JAMES | Journal of Advances in Modeling Earth Systems

RESEARCH ARTICLE
10.1029/2022MS003507

Accelerating the Lagrangian Particle Tracking in Hydrologic Modeling to Continental-Scale

Chen Yang[1,2], Carl Ponder[3], Bei Wang[4], Hoang Tran[5], Jun Zhang[6], Jackson Swilley[1], Laura Condon[7], and Reed Maxwell[1,2,8]

[1]Department of Civil and Environmental Engineering, Princeton University, Princeton, NJ, USA, [2]Integrated GroundWater Modeling Center, Princeton University, Princeton, NJ, USA, [3]NVIDIA Developer Technology, Austin, TX, USA, [4]Research Computing, Princeton University, Princeton, NJ, USA, [5]Atmospheric Science & Global Change Division, Pacific Northwest National Laboratory, Richland, WA, USA, [6]Key Laboratory of VGE of Ministry of Education, Nanjing Normal University, Nanjing, China, [7]Department of Hydrology and Atmospheric Sciences, The University of Arizona, Tucson, AZ, USA, [8]High Meadows Environmental Institute, Princeton University, Princeton, NJ, USA

Key Points:

- Numerical models for large-scale water age/quality simulations are absent in communities of hydrology and Earth Surface Processes
- A parallel framework for accelerating Lagrangian particle tracking to continental-scale on distributed, multi-Graphics Processing Unit platforms is established
- The parallelized particle tracking model, EcoSLIM, is a promising tool to accelerate our understanding of the terrestrial water cycle

Supporting Information:

Supporting Information may be found in the online version of this article.

Correspondence to:

C. Yang, L. Condon and R. Maxwell,
cy15@princeton.edu;
lecondon@arizona.edu;
reedmaxwell@princeton.edu

Citation:

Yang, C., Ponder, C., Wang, B., Tran, H., Zhang, J., Swilley, J., et al. (2023). Accelerating the Lagrangian particle tracking in hydrologic modeling to continental-scale. *Journal of Advances in Modeling Earth Systems*, *15*, e2022MS003507. https://doi.org/10.1029/2022MS003507

Received 18 NOV 2022
Accepted 25 APR 2023

Abstract Unprecedented climate change and anthropogenic activities have induced increasing ecohydrological problems, motivating the development of large-scale hydrologic modeling for solutions. Water age/quality is as important as water quantity for understanding the terrestrial water cycle. However, scientific progress in tracking water parcels at large-scale with high spatiotemporal resolutions is far behind that in simulating water balance/quantity owing to the lack of powerful modeling tools. EcoSLIM is a particle tracking model working with ParFlow-CLM that couples integrated surface-subsurface hydrology with land surface processes. Here, we demonstrate a parallel framework on distributed, multi-Graphics Processing Unit platforms with Compute Unified Device Architecture-Aware Message Passing Interface for accelerating EcoSLIM to continental-scale. In tests from catchment-, to regional-, and then to continental-scale using 25-million to 1.6-billion particles, EcoSLIM shows significant speedup and excellent parallel performance. The parallel framework is portable to atmospheric and oceanic particle tracking models, where the parallelization is inadequate, and a standard parallel framework is also absent. The parallelized EcoSLIM is a promising tool to accelerate our understanding of the terrestrial water cycle and the upscaling of subsurface hydrology to Earth System Models.

Plain Language Summary Studies of water ages at multiple spatiotemporal scales are urgent to better understand the connections between different hydrologic compartments. Climate change and anthropogenic activities make this requirement more pressing. Lagrangian particle tracking is a powerful tool to simulate water ages. However, it is computationally demanding, which hampers its wide application. In this study, we provide a Lagrangian particle tracking model, EcoSLIM, with a novel parallel framework that enables it to handle large-scale water age simulations with high spatiotemporal resolutions. We combined the efforts of engineers and scientists from multiple disciplines on this work which cannot be achieved by the knowledge of an individual discipline. To the best of our knowledge, such a modeling tool is absent in communities of hydrology and Earth Surface Processes. In tests from catchment-, to regional-, and then to continental-scale using 25-million to 1.6-billion particles, EcoSLIM shows significant speedup and excellent parallel performance. Although we take EcoSLIM as an example here, the parallel framework is portable to other particle tracking models in Earth System Science, such as those in atmospheric and oceanic disciplines. The parallelized EcoSLIM is a promising tool to hydrologic community and Earth System Model developers for scientific exploration.

1. Introduction

Increasing evidence shows that groundwater regulates water and energy fluxes in the land-atmosphere system and thus is critical in Earth System Modeling (Kollet & Maxwell, 2008; Martínez-de la Torre & Miguez-Macho, 2019; Maxwell et al., 2015; Rahman et al., 2015; C. Yang et al., 2020). Large-scale groundwater modeling configured with lateral groundwater flow has been developing since a decade ago (de Graaf & Stahl, 2022; Fan et al., 2013; Keune et al., 2016; Maxwell & Condon, 2016; Xie et al., 2018), demonstrating the deficiency of one-dimensional free-drainage hydrology in Earth System Models (ESMs) (Fan et al., 2019). However, large-scale groundwater modeling mainly focused on water quantity (van Vliet et al., 2017), and few studies were conducted on water

quality/ages (Hartmann et al., 2021; Luijendijk et al., 2020). Recent studies highlighted that the terrestrial water cycle may have a period much longer than 1 year if we further identify the pathways of water in the annual water balance (Benettin et al., 2021; McDonnell, 2017), which is attributed to the contribution of groundwater to the Earth's surface processes. For example, streamflow or evapotranspiration (ET) may be tens of years old with long-tailed age distributions (Kirchner et al., 2000; Kuppel et al., 2020; Sprenger et al., 2019). Hence, to fully understand the subsurface hydrologic processes and reasonably upscale them to the scales and resolutions of ESMs, it is critical to portray the flow paths of groundwater and their connections with the land-atmosphere system (Fan, 2016; Fan et al., 2019; McDonnell & Beven, 2014).

With the scientific motivation summarized above, hydrologic models tracking water ages and/or flow paths in the land-subsurface coupled framework have emerged in recent years. EcH_2O-iso is a distributed ecohydrological model using the Eulerian method (Kuppel et al., 2018). It tracks water isotopes and ages based on a land surface model of a simplified groundwater treatment (Maneta & Silverman, 2013). Recently, EcH_2O-iso has growing applications in small catchments (Smith et al., 2021; X. Yang, Tetzlaff, et al., 2021). PARTRACE is a Lagrangian particle tracking code based on the three-dimensional variably saturated subsurface flow (Bechtold et al., 2011), which has been used in subsurface-vegetation coupled hydrologic processes for ideal or small modeling domains (Cremer et al., 2016; Schroder et al., 2012; J. Yang et al., 2018). EcoSLIM is also a Lagrangian particle tracking model that works with a three-dimensional variably saturated subsurface flow model configured with the land surface processes (Maxwell et al., 2019). Current applications of EcoSLIM are also limited to small catchments (Rapp et al., 2020; Wilusz et al., 2019). But Fan (2015) pointed out the potential hydraulic connections from the Appalachian ridges to the sea level driven by regional groundwater gradients, which act like giant hillslopes. Wörman et al. (2007) proposed continent-scale groundwater flow paths using spectral and numerical models for shallow and deep systems of simplified conceptualizations. Hence, large-scale water parcel tracking is needed to understand these regional to continental groundwater flow systems and thus to reasonably configure them in ESMs. With intensified climate change and human activities, this demand has become more pressing to understand the terrestrial water cycle in the changing world, which is one of the 23 unsolved hydrologic questions (Bloschl et al., 2019). However, we face an incredible computational burden to expand the above listed models to the scale of ESMs (Kollet et al., 2010).

Hydrologic models based on Eulerian approaches have been widely parallelized on distributed platforms facilitating their applications across scales (Hammond et al., 2014; Kollet et al., 2010), whereas the parallelization of Lagrangian approaches, particularly that on distributed platforms, is still lacking, preventing their extension to large scales. Parallelization on one computing node using OpenMP of shared memory was explored in our previous work on EcoSLIM (Maxwell et al., 2019; C. Yang, Zhang, et al., 2021) and in Ji et al. (2019) on MODPATH (Pollock, 2016). PARTRACE has been performed on distributed platforms by dividing the simulated particles into equal portions (Englert et al., 2003). Each processor calculates one portion but stores the entire flow field. This is similar to a previous parallelization of EcoSLIM (C. Yang et al., 2022) and that of MODPATH by Ji et al. (2019). This approach is fast because it avoids communication between different processors for particle exchange. However, the memory requirement to save the entire velocity field for the entire modeling domain prevents the application of these models to large domains. Engdahl et al. (2019) proposed KD tree and domain decomposition (DDC) to accelerate and parallelize particle tracking with mass transfer. Schauer et al. (2022) improved Engdahl et al. (2019)'s concept through small ideal cases with pure diffusion and without advection and reactions. Atmospheric and oceanic particle tracking models for pathways of gases, water, and other tracers have a theoretical basis similar to that in hydrology. Most of them are offline models that calculate the trajectories of particles based on given fields, such as velocities generated from general circulation models (GCMs) (Döös et al., 2013). However, parallelization of these models falls behind their functionalities (van Sebille et al., 2018). For example, TRACMASS (Doos et al., 2017) has not been parallelized to the best of our knowledge; FLEXPART (Pisso et al., 2019) has been parallelized using the single OpenMP or Message Passing Interface (MPI) by dividing the simulations into several small runs for different simulation periods or particle sources, which has the same memory bottleneck mentioned above (Englert et al., 2003; Ji et al., 2019; C. Yang et al., 2022). Obviously, a standard parallelization framework on distributed platforms for Lagrangian particle tracking, which can thoroughly break the memory bottleneck towards massively parallel computing, is absent in Earth System Science.

