Accurate and Efficient Multi-scale Flow Simulation using a Hybrid CFD-MD Approach

**Abstract**

We investigate numerical and computational issues associated

with hybrid computational fluid dynamics (CFD) -

molecular dynamics (MD) simulation methodologies.

Our current hybrid CFD-MD simulation framework is based on reliable CFD and MD codes linked together through hybrid interfaces that are often based on constrained Lagrangian dynamics for data exchange between continuum and MD regions.

We argue that (1) the statistical error associated with

the sampled average of the molecular solution, and

(2) hybrid boundary conditions via

an extrapolation methodology, may lead to

non-negligible spatial and temporal inaccuracies

of the hybrid solution.

Hence, in this study we propose two strategies for minimizing

the statistical noise inherent in the MD solution:

(1) optimization of coupling parameters through

the molecular dynamic simulation of a stationary flow field,

and (2) averaging over multiple independent replicas

of the simulation system.

In addition, we introduce a new temporal coupling scheme

based on a 'prediction-correction' approach

together with interpolation in time, which provides

a more time-accurate hybrid constraint/boundary condition.

With respect to computational performance, co-scheduling and

load-balancing of logically distributed tasks are

two important considerations in CFD-MD coupled simulations.

We employ a BigJob computational framework, similar to

a Pilot-job, and incorporate a load-balancing function

for the computing efficiency of coupled simulation.

The optimized hybrid framework is applied to solving two classical flow problems under Stokes (no interial effects) conditions. One is the impulsively started Couette flow, and the other is an oscillating boundary Stokes flow, which is induced by the periodic motion of one flat plate in the channel system.

We emphasize that

(1) the determination of coupling parameters

through the molecular dynamic simulation of a stationary flow

and the averaging of multiple independent replicas contribute

much on suppressing the statistical noise

in relatively low-speed flow fields of O(10) m/s velocity,

(2) the accuracy of unsteady flow solution is increased

by applying a 'prediction-correction' temporal coupling

scheme,

and (3) efficient coupled simulation is conducted

by employing a BigJob computational framework

along with the load-balancing capability.

Keywords: Hybrid computational fluid dynamics particle dynamics approach, Constrained Lagrangian dynamics, Temporal coupling scheme, Simple API for Grid Applications, BigJob framework

1. **Introduction and Motivation**

A hybrid continuum dynamics -particle dynamics approach is defined as expressing macroscopic flow physics by the continuum hypothesis while capturing the complex flow characteristics around obstacles by solving finer-resolution particle dynamics. The hybrid method is more accurate than the traditional Computational Fluid Dynamics (CFD) technique, since it does not require empirical boundary condition modeling in describing intermolecular effect near the solid-fluid interface. Also, the hybrid approach provides a better computational efficiency than particle dynamics simulation by replacing numerous particles in moderate flow regime to sparser continuum mesh system. In this section, we address a number of previous studies on the hybrid simulation and raise unsolved numerical and computational diﬃculties which shall be covered in this paper.

* 1. *Previous Works*

Hybrid simulation approaches can be grouped into two according to the microscopic solution method coupled into macroscopic continuum domain. [1] One is the coarse-grained particle simulations by using Direct Simulation Monte Carlo (DSMC) for dilute gas [2, 3] or Dissipative Particle Dynamics (DPD) for liquid [4, 5], where the simulation molecule represents a number of atoms/molecules. The coarse-grained simulation can provide better computational cost, while the state-of-the-art modeling is required to correctly impose the wall boundary condition. The other is the use of traditional Molecular Dynamics (MD) solvers that describe the motion of individual molecules based on Newtonian dynamics. The hybrid CFD-MD model is preferred in solving dense liquid systems where the accurate description of strong inter-molecular interaction near the solid obstacle is very important.

Hybrid CFD-MD simulations can be classified to constrained Lagrangian dynamics [6–12], alternating Schwarz method [13–17], and direct flux exchange [18–23], according to which variables are exchanging and how the macroscopic solution is imposed on molecular domain. O’Connell and Thompson [6] first tried a hybrid CFD-MD simulation by applying constrained Lagrangian dynamics in the overlapping region between CFD and MD domains. In this method, two solvers exchange density properties (in other words, conserved variables) in the overlapping boundary and they are coupled in time space. Constrained Lagrangian dynamics equation is used to accelerate/decelerate particles in hybrid MD boundary to follow the continuum velocity, without harming the degree-of-freedom of molecular motion. This approach is refined by Nie et al. [7] by including the mass flux modeling and has been applied to the sudden-start Couette flow [6], channel flow with rough walls [7], cavity flow [8], Poiseuille flow [11] and the oscillating boundary problem [10, 12]. The use of a straightforward constrained dynamics equation makes this approach easy to implement and computationally efficient. However, the absence of energy exchange modeling limits the applications to isothermal systems. [18] Hadjiconstantinou and his collaborators [13–15] proposed another hybrid simulation approach which is based on alternating Schwarz method. In this approach, continuum solution is imposed on particle domain by using Maxwellian distribution function [14] or Chapman-Enskog distribution function [2]. CFD and MD solvers are evolving in individual time space until solutions in the overlapping region become identical. This approach is further expanded to non-periodic boundary condition problems by Werder et al. [16], who also refined previous boundary force models [6, 7, 18, 20] by minimizing the local disturbance. Alternating Schwarz approach has been applied to moving contact line problem [14], Poiseuille flow [15], flow around the cylinder [16], and Couette flow of water [17]. The characteristics of decoupled time space between two domains makes this approach better in solving the flow field where hydrodynamic characteristic timescale is much larger than molecular dynamic timescale, i.e., micrometer system size. [14] However, the feature of decoupling in time space limits this approach to steady or quasi-steady flow simulations. [18]

