, He Xiaoyan, Yu Haijun, Zhang Dawei, Zuo Depeng, Bao Zhenxin, Amo-Boateng Mark, Hu

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences
Discussions

Youbing, Zhang Mengjie. A multi-core CPU and many-core GPU based fast parallel shuffled complex evolution global optimization approach. IEEE Transactions on Parallel and Distributed Systems, 28 (2), 332-344, FEB 2017.

Kan Guangyuan, He Xiaoyan, Ding Liuqian, Li Jiren, Hong Yang, Liang Ke. Heterogeneous parallel computing accelerated generalized likelihood uncertainty estimation (GLUE) method for fast hydrological model uncertainty analysis purpose. Engineering with Computers, 2018-12.

Kan Guangyuan, Yao Cheng, Li Qiaoling, Li Zhijia, Yu Zhongbo, Liu Zhiyu, Ding Liuqian, He Xiaoyan, Liang Ke. Improving event-based rainfall-runoff simulation using an ensemble artificial neural network based hybrid data-driven model. Stochastic Environmental Research and Risk Assessment, 29 (5), 1345-1370, JUL 2015.

Kan Guangyuan, He Xiaoyan, Ding Liuqian, Li Jiren, Liang Ke, Hong Yang. Study on applicability of conceptual hydrological models for flood forecasting in humid, semi-humid semi-arid and arid basins in China. Water, 9 (10), 719, OCT 2017.

Kan Guangyuan, Tang Guoqiang, Yang Yuan, Hong Yang, Li Jiren, Ding Liuqian, He Xiaoyan, Liang Ke, He Lian, Li Zhansheng, Hu Youbing, Cui Yaokui. An improved coupled routing and excess storage (CREST) distributed hydrological model and its verification in Ganjiang river basin, China. Water, 9 (11), 904, NOV 2017.

Kan Guangyuan, He Xiaoyan, Ding Liuqian, Li Jiren, Hong Yang, Zuo Depeng, Ren Minglei, Lei Tianjie, Liang Ke. Fast hydrological model calibration based on the heterogeneous parallel computing accelerated shuffled complex evolution method. Engineering Optimization. 50 (1), 106-449, 2018.

Kan Guangyuan, He Xiaoyan, Ding Liuqian, Li Jiren, Liang Ke, Hong Yang. A heterogeneous computing accelerated SCE-UA global optimization method using OpenMP, CUDA and OpenACC. Water Science and Technology, 76 (7), 1640-1651, OCT 2017.

Kan Guangyuan, Liang Ke, Li Jiren, Ding Liuqian, He Xiaoyan, Hu Youbing, Amo-Boateng Mark. Accelerating the SCE-UA Global Optimization Method

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences
Discussions

Based on Multi-Core CPU and Many-Core GPU. Advances in Meteorology, 8483728, 1-10, 2016.

Kan Guangyuan, He Xiaoyan, Ding Liuqian, Li Jiren, Hong Yang, Ren Minglei, Lei Tianjie, Liang Ke, Zuo Depeng, Huang Pengnian. Daily streamflow simulation based on improved machine learning method. Tecnologia y Ciencias del Agua, 8 (2), 51-60, 2017-04.

Lei T, Pang Z, Wang X, Li L, Fu J, Kan G, Zhang X, Ding L, Li J, Huang S, Shao C, 2016. Drought and carbon cycling of grassland ecosystems under global change: a review. Water, 2016, 8, 460, doi:10.3390/w8100460.

Li C, Cheng X, Li N, Du X, Yu Q, Kan G, 2016. A framework for flood risk analysis and benefit assessment of flood control measures in urban areas. International Journal of Environmental Research and Public Health, 13, 787, doi:10.3390/ijerph13080787.

Li S, Liang Z, 2006. Application research of the Xinanjiang model parameter uncertainty based on GLUE. Water Resources & Hydropower of Northeast China, 24 (259), 31-33.

Li X, 2005. Research on hydrological model parameter optimization and uncertainty analysis method. Dalian: Dalian Institute of Technology.

Li Z, Kan G, Yao C, Liu Z, Li Q, Yu S, 2014. An improved neural network model and its application in hydrological simulation. Journal of Hydrologic Engineering, 19 (10), 04014019-1 – 04014019-17.

Liu Y, 2008. Study on uncertainty of runoff forecasting model and risk analysis of reservoir flood control. Dalian: Dalian Institute of Technology.