Additionally, hydrologic particle tracking based on transient flow fields with land-subsurface coupled processes at large scales is a complex modeling system. Simulations of groundwater age require a large number of particles to fill the entire subsurface to ensure a high spatial resolution of the age. This differs from tracking specific

pollutants or water parcels of limited volumes, which can be conceptualized using a small number of particles. Simulations of groundwater age also require a long simulation time to evolve the groundwater flow system, thus removing the effects of initial conditions on the age distribution. More importantly, coupling with land-surface processes, such as ET, requires small timesteps, which further increases the computational burden. Once the potential uncertainties at large scales caused by variable climatic conditions, subsurface heterogeneities, and topographies are considered, the parallelization of the land-subsurface Lagrangian system becomes more challenging. Finally but importantly, among the listed parallel studies above, Ji et al. (2019) and our previous work (C. Yang, Zhang, et al., 2021; C. Yang et al., 2022) leveraged the heterogeneous parallel architecture of multi-GPU (Graphics Processing Unit) with OpenMP or MPI. GPU acceleration is growing in hydrologic models, including some particle tracking models (Hokkanen et al., 2021; Ji et al., 2014; Morales-Hernandez et al., 2021; Rizzo et al., 2019; Wang et al., 2022), and in GCMs/ESMs (Fuhrer et al., 2018; Leutwyler et al., 2016) in recent years. This brings new opportunities to pursue faster speed of particle tracking models in Earth System Science but also induces more technical requirements to build an efficient parallel framework handling multi-GPU.

Motivated by this background, we provide the EcoSLIM model with a parallel framework on distributed, multi-GPU platforms that can handle the Lagrangian particle tracking at a continental-scale with high spatiotemporal resolutions. The modeling domain is decomposed into small subdomains to remove the memory limitation analyzed above. Particles are transferred among subdomains via CUDA-Aware (Compute Unified Device Architecture) MPI once they move out of the subdomains. Two Load balancing (LB) schemes are included to fully utilize the computational resources, to further speed the code, and, more importantly, to ensure the immediate application of the model to real world.

2. Overview of EcoSLIM

2.1. Mechanistic Processes

The position of a particle in subsurface is described as for example, (Tompson & Dougherty, 1988; Tompson & Gelhar, 1990):

$$\mathbf{S}(t+\Delta t)-\mathbf{S}(t)=\mathbf{A}\Delta t+\mathbf{B}\cdot\mathbf{Z}\sqrt{\Delta t} \tag{1}$$

$$\mathbf{A}\equiv\mathbf{v}+\nabla\cdot\mathbf{D}+\frac{\mathbf{D}}{n\theta}\cdot\nabla(n\theta) \tag{2}$$

$$\mathbf{B}\cdot\mathbf{B}^{T}\equiv 2\mathbf{D} \tag{3}$$

where $\mathbf{S}(t)$ [L] is the position vector of the particle at time t [T], Δt [T] is the local timestep of the particle, $\mathbf{Z}$[–] is a vector of pseudorandom numbers, $\mathbf{v}$ [L/T] is the velocity vector at time t [T], $\mathbf{D}$ [L^2/T] is the hydrodynamic dispersion tensor, n[−] is the porosity, and θ[–] is the saturation. The number density of particles $p(\mathbf{S}, t)$ satisfies the following balance equation known as the Ito-Fokker-Planck approximation:

$$\frac{\partial p}{\partial t}+\nabla\cdot(\mathbf{A}p)-\Delta\left(\frac{1}{2}\mathbf{B}\cdot\mathbf{B}^{T}p\right)=0 \tag{4}$$

EcoSLIM is an open-source particle tracking model that calculates the advection and molecular diffusion of water parcels in subsurface based on transient flow fields. The position of a water parcel is determined by neglecting the macrodispersion in Equations 1–3 as:

$$\mathbf{S}(t+\Delta t)-\mathbf{S}(t)=\mathbf{v}\Delta t+\mathbf{Z}\sqrt{2D_m\Delta t} \tag{5}$$

where D_m [L^2/T] is the molecular diffusion coefficient. The main mechanistic processes in EcoSLIM are as follows and readers can refer to Maxwell et al. (2019) for more details. Particles are dynamically added into the modeling domain where precipitation minus ET is positive (PME > 0). Particles move in subsurface until they exit the modeling domain as outflow or ET. Particles exit as outflow if they move upward to the stream bottom and as ET at places where PME < 0. Hence, the travel/residence time of outflow, ET, and groundwater can be calculated by aggregating the travel times of corresponding particles. The source composition of outflow, ET, and groundwater can be obtained by labeling particles with their water sources (e.g., snow, rainfall, or initial water). EcoSLIM is designed to work seamlessly with integrated hydrological models. Here, we apply EcoSLIM

to ParFlow coupled with CLM (ParFlow-CLM). ParFlow-CLM is a well-established integrated hydrologic model which has been applied from the pore scale to the continental simulations (Beisman et al., 2015; Maxwell & Condon, 2016). Details of ParFlow-CLM coupling and the numerical approach are described in Kollet and Maxwell (2008). The combined model simulates 3-D variably saturated flow in the subsurface and solves the full water and energy balance at the land surface. ParFlow-CLM generates three-dimensional velocities, saturation, PME, and land surface states (e.g., land surface temperature to determine the input particles as rain or snow) which are used as inputs to EcoSLIM simulations.

2.2. Prior Parallelization of EcoSLIM

EcoSLIM is written in Fortran. The particle movement routines (*Particle move* in Figure 1) consume >90% of the total simulation time in a serial run (C. Yang, Zhang, et al., 2021), and was parallelized using Fortran-OpenMP extensions on shared-memory CPU architecture in the original version (Maxwell et al., 2019). To port and scale EcoSLIM to very large domains, multiple development steps were required. First, the EcoSLIM code needed to be ported to GPUs using CUDA Fortran extensions (C. Yang, Zhang, et al., 2021) running on a single GPU. Next, the code was ported to multi-GPU systems by decomposing the total number of particles into batches that run on each GPU. In this simple setup, each GPU is responsible for a portion of particles, and OpenMP is used to manage the multiple GPUs housed within one system. This initial architecture created three primary bottlenecks to scalability. One is that the velocity field used to drive the particle movements for the entire domain needs to be stored entirely in GPU memory under this configuration. A second is that OpenMP is a shared memory parallel architecture, and thus the total number of available GPUs is limited to that equipped on one node of a cluster. A third is that certain OpenMP Fortran routines may show poor performance when ported to CUDA Fortran and need refactoring and CPU-GPU memory management has a large impact on overall code performance. To utilize GPUs across nodes on distributed platforms and remove this second bottleneck, we developed an MPI-based

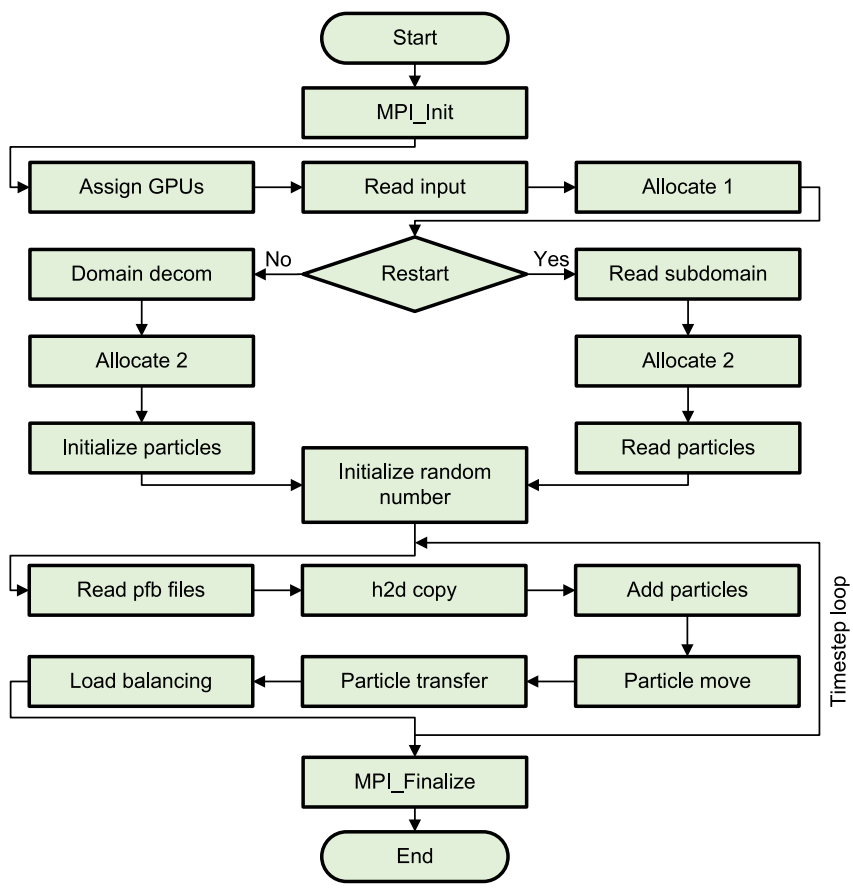

Figure 1. Flow chart of parallelized EcoSLIM in this study.