The direct flux exchange approach exchanges flux properties along the interface of CFD and MD domains at the same physical time. Flekkøy et al. [18] first proposed a model for solving an isothermal flow and the model is improved by considering energy transfer [19, 20]. Also, Delgado-Buscalioni and Coveney [21] designed a particle insertion algorithm which satisfies the mass flux along the interface while preserving the mean potential energy of a system. This approach has been applied to solving Couette and Poiseulle flow [18], transversal wave [20], and oscillating boundary problem [22]. According to Flekkφy et al. [18], flux exchange directly implies adherence to the relevant conservation laws without the use of constitutive relations and equations of state (to maintain the conservation laws). However, it is pointed out that the sampling time to measure fluxes within acceptable statistical error is orders of magnitude larger than the time to measure densities [15].

* 1. *Numerical and Computational Issues*

Despite the clear advantage of hybrid approach over conventional CFD or MD implementations, a number of numerical and computational diﬃculties prohibit this technique from being widely used. The first issue is an easy and intuitive way of determining the state-of-the-art coupling condition. The solution of a hybrid simulation by the crude coupling suffers from a strong noise of spatially averaged particle velocity, which is the response of a natural physics of molecular fluctuation. So, coupling parameters such as layer size and the position of overlap region, and sampling duration with its interval, should be delicately determined to achieve an accurate numerical solution. It is impossible to design a mathematical model to determine coupling conditions, since the strength of this uninvited noise is affected by not only the flow condition but also the characteristics of fluid and solid elements along with its geometric configuration. There have been some attempts to alleviate this statistic fluctuation by introducing numerical damping terms [6, 9] or dynamic parameter for coupling intensity [10], but this can break up the conservation law.

The next issue is the hybrid simulation in moderate flow conditions. So far, all experiments have been confined to the extremely high-speed flow conditions of O(100) m/s in the nano-scale systems, to maintain a strong shear rate/speed in the overlapping region. This unrealistic flow condition is unavoidable according to the mathematical analyses on the strength of statistical error (in other words, a signal-to-noise ratio) [15, 22]. Sampling duration to maintain the same statistical error is proportional to 1/u∞ 2 and 1/N0, where u∞ is a free stream condition and N0 denotes the number of particles contained in sampling layer. So, reducing the free stream velocity by 1/10 requires 100 times more particles to be averaged to maintain the same statistical error. (Increasing the sampling duration is avoided because it can fail to describe the unsteady flow evolution.) This necessitates a different design of a hybrid simulation framework.

The accuracy of temporal coupling scheme with boundary condition imposition is of another importance in solving the unsteady problems. Temporal coupling scheme denotes the scheduling of data exchange routine in time space between CFD and MD solvers, to synchronize both solutions at the same physical time. As to be discussed in Section 2.5 in detail, initial models inherently contain the time-lagging phenomenon because backward-averaged molecular dynamic properties is adopted as the instantaneous solution at continuum domain. As the remedy, Liu et al. [12] proposed a multi-timescale algorithm by considering quasi-steadiness and Wang and He [10] proposed passing the extrapolated continuum solution to the MD solver. However, a multi-timescale algorithm is only valid if the period of unsteady phenomenon is sufficiently longer than sampling duration and particles lose their history during the re-initialization process in between sampling interval: Approach by Wang and He is a possible approach to mitigate the time-lagging effect though not in perfection.

In addition to above numerical issues, there also exist computational challenges of integrating multiple application domains into a single problem set. Considering very different computational kernels (one could be mesh-based, the other unstructured particle simulations), it is nearly impossible to incorporate distinct CFD and MD codes under the umbrella of a single tightly-coupled application (i.e. unifying two application codes to share a single MPI communicator). One possible alternative will be to implement coupling interface on individual code and control these logically separated codes as a virtually unified simulation package.