Liu Y, Liang G, Zhou H, 2009. Multiple criteria likelihood GLUE method for hydrological model uncertainty analysis. Journal of Sichuan University (Engineering Science Edition), 41 (4), 89-96.

Lin K, Chen X, Jiang T, 2009. Hydrological model parameter uncertainty analysis based on Copula-Glue. Journal of Sun Yatsen University (Natural Science Edition), 48 (3), 109-115.

Lindensehmidt K, Fleischbein K, Baborowski, 2007. Structural uncertainty in a river

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


Natural Hazards and Earth System Sciences Discussions

water quality modeling system. Ecological Modelling, 204 (3-4), 289-300.

Muleta M, Nicklow J, 2005. Sensitivity and uncertainty analysis coupled with automatic calibration for a distributed watershed model. Journal of Hydrology, 306, 127-145.

Rosenblueth E, 1975. Point estimates for probability moments. Proceedings of the National Academy of Sciences, 72 (10), 3812-3814.

Shao, W., Bogaard, T. A., Bakker, M., and Berti, M.: Analysing the influence of preferential flow on pressure propagation and landslide triggering of Rocca Pitigliana landslide. Journal of Hydrology, 2016, 543, 360-372

Shao, W., Coenders-Gerrits, M., Judge, J., Zeng, Y. J., Su, Y. (2018a), The impact of non-isothermal soil moisture transport on evaporation fluxes in a maize cropland. Journal of Hydrology, 561, 833–847

Shao, W., Yang, Z.J., Ni, J. J., Su, Y., Nie W., Ma, X.Y. (2018b), Comparison of single- and dual-permeability models in simulating the unsaturated hydro-mechanical behaviour in a rainfall-triggered landslide. Landslides, 15(12), 2449-2464

Shu C, Liu S, Mo X, et al., 2008. Uncertainty analysis of the Xinanjiang model parameter. Geographical Research, 27 (2), 343-352.

Singh V, 1995. Computer Models of Watershed Hydrology. Water Resources Publications, P. O. Box 260026, Highlands Ranch, Colorado 80126-0026, USA.

Yu, W., Li, Y., Cao, Y., Schillerberg, T., 2019. Drought assessment using GRACE terrestrial water storage deficit in Mongolia from 2002 to 2017. Water. 11(6), 1301; https://doi.org/10.3390/w11061301

Wang S., Zuo H., Yin Y., Hu C., Yin J., Ma X., Wang J. (2018): Interpreting rainfall anomalies using rainfall's non-negative nature. Geophysical Research Letters, 46(1): 426-43

Wei X, Xiong L, 2008. Improved GLUE method and its application in hydrological model uncertainty research. Water Conservancy and Hydropower Express, 29 (6), 23-25.

Xu Z, Li Z, 2009. Runoff simulation and model uncertainty analysis of Hei River

https://doi.org/10.5194/nhess-2021-344
Preprint. Discussion started: 17 November 2021


headwaters. Proceedings of the 2009 Annual Conference of Water Resources Specialized Committee. Dalian: Dalian Institute of Technology, 148-155.

Zhao R, Zhuang Y, Fang L, Liu X, Zhuang Q, 1980. The Xinanjiang Model, Hydrological Forecasting Proceedings Oxford Symposium, IAHS 129, 351-356.

Zhao R, 1983. Watershed hydrological model-Xinanjiang model and Northern Shaanxi model. Beijing: Water Resources and Electric Power Press.

Zhao R, 1992. The Xinanjiang model applied in China. Journal of Hydrology, 135, 371-381.

Zhao R, 1993. A non-linear system model for basin concentration. Journal of Hydrology, 142, 477-482.

Zhao R, 1994. Anthology of Hydrological Forecasting. Beijing: Water Resources and Electric Power Press.

Zuo D, Cai S, Xu Z, Li F, Sun W, Yang X, Kan G, Liu P, 2016. Spatiotemporal patterns of drought at various time scales in Shandong Province of Eastern China. Theor. Appl. Climatol., doi:10.1007/s00704-016-1969-5.

Zhu, Y. , Liu, Y., Ma, X., Ren, L., and Singh, V. P.,(2018) Drought analysis in the Yellow River basin based on a short-scalar Palmer drought severity index. Water, 10, 1526. doi: 10.3390/w10111526.