approach that balanced particles across GPUs but did not have any parallel communication of particles between GPUs (C. Yang et al., 2022). This work also optimized the GPU memory use. This advancement relaxed the first limitation but still required that the velocity field for the entire domain was available to each GPU. In this configuration, each MPI process is assigned one GPU, and each GPU is responsible for particles added into a specified area in the modeling domain. In other words, new particles are added to different GPUs based on the locations they were added into the modeling domain. Once added, particles on each GPU can move in the entire modeling domain. The advantages of the approach in C. Yang et al. (2022) are the ability to exploit the independent nature of particle movements (for the cases where particle-particle interactions are not considered) and the simplicity of using multiple GPUs without the need for complicated and potentially expensive MPI communication between GPUs. However, a primary limitation of this approach is that each GPU needs the ParFlow-CLM outputs of the entire modeling domain. The domain size is thus limited by a single GPU's memory. For very large modeling domains, such as at the continental scale, a new approach is needed.

Hence, in this study, we further remove the memory limitation by constraining the movement of particles in the subdomain where they were added. Once particles move out of the subdomain, particles will be transferred to the neighbor subdomains. As such, we totally remove the memory bottleneck either by shared memory of OpenMP or by too large modeling domains and thus realize the parallelization of particle tracking at very large scales or for very large domains. However, the details of how particles are balanced across GPUs (i.e., parallel LB) may have a significant impact on scalability and performance.

2.3. Flow Chart of EcoSLIM

A flow chart of the parallelized EcoSLIM in this study is shown in Figure 1. Once the simulation is started, MPI is initialized (*MPI_Init*), and each MPI process is assigned a unique GPU (*Assign_GPUs*). On each MPI process, input parameters are read (*Read_input*). Arrays independent of the dimensions of the subdomain are allocated (*Allocate* 1). If the simulation is newly started, we decompose the modeling domain (*Domain decom*), allocate arrays with dimensions of the subdomain (*Allocate* 2), and initialize particles in each subdomain (*Initialize particles*). Please refer to Text S1 and Figure S1 in Supporting Information S1 for more details of DDC. If the simulation is restarted, we read the decomposition information from the restart files (*Read subdomain*), allocate arrays for the subdomain based on this decomposition information (*Allocate* 2), and load particles from the restart files (*Read particles*). The pseudorandom number generator is initialized before the timestep loop (*Initialize random number*). Please refer to Text S2 in Supporting Information S1 for more details on the setup of pseudorandom numbers. In each timestep, we read necessary *.pfb* files generated by ParFlow-CLM (*Read pfb files*) and copy them from host (CPU) to device (GPU) (*h2d copy*). New particles are inserted into the subdomain where PME > 0 (*Add particles*). The particles then move in the subdomain by advection and molecular diffusion (*Particle move*). After the particle movement, particles that exit their subdomain are transferred to neighbor subdomains (*Particle transfer*). LB is optionally performed (*Load balancing*). The simulation then continues to the next time step.

3. Parallel Implementation

3.1. Domain Decomposition

The modeling domain is decomposed into subdomains (solid black boxes in Figure 2a) in horizontal directions. p and q represent splits in x- and y-directions, respectively. Variables used in the code are defined in Table 1. The number of subdomains ($p \times q$) equals the number of GPUs used in the simulation. For each subdomain, one more row/column (i.e., halo cells) is set around the real subdomain. Halo cells aim to continue the movement of particles that are out of the real boundaries in a timestep (i.e., during the execution of the particle-movement kernel). Hence, the subdomain on each GPU actually has the dimensions of the blue box with a data structure of $data(-buff + 1{:}nnx1 + buff, -buff + 1{:}nny1 + buff, 1{:}nz)$. $nnx1$, $nny1$, and nz are the dimensions of the real subdomain, whereas *buff* is the number of rows/columns expanded as halo cells. *buff* of 1 (i.e., one row/column) represents a 1 km expansion if the grid resolution is 1 km, which is the setup in all tests in Section 4. We set *buff* to 1 because groundwater moves slowly, and 1 km is long enough to avoid particles moving out of the halo cells during one timestep. Users can flexibly set *buff* in their applications. Halo cells borrowed from neighbors are labeled by each neighbor's GPU rank, while virtual cells are labeled by −1 (Figure 2). Virtual cells aim to

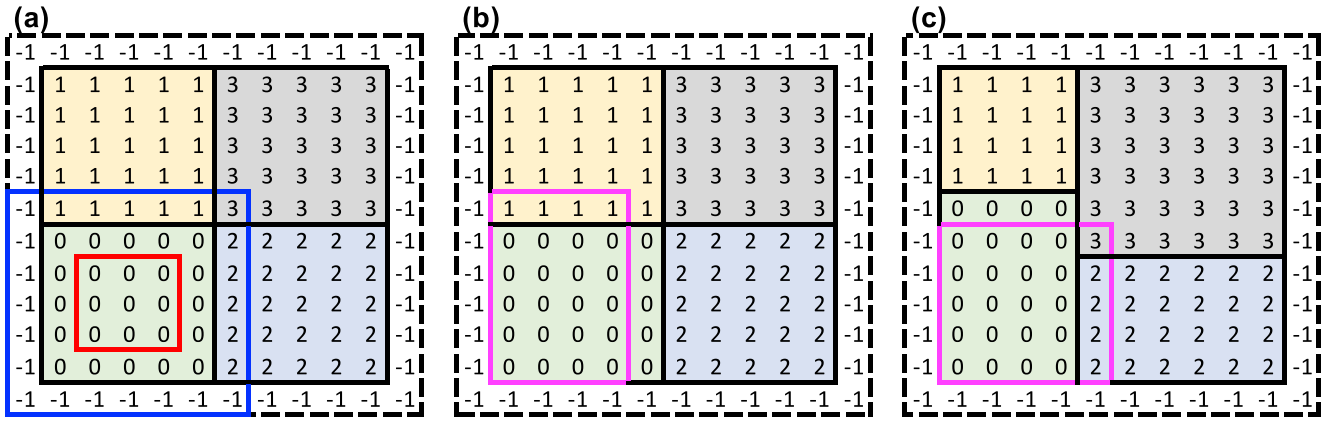

Figure 2. Diagram of subdomains. We take the lower-left subdomain in (a) as an example. The solid black box represents the real boundaries. The blue box represents the subdomain with halo cells. Grid-cells in the real domain are labeled by the Graphics Processing Unit (GPU) rank, for example, GPU0 is assigned to the lower-left subdomain. (b) and (c) show how to identify neighbor GPUs for sending and receiving particles in dynamic domain decomposition in Section 3.3.1. GPU0 receives particles from GPU1 (b) and sends particles to GPU2 and GPU3 (c). For the red rectangle in (a), please refer to Text S4 in Supporting Information S1.

maintain the consistency of the data structure of all subdomains with or without global boundaries (i.e., the real boundaries of the entire modeling domain).

3.2. Particle Transfer

Particle transfer in this section refers to the exchange of particles in halo cells between neighbor subdomains. This type of transfer can be performed either every timestep or several timesteps because *buff* of 1 km is long enough for groundwater movement in one timestep. Particle transfer in Section 3.3.1 is to exchange particles among GPUs due to the change in the topological structure of subdomains after a new DDC for LB. CUDA-Aware MPI is used to perform both types of transfers among GPUs.