However, confined to parallel execution on conventional production system with batch queue, it cannot be guaranteed that two separate jobs will execute concurrently. Considering the computational characteristics of current application, where CFD and MD codes conduct frequent information exchange, the first job to run will inevitably experience the idle waiting for its counterpart without the explicit support for co-scheduling. Another important challenge is the need for eﬃcient load-balancing which takes into account the individual application’s performance. Even if the two simulations could run concurrently, without explicit load-management/balancing support, there is likely to be ineﬃcient utilization of compute resources due to load imbalance. As the performance of each simulation component changes with computing resource and problem size, re-adjustment of allocated resources to each task according to their performance is required during the simulation.

* 1. *Objectives and Outline of the Paper*

We are focusing on investigating numerical issues of applying the hybrid CFD-MD scheme to a nano-fluidic system, as well as designing and developing an eﬃcient runtime framework for a coupled multi-component simulation. We consider the implementation of a ’generic’ hybrid interface which can be easily attached to various kinds of incompressible CFD codes and MD packages, and the building of a ’portable’ framework which is acceptable for most computer architectures.

Regarding numerical simulation, we develop the hybrid simulation framework based on the constrained Lagrangian dynamics and apply it to prototype examples. We propose the determination of coupling parameters based on the strength of statistical noise in stationary flow. Also we explore to the moderate-speed flow simulation in O(10) m/s velocity, which is the first successful analysis by a hybrid approach. Excessive statistical noise in this flow condition is shrunk by sampling multiple individual experiments. For physically unsteady flow simulation, we design a novel temporal coupling scheme, named ’prediction-correction approach’. Compared to conventional scheme, our approach provides better accuracy by eliminating the overshoot/undershoot phenomena and diminishing the time-lagging pattern.

In view of computational science, we believe our trial is the first documented coupled multi-physics simulation utilizing a virtually unified simulation package, called ”Pilot-Job”. We claim that there are several distinct advantages of using Pilot-Job: (i) obviates the need for a co-scheduler while preserving performance, (ii) enables dynamic resource allocation, which in turn is important for load-balancing across coupled simulations.

We begin the next section with the numerical details of individual CFD and MD code and implementation of hybrid interface. The construction of an efficient CFD-MD simulation framework will be discussed in Section 3. In Section 4, we will demonstrate our numerical results of a famous validation problem (a sudden-start Couette flow) and the physically unsteady flow simulation (oscillating boundary problem). The performance of this coupled simulation is described in Section 5. Finally, our next plan and the summary of current work are expressed in Sections 6 and 7.

1. **Fundamentals of the Hybrid Coupled CFD-MD Simulation Toolkit**

The accuracy of a hybrid CFD-MD solution is governed by the underlying numerical schemes of the individual continuum and particle solvers as well as the coupling scheme. This section details and explains the features of baseline solvers and the structure of the hybrid coupling interface. Additional numerical treatments to improve the accuracy of hybrid CFD-MD simulations in various types of fluid systems are also addressed.

* 1. *Overview of the Continuum-Molecular Coupled Approach*

The hybrid CFD-MD approach stands on the proposition that (a) the molecular dynamic simulation produces a more physically accurate solution in its domain than the continuum model would, (b) a faster and more efficient CFD solver provides a physically accurate solution in the continuum domain, which would be prohibitive if a particle representation were to be used, and (c) the two solutions can be reconciled by the mutual exchange of information, the continuum solution providing the constraint to the particle solution while the latter providing a boundary condition to the former. So, the region within which molecular-level physics are important (e.g. a material interface) is treated and resolved by a molecular dynamic simulation, while the region within which the continuum-level physics adequately represent the transport processes on a macroscopic scale is treated and resolved by the continuum CFD. The continuum and particle (molecular) domains overlap, in this approach, so as to exchange information for the coupling. Obviously, the overlap domain must be one where both treatments/formulations are valid.

A detailed structure of the various domains associated with the hybrid CFD/MD methodology is described in Fig. 1. CFD solves the continuum equations in the “Pure CFD Region”, while MD resolves the microscopic (particle) flow in the “Pure MD region”. The latter is a solid stationary wall for the two test problems examined herein. The hybrid (overlap) region is placed sufficiently far from the solid wall to prevent direct constraint of the molecular-level physics in the proximity of the solid-fluid interface.



Figure 1: Schematic Diagram of the Hybrid Domain with Detailed View of Overlapping Zone; Overall continuum/atomistic computational domain including overlap region is shown on left figure. Detailed layer by layer explanation of overlapping region is indicated by right figure.

This overlap region is designed sufficiently large to contain five individual layers of sufficient width for the purpose of their existence. The widths of these layers are of course problem dependent. Their roles in the hybrid region are as follows. One layer (the bottom one in Fig. 1-left) provides information from the particle (MD) domain to the continuum one (denoted as ’MDtoCFD’). A second layer (the third one from the bottom in Fig. 1-left) provides information from the continuum domain to the particle one (denoted as ’CFDtoMD’). A third layer (the top one in Fig. 1-left) applies a fictitious external force which prevents particles from escaping from the hybrid domain. These three “active” layers are separated by buffer layers, which are placed so as to ensure smooth transitions and that there is no direct correlation between the “active” layers. Properties (e.g. velocity) of particles spatially located in the MDtoCFD layer are ensemble-averaged over all the particles in the layer and also averaged over a finite time (’sampling duration’). The average value is communicated to the continuum solver periodically with a fixed period (’sampling interval’).

The height of each layer is the same as the CFD cell height and averaged conservative properties in two consecutive layers are passed to continuum domain to be directly imposed as the viscous boundary condition on the Navier-Stokes solver with collocated data structure. The CFDtoMD layer imposes the instantaneous values of properties, velocity in this case, resulting from the continuum solver. This is done via and appropriate constraint to the MD particle-based conservation of momentum on every single particle (constrained dynamics) in the CFDtoMD layer. Thus, particles in this layer are constrained to attain the macroscopic flow property (velocity in this case) on average, while preserving their degree-of-freedom of translational motion. In the uppermost layer, a fictitious external force is exerted on particles to prevent them from escaping the particle domain. This force function is designed to be short-range so as not to be strong enough to influence the motion of the particles past the buffer layer in the CFDtoMD domain. The force stiffens as the particles approach the location where the force becomes infinite in a way that minimizes reflections while preventing the particles from drifting out of the particle domain. The buffer layers existing in between each “active” layer are set up to be wider than the interaction length scale of the particles (cutoff radius), in order to prevent direct interaction between particles in neighboring “active” layers.

2.2 *Governing Equations and Numerical Schemes*

*2.2.1. Continuum Incompressible Flow Formulation (CFD)*

The current in-house continuum hydrodynamics code solves the unsteady incompressible Navier-Stokes equations to demonstrate the isothermal nano-scale flow field:



where *ν* is the kinematic viscosity.

In this work, we adopted the pseudo-compressibility method [24] is adopted to form a hyperbolic system of equations which can be marched in pseudo-time. A time derivative of pressure is added to the continuity equation resulting in



where β denotes a pseudo-compressibility parameter, currently set to 2.5.

For time-accurate unsteady simulation, a dual time stepping method is adopted and it is combined with the LU-SGS (Lower-Upper Symmetric Gauss-Seidel) scheme [25] for the implicit time integration. The inviscid fluxes are upwind-differenced using Osher’s flux-difference splitting scheme [26]. For higher-order spatial accuracy, the MUSCL (Monotone Upstream-centered Schemes for Conservation Laws) [27] approach is used on an inviscid flux calculation. Viscous fluxes are calculated using the conventional second-order central differencing.

* + 1. *Molecular-Level Formulation (MD)*

In MD, an initial velocity is assigned to each atom, and Newton’s laws are employed at the atomic level to propagate the system’s motion through time evolution. To calculate pairwise interactions of particles in the system, the most commonly used Lennard-Jones (12-6) potential interaction model is employed and is define as:



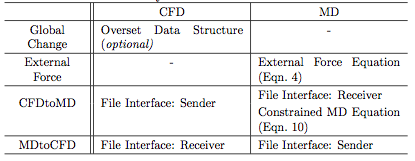
where єij and σij denote the pairwise potential well depth and the atom size parameter respectively, and rij is the distance between the particle i and j . The term 1 /r12 dominating at short range distance repulsive behavior ij based on the Pauli principle to avoid overlapping the electronic clouds when particles are brought very close to each other. The term 1 /r6 dominates at ij long range attractive forces by Van der Waals dispersion forces. The cut-off distance σc is introduced here to reduce the computational cost and is set to be 2.2σ [28]. Namely when rij exceeds the cutoff the intermolecular force is set to zero without being calculated.

The time integration algorithm is required to integrate the equation of motion of the interacting particles and computing molecular trajectories, one of most common velocity Verlet algorithm is employed to compute the simulation.

In this work, the MD simulations were performed by using an appropriately modified version of the Large Atomic Molecular Massively Parallel Simulator(LAMMPS). It is a classical molecular dynamics open-source code written in C++ and developed by Sandia National Labs. [29]

Table 1: Implementation of Hybrid Interface on CFD and MD Codes. Both codes are equipped with the ﬁle-based information exchange routine, to update the hybrid boundary condition. CFD code experiences the global change of its data structure to store the information of the entire ﬂuid system. MD code adopts

hybrid equations to impose the macroscopic information on microscopic domain and to ensure numerical stability.



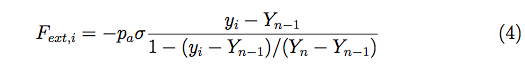
* 1. *Hybrid Interfaces and Schemes*

The hybrid simulation requires the implementation of hybrid interfaces and schemes on individual code. In the current study, the file interface is designed to schedule the information exchange between continuum and discrete particle descriptions. A constrained Lagrangian dynamics model is implemented for hybrid simulation. Unit conversion routine is also implemented in the application code. These changes are summarized in table 1.