3.2.1. Packed Transfer

Particle transfer is performed after the particle movement (i.e., the particle movement kernel is finished). On each GPU, sending and receiving are performed sequentially (Figure 3). In the sending part, we go through all neighbors of the subdomain. For a given neighbor *i*, we first send it a number *N_send(i),* representing the number of particles that will be sent to this neighbor. The number *N_send(i)* needs to be sent even if it is zero, so neighbor

Table 1
Physical Meanings of Variables Cited From EcoSLIM Code

Variables	Physical meaning
p	Splits of the modeling domain in *x* direction
q	Splits of the modeling domain in *y* direction
data	Gridded data such as velocities, porosity, saturation etc.
buff	The width of halo cells around a subdomain (i.e., the number of rows or columns used as halo cells)
*nnx*1	The dimension of a subdomain in *x* direction (without halo cells)
*nny*1	The dimension of a subdomain in *y* direction (without halo cells)
nz	The dimension of a subdomain in *z* direction
N_send(i)	The number of particles sent to neighbor *i* of a given subdomain
P	Array of all particles in a subdomain
P_send	A specific array for temporally saving particles to be sent
N_recv(i)	The number of particles received on a given subdomain from its neighbor *i*
P_recv	A specific array for temporally saving particles received
MPI_Waitall	MPI function

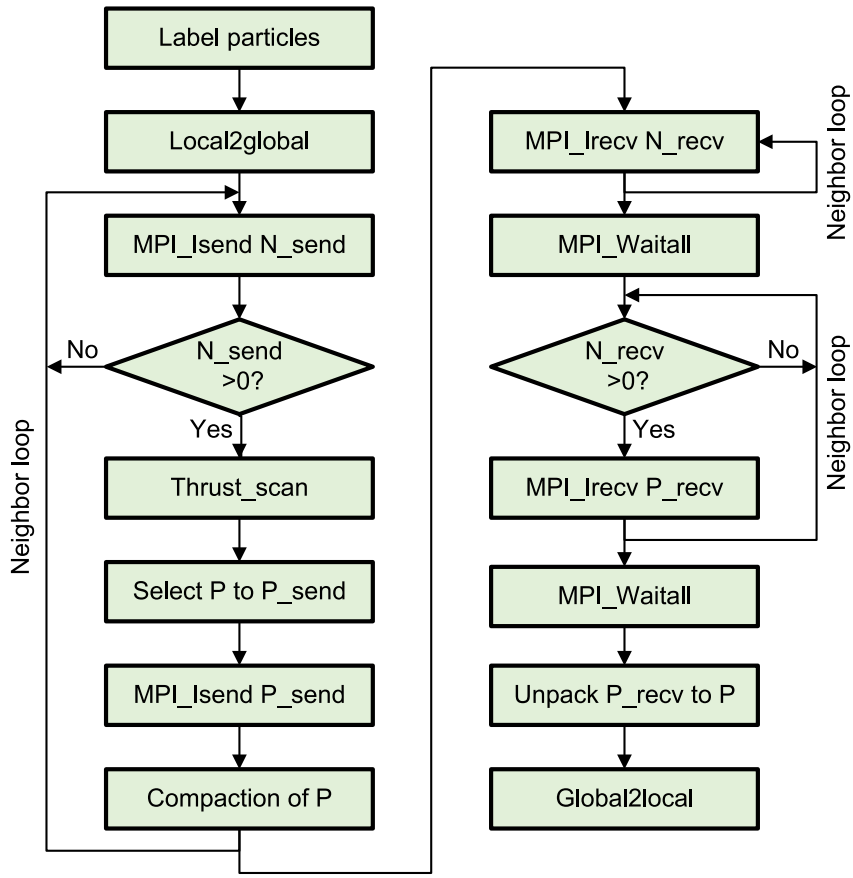

Figure 3. Flow chart of packed transfer of particles. *Label particles* means updating a specific attribute of each particle, which distinguishes particles to be sent from all particles. *Local2global* transforms particle coordinates from local coordinates in the subdomain to global coordinates in the entire domain. *Global2local* is the inverse transform of *Local2global*. *MPI_Isend* and *MPI_Irecv* are Message Passing Interface functions for sending and receiving. *Thrust_scan* is some preparation work to select particles to *P_send*; please refer to Text S4 in Supporting Information S1 for details of *Thrust_scan*.

i knows whether it will receive particles in the receiving part. If the number of particles to be sent is larger than zero [*N_send(i)* > 0], *N_send(i)* particles are selected from *P* (array of all particles, on device) to *P_send* (array of particles to be sent, on device), and *P_send* is sent to the given neighbor *i*. After sending, *P* is compacted by filling the slots of the particles that have been sent out (Figures S2 and S3 in Supporting Information S1). In the receiving part, we first go through the neighbor list to receive a number *N_recv(i)* from each neighbor *i*. This number represents the number of particles that will be received from neighbor *i*. Then, we go through the neighbors in a second round to receive *N_recv(i)* particles by *P_recv* (array of particles received, on device) from neighbor *i* with *N_recv(i)* > 0. Finally, we unpack the received particles in *P_recv* by connecting them to the end of the active particles in *P*. All MPI communications for *N_send*, *N_recv*, *P_send*, and *P_recv* are nonblocking. Hence, *MPI_Waitall* is performed after the neighbor loops for *N_recv* and *P_recv*.

3.2.2. One-By-One Transfer

One-by-one transfer is built as an alternative scheme, which has the following differences from the packed transfer. All particles that will be sent are selected to *P_send* without differentiating the neighbors. Then, a do-loop is used to send particles individually to different neighbors by going through all particles in *P_send.* In particle receiving, in the neighbor loop, particles from a given neighbor are received one by one. This scheme has the advantage when the number of particles to be transferred is small. This is because the selection of particles to *P_send* and the compaction of *P* are conducted only once, and thus it saves time relative to the packed transfer, which performs selection and compaction multiple times for multiple neighbors. When the number of particles to be transferred is large, the packed transfer is better because particles are queued in the network during the one-by-one transfer, which significantly increases the MPI communication time.

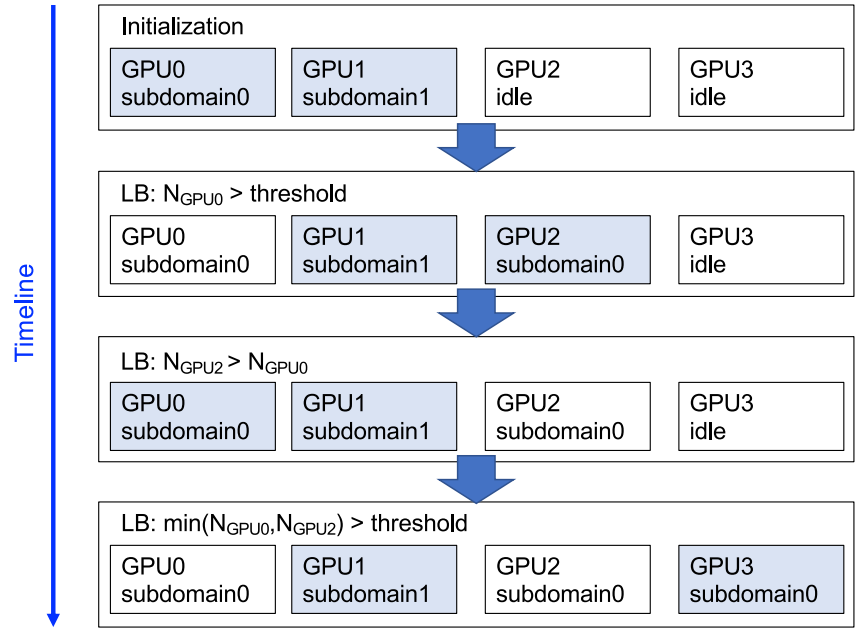

Figure 4. Diagram of the Graphics Processing Unit (GPU) help scheme. The colored GPU represents the GPU for adding new particles in a GPU group. *N* is the number of particles on a GPU.

3.3. Load Balancing

3.3.1. Dynamic DDC (LB1)

In a simulation, LB can be performed periodically based on DDC. DDC uses the orthogonal recursive bisection (ORB) adopted from other disciplines such as molecular dynamics (see details in Text S1 in Supporting Information S1). Particle sending and receiving are needed for this scheme. Figure 2 shows the identification of particles that are sent and received among the GPUs. After an operation of DDC, an updated topological structure of the subdomains is generated (Figure 2c). To find particles to receive, we use boundaries of the new subdomains to frame the old distribution of subdomains (pink rectangle in Figure 2b). To find particles to send, we use boundaries of the old subdomains to frame the new distribution of subdomains (pink rectangle in Figure 2c). Taking GPU0 as an example, GPU0 will receive particles from GPU1 and send particles to GPU2 and GPU3. Sending and receiving are sequentially performed. In each part, we use a method similar to that in the packed transfer described in Section 3.2.1. We go through the neighbor list to send/receive particle numbers first and then send/receive particles of that quantity.

3.3.2. GPU Help (LB2)

When using this novel LB scheme (Figure 4), the simulation is started using a number of GPUs smaller than the planned number of GPUs. The number of subdomains equals the starting number of GPUs. A threshold of the number of particles is set as input by users. Once the number of particles on a GPU exceeds this threshold, one idle GPU is enabled to help this GPU, that is, the new and old GPUs are responsible for the same subdomain. With the progress of the simulation, all idle GPUs are gradually enabled. Each GPU is responsible for one subdomain, and each subdomain can be assigned more than one GPU (a GPU group). For the subdomains with a GPU group, new particles are added to the GPU with the smallest number of particles in the GPU group. This LB scheme is a "light" scheme meaning that the LB itself consumes limited time since it avoids particle transfer among GPUs. However, with this LB scheme, particles are only well balanced among GPUs in the same GPU group, and particles are not rigorously balanced among all GPUs because of the lack of particle transfer between different GPU groups.

4. Parallel Performance

The code-to-code verification is presented in Text S5 and Figure S4 in Supporting Information S1. After that, we conducted tests across three spatial scales (Figure 5): the catchment scale on the Little Washita watershed (LW) in the US, the regional scale on the North China Plain (NCP), and the continental-scale on the Continental

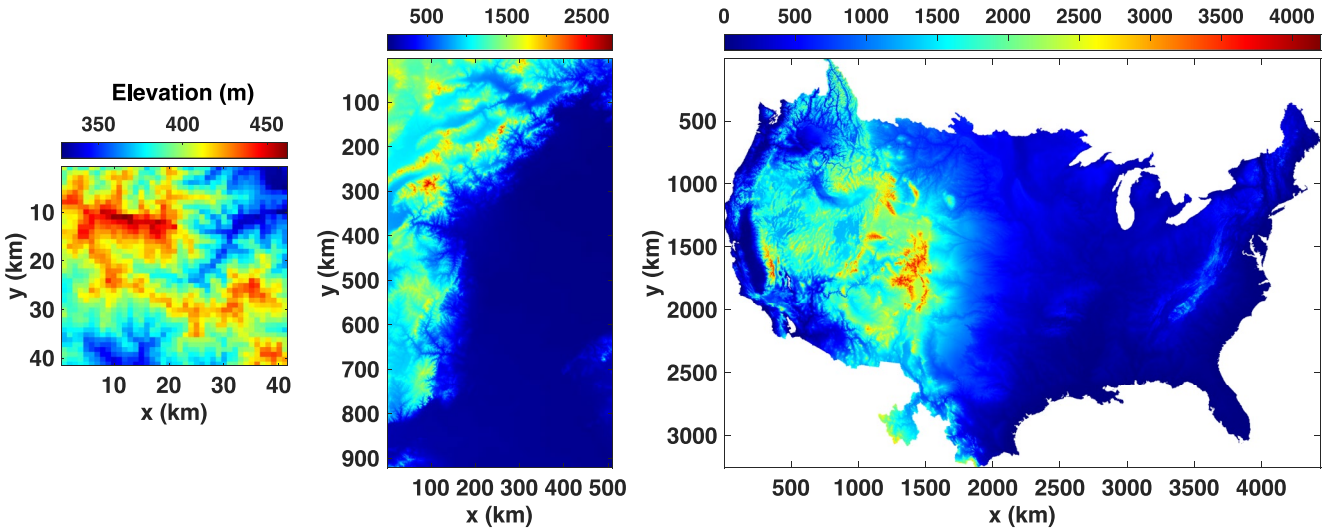

Figure 5. Modeling domains across spatial scales: the Little Washita watershed, the North China Plain, and the Continental US, from left to right.

US (CONUS). Our test cases were intentionally chosen to span a range of relevant problem scales. The first two (catchment and regional) were selected to be relevant to watershed scale hydrologic modeling (the most common applications of this type of code). Our continental scale simulation is an end member case designed to demonstrate the scalability of the problem. To our knowledge, no other particle tracking codes have ever been run at this resolution at the continental scale. The goal of our test cases is to provide a tangible demonstration of the LB and parallel performance of the new code in real world situations.

The modeling domains of LW, NCP, and CONUS have dimensions of 41-km × 41-km × 100-m, 509-km × 921-km × 102-m, and 4,442-km × 3,256-km × 392-m, respectively. The horizontal resolution is 1-km × 1-km for all three domains. The vertical resolutions from bottom to top are uniform 2-m for LW of 50 layers, 100-, 1-, 0.6-, 0.3-, and 0.1-m for NCP of 5 layers, and 200-, 100-, 50-, 25-, 10-, 5-, 1-, 0.6-, 0.3-, and 0.1-m for CONUS of 10 layers. The timestep is hourly. The ParFlow-CLM models are adopted from Maxwell et al. (2016) for LW and C. Yang et al. (2020) with modifications for NCP (C. Yang et al., 2022). The ParFlow-CLM model with irregular boundaries is an advanced version of Maxwell et al. (2015) and has not been published yet. Tests were conducted on Della-GPU cluster at Princeton University. Each GPU node is equipped with two NVIDIA A100 GPUs and two 2.60-GHz AMD EPYC 7H12 sockets. Each socket has 64 cores without hyper-threading. In C. Yang et al. (2022), a test of ~5.6-million particles using one NVIDIA A100 GPU had >5-fold speedup relative to that using 128 2.60-GHz AMD EPYC 7H12 cores. A better performance of ~7-fold speedup was shown for more particles of ~17.4-million. Though the tests provided here were conducted on A100, the code can be run on a wide range of NVIDIA GPU architectures such as Pascal, Volta, and Ampere (A100 included).

4.1. Little Washita

Each test on LW domain had 24,000 timesteps. Each test repeatedly used the outputs of 120 timesteps from ParFlow-CLM for 200 cycles. Seven tests were conducted, and the results are listed in Table 2. LB was conducted every 20-hr in all tests except for test 5 because test 5 was run without LB. Tests 1–4 of two subdomains were planned with four GPUs and started with two GPUs. Tests 5 and 7 of four subdomains had the same number of planned and started GPUs of four. Test 6 was planned with two GPUs and started with one GPU, so the domain was not decomposed in test 6. Speedup is the time of particle movement used by the parallelized code on GPUs relative to that used by the CPU code with 128 EPYC cores. For results from the parallelized code, the time used for data copies between host and device, particle transfer, and LB were also included.

In test 1, two particles were added into a grid cell for each precipitation event (i.e., PME > 0) at each timestep (*Injected number* in Table 2). A speedup of ~8-fold using four A100 GPUs relative to 128 EPYC cores meets the basic requirement of the GPU parallelism (Hokkanen et al., 2021). When the injected number of particles was increased to 32 (test 2), the speedup was significantly increased to ~25-fold. With a further increase of the

Table 2
Test Results for Parallelized EcoSLIM in This Study

Tests	Started GPUs	Planned GPUs	*p*	*q*	Injected number	Particle transfer	LB	Speedup
Test results on Little Washita watershed								
Test 1	2	4	2	1	2 per hr	Every timestep	LB2	8.18
Test 2	2	4	2	1	32 per hr	Every timestep	LB2	**25.49**
Test 3	2	4	2	1	128 per hr	Every timestep	LB2	26.75
Test 4	2	4	2	1	32 per hr	/	LB2	26.98
Test 5	4	4	2	2	32 per hr	/	/	12.72
Test6	1	2	1	1	32 per hr	Every timestep	LB2	13.73
Test 7	4	4	2	2	32 per hr	Every timestep	LB1	22.26

Tests	Started GPUs	Planned GPUs	*p*	*q*	Injected number	LB	LB frequency	Speedup
Test results on North China Plain								
Test1	4	4	2	2	1 per day	/	/	/
Test2	2	4	1	2	1 per day	LB2	24 hr	−16.29%
Test3	1	4	1	1	1 per day	LB2	24 hr	−23.51%
Test4	4	4	2	2	1 per day	LB1	24 hr	−19.51%
Test5	4	4	2	2	1 per day	LB1	240 hr	−37.56%
Test6	4	8	2	2	1 per day	LB2	24 hr	−52.34%

Tests	Started GPUs	Planned GPUs	*p*	*q*	Injected number	LB	LB frequency	Wall-clock
Test results on continental US								
Test1	16	16	4	4	1 per day	/	/	14-min (0.8 B)
Test2	16	16	4	4	1 per day	LB1	240 hr	65-min (1.6 B)
Test3	8	16	4	2	1 per day	LB2	24 hr	36-min (1.2 B)

injected number of particles to 128 (test 3), the speedup was ~26-fold, without any significant improvement. If we disabled particle transfer (test 4), an improvement from 25.49-fold to 26.98-fold was observed. If we disabled both particle transfer and LB (test 5), speedup largely decreased to less than half of that in test 4. Increasing the GPU number from two to four (test 6 relative to test 2) showed great parallel scaling with an increase in speedup from 13.73- to 25.49-fold. Test 7, with *dynamic DDC* for LB, also showed a significant speedup of 22.26-fold. The maximum number of active particles in the simulation on LW was ~25-million (summation of all GPUs) in test 2. The excellent performance of the two LB schemes is shown in Figure 6.

4.2. North China Plain

Each test on the NCP domain was a 10-year simulation by repeatedly using 1-year outputs from ParFlow-CLM. The maximum number of active particles in the simulations was ~70-million (summation of all GPUs). Particle transfer was performed at each timestep. The timing parts were the same as those on LW domain, and the speedup in this section was the change in time consumption of each test relative to test 1 (baseline). Test 1 was conducted using four GPUs without LB. Tests 2 and 3 used *GPU help* for LB. Test 2 decomposed the modeling domain into two, whereas Test 3 did not decompose the modeling domain. Tests 2 and 3 reduced the time used in test 1 by 16.29% and 23.51%, respectively. Tests 4 and 5 used *dynamic DDC* for LB with frequencies of every 24-hr and every 240-hr, respectively. They reduced the time used in Test 1 by 19.51% and 37.56%, respectively. Hence, tests 2–5 demonstrate the efficiency of both LB schemes. Test 6 increased the number of GPUs to eight by using *GPU help* scheme for LB. It showed a time reduction of 52.34%, 47.84%, 50.43%, and 36.10% relative to that used in tests 2–5, demonstrating great parallel scaling with increasing GPU numbers. The particle distribution on GPUs for each test is shown in Figure S5 in Supporting Information S1. Except for the input/output time, all simulations can be finished around one-hour (wall-clock time), which also included the time consumption before the timestep loop.