The CFD code employs the data structure of overset mesh technique [30] to ease the handling of coupling parameters. In other words, the entire fluid domain is generated as the CFD mesh system and pure MD region is turned off as the `Hole' cell in the terminology of overset technique. Likewise, MDtoCFD and CFDtoMD boundary cells are declared as `Fringe' and `Donor' cells, respectively. The labor of mesh regeneration according to the change of coupling parameters (position and depth of hybrid layers) disappears with the overset data structure.

Both codes are equipped with the information exchange routine which consists of one file sender and one file receiver. These file interfaces are scheduled to turn on every sampling interval. The instantaneous properties in the Donor cells of continuum domain are transferred to MD site and referenced when applying constrained Lagrangian dynamics equation. Averaged molecular properties are sent to CFD domain and they are used as the boundary conditions of Fringe cells. All exchanged properties are written in MD unit: thus, CFD code is equipped with velocity unit conversion function and equation of state which changes the pressure solution from CFD site to equivalent density property in MD domain.

Along with the file interface, additional equations of motion are employed on MD code to accurately describe the influence of macroscopic flow variation on particle domain. First, the external force should be imposed to prevent leaving particles from the control domain and the force is applied to the normal direction of uppermost MD layer. A cost-effective classical external force model by [7] is employed as,



where *pa* denote the average pressure in the MD region, *Yn* - *Yn-1* is the thickness of the uppermost layer which is applied the force and *Fext* is the external force acting on *ith* particle located on position *Yi* .

Next, on CFDtoMD layer, the macroscopic flow properties at specific time shall be introduced to lead the motion of multiple particles in that layer. To satisfy mass conservation, a certain number of particles are inserted into or removed from this layer according to the mass flux by CFD solution,

:5.tiff

where *A* is the horizontal area, *uy* is the vertical velocity component by CFD solution and *Δt* is the sampling interval.

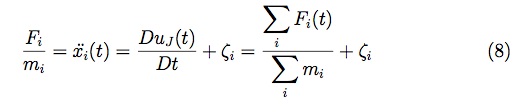
A very complicated numerical intervention is required to maintain momentum conservation. The average velocities of particles in *Jth* cell is equal to the velocity *uJ* in continuum cell.

:6.tiff

where *vi* is velocity of *ith* particle and *NJ* is the number of particles in the cell. With taking Lagrangian derivative of Eq. (6),



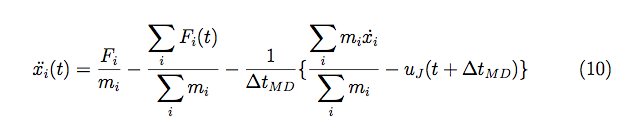
The Classical MD equation of motion can be generalized to obtain constraint by adopting the fluctuation in acceleration of each particles, *ζi*



where *Fi* is the force on *ith* particle based on the interactions between particles, *mi* is mass of each atom and Eqn. (9) satisfies,

:9.tiff

Finally, the constrained particle dynamics with conventional equation of motion can be written as:



The continuum velocity and the mean microscopic velocity from MD over control domain provide the synchronization of the mass and momentum consistent with Eqn. (10).

* 1. *Statistical Error and Coupling Parameters*

How to reduce the statistical error of averaged MD profile, which is the response of innate spatial/temporal locality in molecular dynamic systems, determines the accuracy of CFD solution in hybrid simulation. According to the mathematical expression in statistical error [15, 22], the ratio of sampling noise compared to the macroscopic velocity is inversely proportional to the square root of spatial layer size and temporal sampling duration. For example, reducing the macroscopic velocity by half requires either 4 times larger system domain or 4 times longer sampling to maintain the same order of accuracy.

Sampling duration with interval, the size of sampling layer and its position are coupling parameters which define the scale and pattern of this noise. The layer size and sampling duration collectively work for reducing the noise, by increasing the spatial and temporal sampling scales. Sampling interval is a factor which restricts the sampling duration. On unsteady simulations, a short sampling interval is preferred to frequently update temporal variation of flow field in hybrid boundary zones. This interval acts as the upper bound of sampling duration. The location of the sampling layer is a secondary factor which can locally increase the strength of fluctuation. Conventionally, sampling layer is placed far from the solid obstacle, e.g., at least 10σ above the bottom wall in solving the flow of the liquid argon [11].

We introduce two numerical ideas to acquire the accurate hybrid solution. One is to quantitatively measure the scale of statistical noise in particular domain and the other is to numerically get the acceptable solution. The former shows the initial guideline to determine the coupling parameter and the latter refines the accuracy of the former solution.

*2.4.1. Determining Coupling Conditions*

The statistical noise is a function of the characteristics of fluid and surrounding solid elements, and geometric configurations, as well as the flow condition.Unfortunately, previous analyses on the strength of statistical error [15, 22] fails to consider the stronger interaction near the fluid-solid interface and the shape of the domain. We sense that the clear coupling parameters can be determined after the look-up of that specific system

Our intuitive idea of numerically detecting the strength of statistical noise is to solve the stationary flow of the same fluid domain by pure MD method. The procedure is as follows.

1. Empirical design of the sampling interval and the sampling layer's position.[[1]](#footnote-1)

2. Structure construction.[[2]](#footnote-2)

3. Simulation and collecting the temporal history of sampled velocity.[[3]](#footnote-3)

4. Data processing.[[4]](#footnote-4)

5. Determining the sampling layer size and the sampling duration.[[5]](#footnote-5) [[6]](#footnote-6)

*2.4.2. Sampling Multiple Independent Experiments*

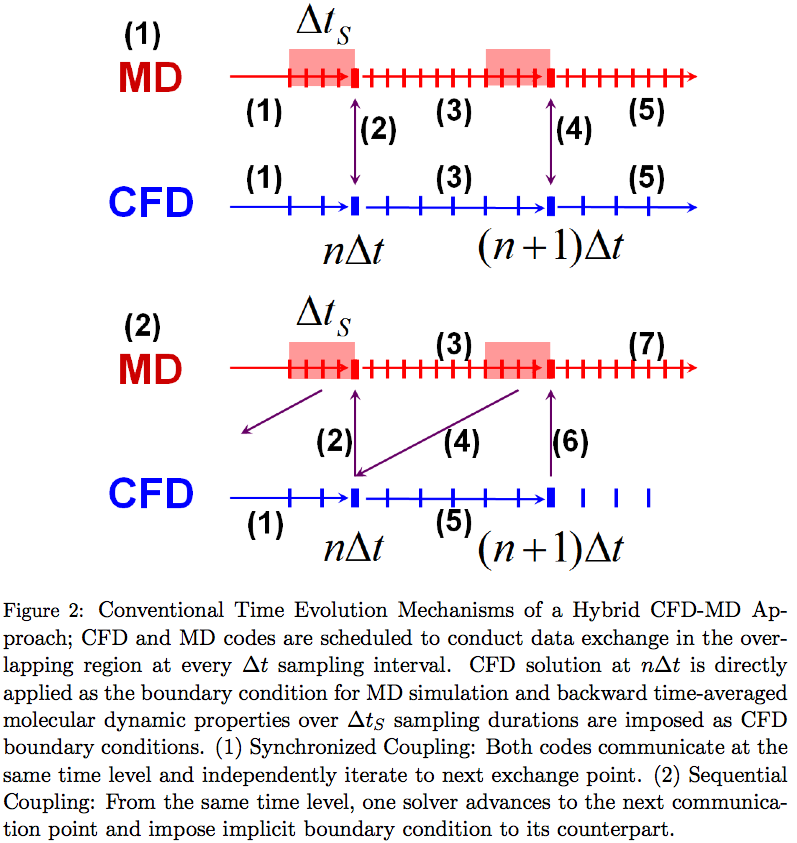
So far, all hybrid CFD-MD applications have been restricted to extremely fast flow field of O(100) m/s velocity.The difficulty of solving moderate-speed flow field by a hybrid approach is explained as follows. The hydrodynamic time scale is expressed as a function of characteristic size and kinematic viscosity. This implies that the sampling interval is fixed regardless of the change in velocity. This, in turn, results in the impossibility to handle the sampling duration. Increasing the system size proportional to the square of the velocity change remains the only possible way to maintain the same statistical accuracy. Unfortunately, submitting excessively large-scaled simulation on public supercomputers unfavorably takes far long time to get allocated. In worse case, the simulation may exceed the capacity of the system. Thus, a different approach is necessary for efficiently simulating the low-speed flow field.

We propose sampling multiple independent hybrid simulations from a smaller domain instead of trying to run a single hybrid simulation in the large domain. The initial velocity component at individual problem set is differently determined from a Maxwell-Boltzmann distribution and solutions from independent simulations are averaged to produce the final solution. The labour of manually administrating multiple job executions are resolved by using a BigJob framework, which is to be discussed in the Section 3.

Along with the advantage in computing capability, sampling multiple experiments provides another advantage of less-sensitivity to coupling conditions. Even if the coupling parameters are ill-chosen, highly noisy individual solution can be refined by the enough number of independent samples. This advantage also increases the value of capturing the magnitude of statistical noise in Section 2.4.1., because the requirement of predicting the macroscopic velocity in the sampled layer becomes less important.