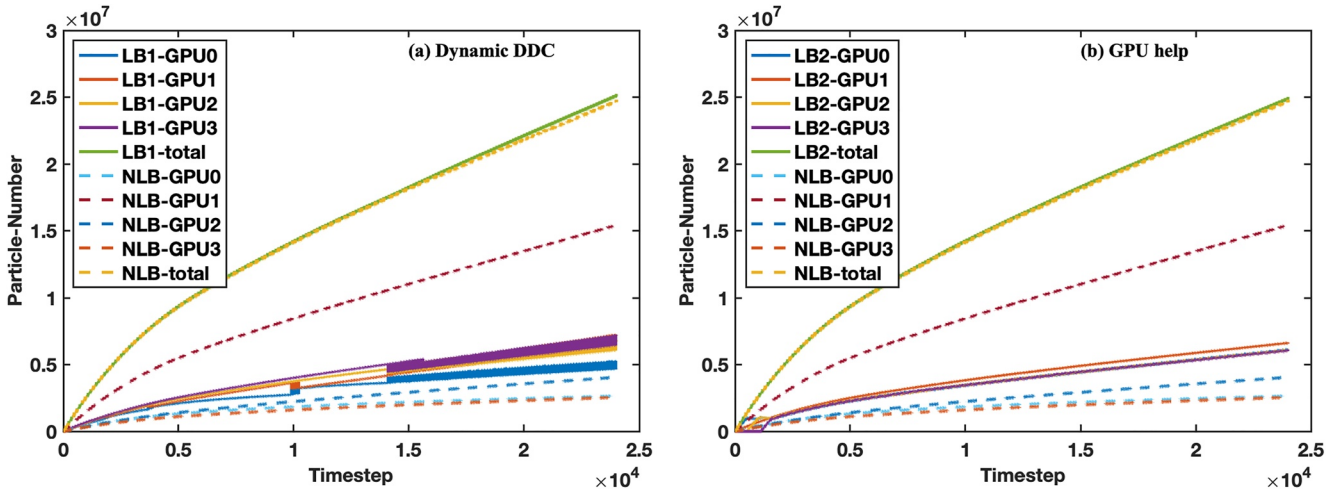

Figure 6. Effects of load balancing schemes on Little Washita domain: (a) Dynamic domain decomposition (DDC) and (b) Graphics Processing Unit help.

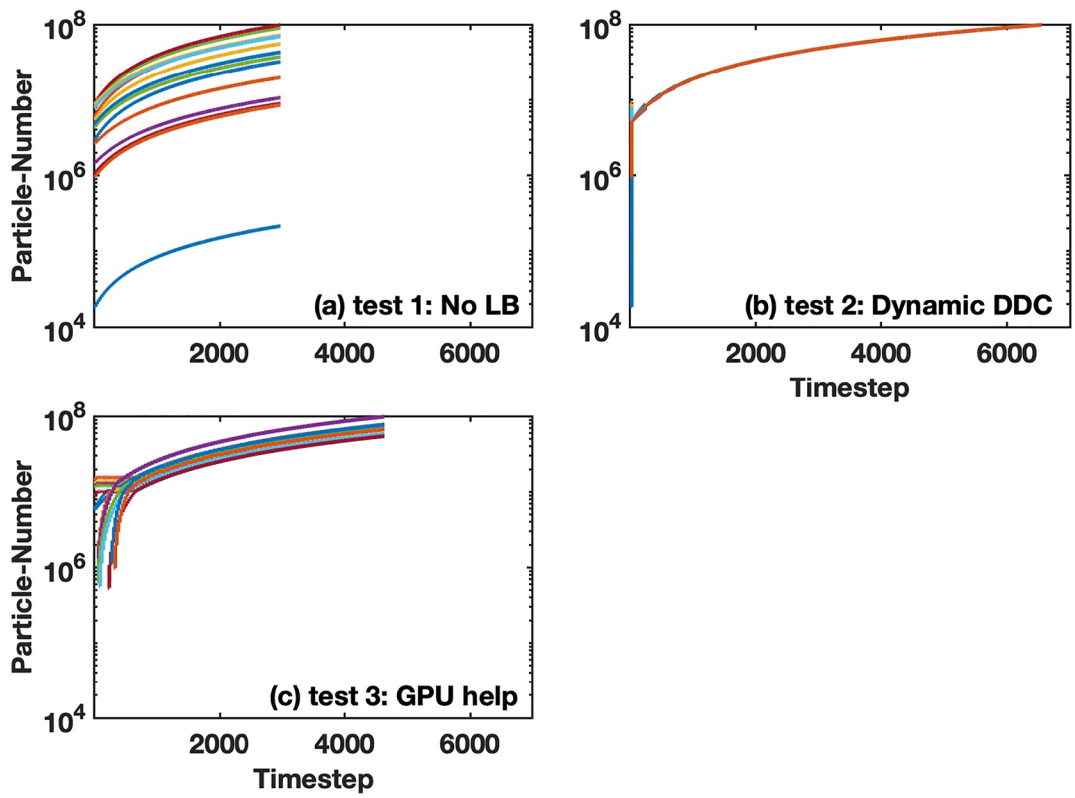

Figure 7. Distributions of particles on Graphics Processing Units (GPUs) for the tests on Continental US. Each line represents a GPU. Different colors of the lines differentiate GPUs. Test 1 in (a) ran without load balancing (LB), test 2 ran with dynamic domain decomposition for LB, and test 3 ran with GPU help for LB.

4.3. Continental US

Each test on CONUS used 16 GPUs with a maximum permitted particle number of 100-million on each GPU. Each test stopped due to this limitation, that is, the number of particles on any of the 16 GPUs first achieved 100-million. The simulation time (wall-clock) mentioned in this section was the total simulation time, excluding the input/output time. Without LB, 2,975 timesteps were finished with a maximum active particle number of 0.8-billion (summation of all GPUs), and the simulation time was 14 min. With *dynamic DDC*, 6,551 timesteps were achieved with a maximum active particle number of 1.6-billion (summation of all GPUs), and the simulation time was 65-min. With *GPU help*, 4,631 timesteps were achieved with a maximum active particle number of 1.2-billion (summation of all GPUs), and the total simulation time was 36-min. Figure 7 shows the particle distribution on GPUs in the CONUS tests. All three tests showed reasonable wall-clock time and excellent performance of the LB schemes.

5. Conclusions

In this study, we develop and test a parallel framework on distributed, multi-GPU platforms for EcoSLIM, enabling large-scale particle tracking with high spatiotemporal resolutions. EcoSLIM is a Lagrangian particle tracking model which simulates water flow paths, water ages, and source water mixing based on an integrated hydrologic model configured with land surface processes. To the best of our knowledge, such a particle tracking tool handling cross-scale simulations is lacking in communities of hydrology and Earth Surface Processes (Clark et al., 2015; Evaristo & McDonnell, 2017; Fan, 2016; Fan et al., 2019; McDonnell, 2017; McDonnell & Beven, 2014). Tests (4 NVIDIA A100 GPUs relative to 128 AMD EPYC cores) based on Little Washita watershed showed a significant speedup of 25.49-fold (8-fold is the basic requirement). Tests based on Little Washita watershed and the NCP showed excellent parallel scaling. Tests based on NCP and Continental US demonstrated the capability of EcoSLIM to handle regional- to continental-scale simulations with reasonable wall-clock time. Here, we take EcoSLIM as an example, but the parallel framework is portable for other particle tracking models in Earth System Science, such as atmospheric and oceanic models. The parallelized EcoSLIM is a promising tool for the hydrologic community and ESM developers for scientific exploration. More applications are expected from the community to better understand the speed bottleneck and to further improve the proposed particle transfer and LB schemes.

Conflict of Interest

The authors declare no conflicts of interest relevant to this study.

Data Availability Statement

EcoSLIM code, input scripts, and test results can be found at: https://github.com/aureliayang/EcoSLIM_CONUS. A copy (C. Yang et al., 2023) from Github including EcoSLIM code, input scripts, and test results is archived through Zenodo at: https://doi.org/10.5281/zenodo.7302297.

Acknowledgments
We thank the anonymous reviewers and associate editor whose comments substantially improved our manuscript. This work was supported by the National Natural Science Foundation of China (NSFC-41807198). This work was also supported by the U.S. Department of Energy Office of Science, Offices of Advanced Scientific Computing Research and Biological and Environmental Sciences IDEAS project, and Watershed Function Scientific Focus Area under Award Number DE-AC02-05CH11231. The simulations presented in this article were performed on computational resources managed and supported by Princeton Research Computing, a consortium of groups including the Princeton Institute for Computational Science and Engineering (PICSciE), and the Office of Information Technology's High-Performance Computing Center and Visualization Laboratory at Princeton University.