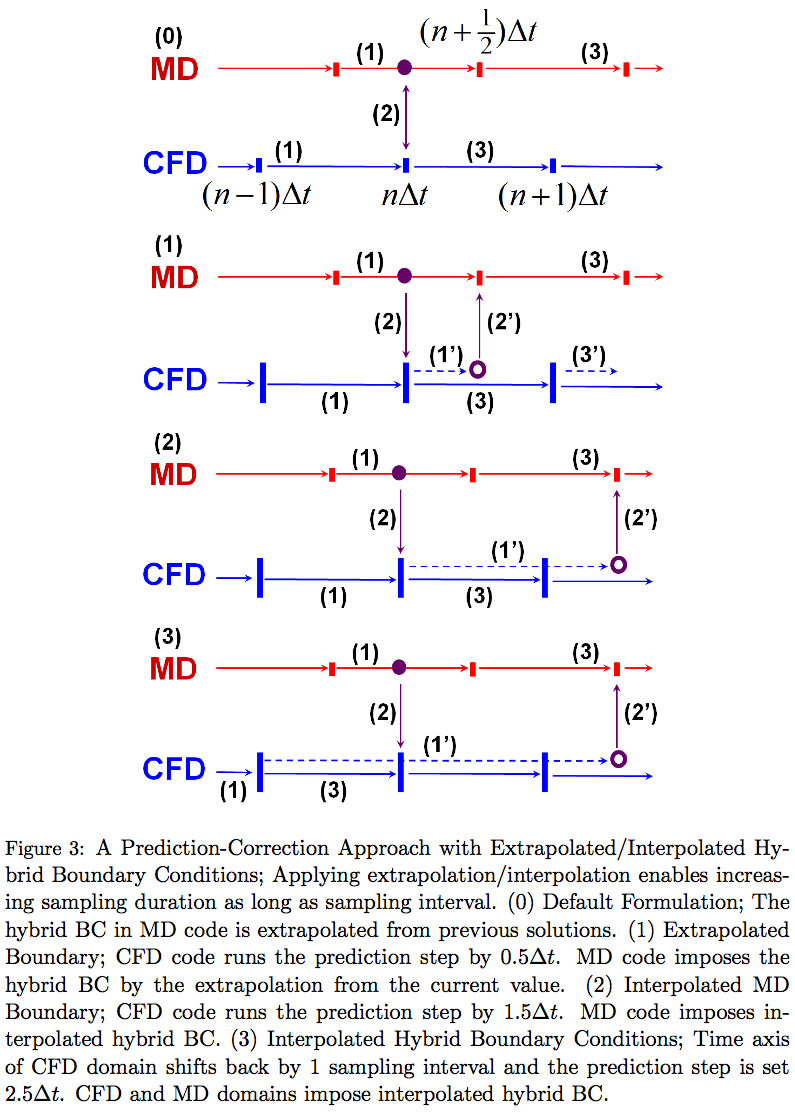
* 1. *Temporal Coupling Schemes*

Two conventionally used coupling approaches of synchronized and sequential coupling strategies are depicted in Fig. 2. In synchronized coupling, CFD and MD codes exchange the information in the overlapping region at each sampling interval (denoted by *Δt*) and independently evolve to the next exchange time step. In contrast, one domain advances to the next exchange time step and leads its counterpart to approach this point in sequential coupling. Comparing these two mechanisms, synchronized coupling has a far better parallel efficiency because both codes independently evolve to the next sampling interval, while sequential approach is expected to produce more time-accurate solution because the boundary condition on following domain (CFD domain in the figure) is implicitly imposed. Unfortunately, both strategies contain the time-lagging phenomenon in CFD boundary region, because averaged molecular dynamic properties over backward sampling duration (from *nΔt* - *nΔts* to *nΔt*) represents the CFD boundary condition at that instance (*nΔt*). Extrapolation from previous MD solutions is necessary to eliminate the time-lagging by half of sampling duration (*nΔts* /2) in the CFD boundary zone.



As is presented in Fig. 3-(0), Wang and He [10] proposed shifting the time axis of one domain by half of the sampling interval to eliminate this lagging effect. After both codes evolve by a sampling interval, the instantaneous CFD solution at *nΔt* is communicated with averaged MD solution from *(n-1/2)Δt* to *(n+1/2)Δt*. Two previous solutions from the counterpart are extrapolated to impose hybrid boundary conditions. The benefit of imposing extrapolated boundary condition is that the sampling duration can be designed as long as the sampling interval (*nΔts* = *nΔt*). However, the accuracy of extrapolated solution is worth to debate. In this scheduling, two previous CFD solutions at *(n-1)Δt* and *nΔt* are extrapolated to produce MD boundary conditions from *(n+1/2)Δt* to *(n+3/2)Δt*. Except the velocity gradient is linear in time space, extrapolated properties fail to predict correct values throughout the simulation interval (from$(n+1/2){\Delta}t$ to $(n+3/2){\Delta}t$). The only way to reduce this extrapolation error is to reduce the sampling interval, which is contradictory to the condition for reducing the statistical error.