References

Bechtold, M., Vanderborght, J., Ippisch, O., & Vereecken, H. (2011). Efficient random walk particle tracking algorithm for advective-dispersive transport in media with discontinuous dispersion coefficients and water contents. *Water Resources Research*, *47*(10), W10526. https://doi.org/10.1029/2010wr010267

Beisman, J. J., Maxwell, R. M., Navarre-Sitchler, A. K., Steefel, C. I., & Molins, S. (2015). ParCrunchFlow: An efficient, parallel reactive transport simulation tool for physically and chemically heterogeneous saturated subsurface environments. *Computational Geosciences*, *19*(2), 403–422. https://doi.org/10.1007/s10596-015-9475-x

Benettin, P., Nehemy, M. F., Asadollahi, M., Pratt, D., Bensimon, M., McDonnell, J. J., & Rinaldo, A. (2021). Tracing and closing the water balance in a vegetated lysimeter. *Water Resources Research*, *57*(4), e2020WR029049. https://doi.org/10.1029/2020WR029049

Bloschl, G., Bierkens, M. F. P., Chambel, A., Cudennec, C., Destouni, G., Fiori, A., et al. (2019). Twenty-three unsolved problems in hydrology (UPH)—A community perspective. *Hydrological Sciences Journal-Journal Des Sciences Hydrologiques*, *64*(10), 1141–1158. https://doi.org/10.1080/02626667.2019.1620507

Clark, M. P., Fan, Y., Lawrence, D. M., Adam, J. C., Bolster, D., Gochis, D. J., et al. (2015). Improving the representation of hydrologic processes in Earth System Models. *Water Resources Research*, *51*(8), 5929–5956. https://doi.org/10.1002/2015wr017096

Cremer, C. J. M., Neuweiler, I., Bechtold, M., & Vanderborght, J. (2016). Solute transport in heterogeneous soil with time-dependent boundary conditions. *Vadose Zone Journal*, *15*(6), 1–17. https://doi.org/10.2136/vzj2015.11.0144

de Graaf, I. E. M., & Stahl, K. (2022). A model comparison assessing the importance of lateral groundwater flows at global-scale. *Environmental Research Letters*. https://doi.org/10.1088/1748-9326/ac50d2

Doos, K., Jonsson, B., & Kjellsson, J. (2017). Evaluation of oceanic and atmospheric trajectory schemes in the TRACMASS trajectory model v6.0. *Geoscientific Model Development*, *10*(4), 1733–1749. https://doi.org/10.5194/gmd-10-1733-2017

Döös, K., Kjellsson, J., & Jönsson, B. (2013). TRACMASS—A Lagrangian trajectory model. In T. Soomere & E. Quak (Eds.), *Preventive methods for coastal protection: Towards the use of ocean dynamics for pollution control* (pp. 225–249). Springer International Publishing. https://doi.org/10.1007/978-3-319-00440-2_7

Engdahl, N. B., Schmidt, M. J., & Benson, D. A. (2019). Accelerating and parallelizing Lagrangian simulations of mixing-limited reactive transport. *Water Resources Research*, *55*(4), 3556–3566. https://doi.org/10.1029/2018wr024361

Englert, A., Hardelauf, H., Vanderborght, J., & Vereecken, H. (2003). Numerical modelling of flow and transport on massively parallel computers. In *NIC symposium 2004, poceedings* (Vol. 20, pp. 409–418).

Evaristo, J., & McDonnell, J. J. (2017). Prevalence and magnitude of groundwater use by vegetation: A global stable isotope meta-analysis. *Scientific Reports*, *7*(1), 44110. https://doi.org/10.1038/srep44110

Fan, Y. (2015). Groundwater in the Earth's critical zone: Relevance to large-scale patterns and processes. *Water Resources Research*, *51*(5), 3052–3069. https://doi.org/10.1002/2015wr017037

Fan, Y. (2016). How much and how old? *Nature Geoscience*, *9*(2), 93–94. https://doi.org/10.1038/ngeo2609

Fan, Y., Clark, M., Lawrence, D. M., Swenson, S., Band, L. E., Brantley, S. L., et al. (2019). Hillslope hydrology in global change research and earth system modeling. *Water Resources Research*, *55*(2), 1737–1772. https://doi.org/10.1029/2018wr023903

Fan, Y., Li, H., & Miguez-Macho, G. (2013). Global patterns of groundwater table depth. *Science*, *339*(6122), 940–943. https://doi.org/10.1126/science.1229881

Fuhrer, O., Chadha, T., Hoefler, T., Kwasniewski, G., Lapillonne, X., Leutwyler, D., et al. (2018). Near-global climate simulation at 1 km resolution: Establishing a performance baseline on 4888 GPUs with COSMO 5.0. *Geoscientific Model Development*, *11*(4), 1665–1681. https://doi.org/10.5194/gmd-11-1665-2018

Hammond, G. E., Lichtner, P. C., & Mills, R. T. (2014). Evaluating the performance of parallel subsurface simulators: An illustrative example with PFLOTRAN. *Water Resources Research*, *50*(1), 208–228. https://doi.org/10.1002/2012wr013483

Hartmann, A., Jasechko, S., Gleeson, T., Wada, Y., Andreo, B., Barbera, J. A., et al. (2021). Risk of groundwater contamination widely underestimated because of fast flow into aquifers. *Proceedings of the National Academy of Sciences of the United States of America*, *118*(20), e2024492118. https://doi.org/10.1073/pnas.2024492118

Hokkanen, J., Kollet, S., Kraus, J., Herten, A., Hrywniak, M., & Pleiter, D. (2021). Leveraging HPC accelerator architectures with modern techniques—Hydrologic modeling on GPUs with ParFlow. *Computational Geosciences*, *25*(5), 1579–1590. https://doi.org/10.1007/s10596-021-10051-4

Ji, X. H., Li, D. D., Cheng, T. P., Wang, X. S., & Wang, Q. (2014). Parallelization of MODFLOW Using a GPU Library. *Groundwater*, *52*(4), 618–623. https://doi.org/10.1111/gwat.12104

Ji, X. H., Luo, M. L., & Wang, X. S. (2019). Accelerating streamline tracking in groundwater flow modeling on GPUs. *Groundwater*, *58*(4), 638–644. https://doi.org/10.1111/gwat.12959

Keune, J., Gasper, F., Goergen, K., Hense, A., Shrestha, P., Sulis, M., & Kollet, S. (2016). Studying the influence of groundwater representations on land surface-atmosphere feedbacks during the European heat wave in 2003. *Journal of Geophysical Research-Atmospheres*, *121*(22), 13301–13325. https://doi.org/10.1002/2016jd025426

Kirchner, J. W., Feng, X. H., & Neal, C. (2000). Fractal stream chemistry and its implications for contaminant transport in catchments. *Nature*, *403*(6769), 524–527. https://doi.org/10.1038/35000537

Kollet, S. J., & Maxwell, R. M. (2008). Capturing the influence of groundwater dynamics on land surface processes using an integrated, distributed watershed model. *Water Resources Research*, *44*(2), W02402. https://doi.org/10.1029/2007wr006004

Kollet, S. J., Maxwell, R. M., Woodward, C. S., Smith, S., Vanderborght, J., Vereecken, H., & Simmer, C. (2010). Proof of concept of regional scale hydrologic simulations at hydrologic resolution utilizing massively parallel computer resources. *Water Resources Research*, *46*(4), W04201. https://doi.org/10.1029/2009wr008730

Kuppel, S., Tetzlaff, D., Maneta, M. P., & Soulsby, C. (2018). EcH(2)O-iso 1.0: Water isotopes and age tracking in a process-based, distributed ecohydrological mode. *Geoscientific Model Development*, *11*(7), 3045–3069. https://doi.org/10.5194/gmd-11-3045-2018

Kuppel, S., Tetzlaff, D., Maneta, M. P., & Soulsby, C. (2020). Critical zone storage controls on the water ages of ecohydrological outputs. *Geophysical Research Letters*, *47*(16), e2020GL088897. https://doi.org/10.1029/2020gl088897

Leutwyler, D., Fuhrer, O., Lapillonne, X., Luthi, D., & Schar, C. (2016). Towards European-scale convection-resolving climate simulations with GPUs: A study with COSMO 4.19. *Geoscientific Model Development*, *9*(9), 3393–3412. https://doi.org/10.5194/gmd-9-3393-2016

Luijendijk, E., Gleeson, T., & Moosdorf, N. (2020). Fresh groundwater discharge insignificant for the world's oceans but important for coastal ecosystems. *Nature Communications*, *11*(1), 1260. https://doi.org/10.1038/s41467-020-15064-8

Maneta, M. P., & Silverman, N. L. (2013). A spatially distributed model to simulate water, energy, and vegetation dynamics using information from regional climate models. *Earth Interactions*, *17*(11), 1–44. https://doi.org/10.1175/2012ei000472.1

Martínez-de la Torre, A., & Miguez-Macho, G. (2019). Groundwater influence on soil moisture memory and land–atmosphere fluxes in the Iberian Peninsula. *Hydrology and Earth System Sciences*, *23*(12), 4909–4932. https://doi.org/10.5194/hess-23-4909-2019

Maxwell, R. M., & Condon, L. E. (2016). Connections between groundwater flow and transpiration partitioning. *Science*, *353*(6297), 377–380. https://doi.org/10.1126/science.aaf7891

Maxwell, R. M., Condon, L. E., Danesh-Yazdi, M., & Bearup, L. A. (2019). Exploring source water mixing and transient residence time distributions of outflow and evapotranspiration with an integrated hydrologic model and Lagrangian particle tracking approach. *Ecohydrology*, *12*(1), e2042. https://doi.org/10.1002/eco.2042

Maxwell, R. M., Condon, L. E., & Kollet, S. J. (2015). A high-resolution simulation of groundwater and surface water over most of the continental US with the integrated hydrologic model ParFlow v3. *Geoscientific Model Development*, *8*(3), 923–937. https://doi.org/10.5194/gmd-8-923-2015

Maxwell, R. M., Kollet, S. J., Smith, S. G., Woodward, C. S., Falgout, R. D., Ferguson, I. M., et al. (2016). ParFlow user's manual. In *Integrated Ground-Water Modeling Center Report GWMI 2016-01* (p. 154).

McDonnell, J. J. (2017). Beyond the water balance. *Nature Geoscience*, *10*(6), 396. https://doi.org/10.1038/ngeo2964

McDonnell, J. J., & Beven, K. (2014). Debates—The future of hydrological sciences: A (common) path forward? A call to action aimed at understanding velocities, celerities and residence time distributions of the headwater hydrograph. *Water Resources Research*, *50*(6), 5342–5350. https://doi.org/10.1002/2013wr015141

Morales-Hernandez, M., Sharif, M. B., Kalyanapu, A., Ghafoor, S. K., Dullo, T. T., Gangrade, S., et al. (2021). TRITON: A multi-GPU open source 2D hydrodynamic flood model. *Environmental Modelling & Software*, *141*, 105034. https://doi.org/10.1016/j.envsoft.2021.105034

Pisso, I., Sollum, E., Grythe, H., Kristiansen, N. I., Cassiani, M., Eckhardt, S., et al. (2019). The Lagrangian particle dispersion model FLEXPART version 10.4. *Geoscientific Model Development*, *12*(12), 4955–4997. https://doi.org/10.5194/gmd-12-4955-2019

Pollock, D. W. (2016). User guide for MODPATH version 7—a particle-tracking model for MODFLOW (report) (2016-1086). (Open-File report, Issue. U. S. G. Survey). Retrieved from http://pubs.er.usgs.gov/publication/ofr20161086

Rahman, M., Sulis, M., & Kollet, S. J. (2015). The subsurface-land surface-atmosphere connection under convective conditions. *Advances in Water Resources*, *83*, 240–249. https://doi.org/10.1016/j.advwatres.2015.06.003

Rapp, G. A., Condon, L. E., & Markovich, K. H. (2020). Sensitivity of simulated mountain block hydrology to subsurface conceptualization. *Water Resources Research*, *56*(10), e2020WR027714. https://doi.org/10.1029/2020WR027714

Rizzo, C. B., Nakano, A., & de Barros, F. P. J. (2019). PAR(2): Parallel random walk particle tracking method for solute transport in porous media. *Computer Physics Communications*, *239*, 265–271. https://doi.org/10.1016/j.cpc.2019.01.013

Schauer, L., Schmidt, M. J., Engdahl, N. B., Pankavich, S. D., Benson, D. A., & Bolster, D. (2022). Parallelized domain decomposition for multi-dimensional Lagrangian random walk, mass-transfer particle tracking schemes. *Geoscientific Model Development*, *16*, 833–849. https://doi.org/10.2139/ssrn.4028727

Schroder, N., Javaux, M., Vanderborght, J., Steffen, B., & Vereecken, H. (2012). Effect of root water and solute uptake on apparent soil dispersivity: A simulation study. *Vadose Zone Journal*, *11*(3), 120009. https://doi.org/10.2136/vzj2012.0009

Smith, A., Tetzlaff, D., Kleine, L., Maneta, M., & Soulsby, C. (2021). Quantifying the effects of land use and model scale on water partitioning and water ages using tracer-aided ecohydrological models. *Hydrology and Earth System Sciences*, *25*(4), 2239–2259. https://doi.org/10.5194/hess-25-2239-2021

Sprenger, M., Stumpp, C., Weiler, M., Aeschbach, W., Allen, S. T., Benettin, P., et al. (2019). The demographics of water: A review of water ages in the critical zone. *Reviews of Geophysics*, *57*(3), 800–834. https://doi.org/10.1029/2018RG000633

Tompson, A. F. B., & Dougherty, D. E. (1988). On the use of particle tracking methods for solute transport in porous media. In M. A. Celia, L. A. Ferrand, C. A. Brebbia, W. G. Gray, & G. F. Pinder (Eds.), *Developments in water science* (Vol. 36, pp. 227–232). Elsevier. https://doi.org/10.1016/S0167-5648(08)70094-7

Tompson, A. F. B., & Gelhar, L. W. (1990). Numerical simulation of solute transport in three-dimensional, randomly heterogeneous porous media. *Water Resources Research*, *26*(10), 2541–2562. https://doi.org/10.1029/WR026i010p02541

van Sebille, E., Griffies, S. M., Abernathey, R., Adams, T. P., Berloff, P., Biastoch, A., et al. (2018). Lagrangian ocean analysis: Fundamentals and practices. *Ocean Modelling*, *121*, 49–75. https://doi.org/10.1016/j.ocemod.2017.11.008

van Vliet, M. T. H., Florke, M., & Wada, Y. (2017). Quality matters for water scarcity. *Nature Geoscience*, *10*(11), 800–802. https://doi.org/10.1038/ngeo3047

Wang, B., Wald, I., Morrical, N., Usher, W., Mu, L., Thompson, K., & Hughes, R. (2022). An GPU-accelerated particle tracking method for Eulerian-Lagrangian simulations using hardware ray tracing cores. *Computer Physics Communications*, *271*, 108221. https://doi.org/10.1016/j.cpc.2021.108221

Wilusz, D. C., Harman, C. J., Ball, W. B., Maxwell, R. M., & Buda, A. R. (2019). Using particle tracking to understand flow paths, age distributions, and the paradoxical origins of the inverse storage effect in an experimental catchment. *Water Resources Research*, *56*(4), e24397. https://doi.org/10.1029/2019wr025140

Wörman, A., Packman, A. I., Marklund, L., Harvey, J. W., & Stone, S. H. (2007). Fractal topography and subsurface water flows from fluvial bedforms to the continental shield. *Geophysical Research Letters*, *34*(7), L07402. https://doi.org/10.1029/2007GL029426

Xie, Z. H., Liu, S., Zeng, Y. J., Gao, J. Q., Qin, P. H., Jia, B. H., et al. (2018). A high-resolution land model with groundwater lateral flow, water use, and soil freeze-thaw front dynamics and its application in an endorheic basin. *Journal of Geophysical Research-Atmospheres*, *123*(14), 7204–7222. https://doi.org/10.1029/2018jd028369

Yang, C., Li, H.-Y., Fang, Y., Cui, C., Wang, T., Zheng, C., et al. (2020). Effects of groundwater pumping on ground surface temperature: A regional modeling study in the North China Plain. *Journal of Geophysical Research: Atmospheres*, *125*(9), e2019JD031764. https://doi.org/10.1029/2019jd031764

Yang, C., Maxwell, R. M., & Valent, R. (2022). Accelerating the Lagrangian simulation of water ages on distributed, multi-GPU platforms: The importance of dynamic load balancing. *Computers & Geosciences*, *166*, 105189. https://doi.org/10.1016/j.cageo.2022.105189

Yang, C., Ponder, C., Wang, B., Tran, H., Zhang, J., Swilley, J., et al. (2023). EcoSLIM CONUS: November 8, 2022 Release (Version 1.0) [Software]. Zenodo. https://doi.org/10.5281/zenodo.7302297

Yang, C., Zhang, Y.-K., Liang, X., Olschanowsky, C., Yang, X., & Maxwell, R. (2021). Accelerating the Lagrangian particle tracking of residence time distributions and source water mixing towards large scales. *Computers & Geosciences*, *151*, 104760. https://doi.org/10.1016/j.cageo.2021.104760

Yang, J., Heidbüchel, I., Musolff, A., Reinstorf, F., & Fleckenstein, J. H. (2018). Exploring the dynamics of transit times and subsurface mixing in a small agricultural catchment. *Water Resources Research*, *54*(3), 2317–2335. https://doi.org/10.1002/2017wr021896

Yang, X., Tetzlaff, D., Soulsby, C., Smith, A., & Borchardt, D. (2021). Catchment functioning under prolonged drought stress: Tracer-aided ecohydrological modeling in an intensively managed agricultural catchment. *Water Resources Research*, *57*(3), e2020WR029094. https://doi.org/10.1029/2020WR029094

References From the Supporting Information

Egorova, M. S., Dyachkov, S. A., Parshikov, A. N., & Zhakhovsky, V. V. (2019). Parallel SPH modeling using dynamic domain decomposition and load balancing displacement of Voronoi subdomains. *Computer Physics Communications*, *234*, 112–125. https://doi.org/10.1016/j.cpc.2018.07.019

Furuichi, M., & Nishiura, D. (2017). Iterative load-balancing method with multigrid level relaxation for particle simulation with short-range interactions. *Computer Physics Communications*, *219*, 135–148. https://doi.org/10.1016/j.cpc.2017.05.015

NVIDIA. (2022a). cuRAND Library. PG-05328-050_vRelease Version.

NVIDIA. (2022b). Thrust quick start guide. DU-06716-001_v11.6.

Ruetsch, G., & Fatica, M. (2014). *CUDA Fortran for scientists and engineers: Best practices for efficient CUDA Fortran programming*. Morgan Kaufmann, an imprint of Elsevier.