A new scheme named `prediction-correction approach' is also depicted in Fig. 3. The main difference from the default model is that CFD code iterates additional time steps after the code evolved to the next data exchange time. For example, in Fig. 3-(1), CFD code additionally evolves by the half of sampling interval after the code approached to *nΔt*. CFD code sends these predicted properties at *(n+1/2)Δt* to the MD site and receives averaged molecular properties around *nΔt*. CFD code loads its previous flow profile at *nΔt* and runs the actual simulation to next communication point at *(n+1)Δt*. Clear benefit of current approach is that both solvers extrapolate their boundary conditions from current solutions (either exact solution or predicted values) instead of using previous history. This eliminates the sensitivity of extrapolated solution according to the size of sampling interval in previous model, which enables increasing the sampling interval.



This approach is further refined to increase the accuracy of the hybrid boundary condition by increasing prediction time scale. In Fig. 3-(2), the prediction time scale is increased by one more sampling interval. While MD solver evolves for one sampling interval, CFD code iterates to the next communication point of MD time space. This enables MD boundary condition being interpolated by predicted CFD profiles. Figure 3-(3) demonstrates the imposition of interpolated boundary conditions on both domains. In this formulation, CFD time space is shifted backward by one sampling interval and prediction step is scheduled to be *(5/2)Δt*.

Current numerical approach provides more accurate time-variant solution by decreasing or eliminating the unfavorable overshoot/undershoot phenomena in extrapolations. Meanwhile, additional computational cost is inevitable for CFD simulation. We propose the current approach to be used in following conditions: (i) computational cost on CFD is quite smaller than that of MD, and (ii) the driving force which causes the flow variation is provided from the CFD domain. Without condition (i), additional computational overhead for prediction process will harm the simulation performance. If (ii) is not satisfied, the pattern of flow evolution cannot be predicted and the accuracy of the predicted solution is not guaranteed.

1. *Coupled Concurrent Multi-Scale (Continuum-Molecular) Simulation Framework*

Two important issues on the performance of hybrid simulations are co-scheduling and load-balancing. Both can be resolved by adopting a Pilot-job concept. We explain the design of a multi-physics simulation framework which operates in the form of a single Pilot-job and contains a load-balancing function between distinct tasks.

*3.1. SAGA and SAGA-based Frameworks - An Efficient Runtime Environment for Coupled Multi-component Computations*

The Simple API for Grid Applications (SAGA) is an API standardization effort within the Open Grid Forum (OGF) [31], an international standards development body concerned primarily with standards for distributed computing. SAGA provides a simple, POSIX-style API to the most common Grid functions at a sufficiently high-level of abstraction so as to be independent of the diverse and dynamic Grid environments. The SAGA specification defines interfaces for the most common Grid-programming functions grouped as a set of functional packages (Fig. 4). Some key packages are:

- File package - provides methods for accessing local and remote filesystems, browsing directories, moving, copying, and deleting files, setting access permissions, as well as zero-copy reading and writing

- Job package - provides methods for describing, submitting, monitoring, and controlling local and remote jobs. Many parts of this package were derived from the largely adopted DRMAA

- Stream package - provides methods for authenticated local and remote socket connections with hooks to support authorization and encryption schemes.

* Other Packages, such as the RPC (remote procedure call) and Replica package

The BigJob [32] is a SAGA-based Pilot-Job, where a number of sub-tasks can run in a pre-defined schedule with the specified number of processors whether or not they are coupled. We basically devise this solution to overcome the concurrent scheduling requirement of coupled CFD and MD jobs and to dynamically allocate resources for load-balancing of these codes. The advantage of a BigJob over other Pilot-Job implementations is that this is infrastructure-neutral, thanks to various adaptors in SAGA.

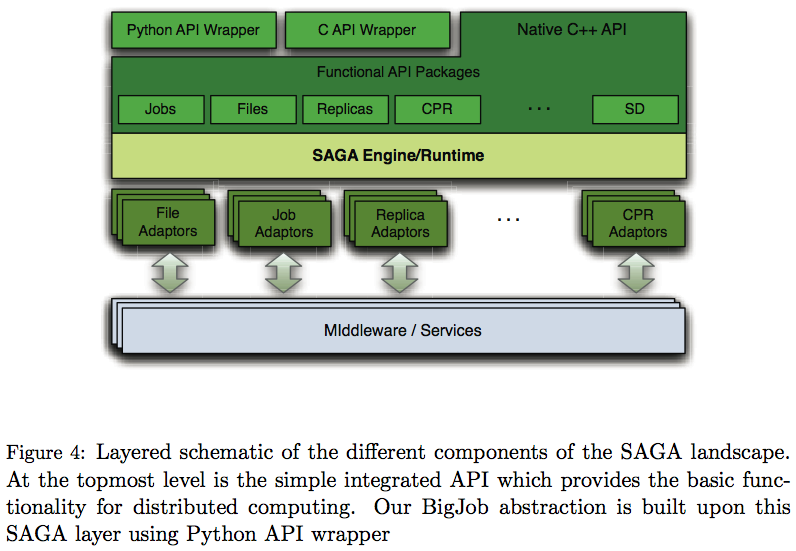
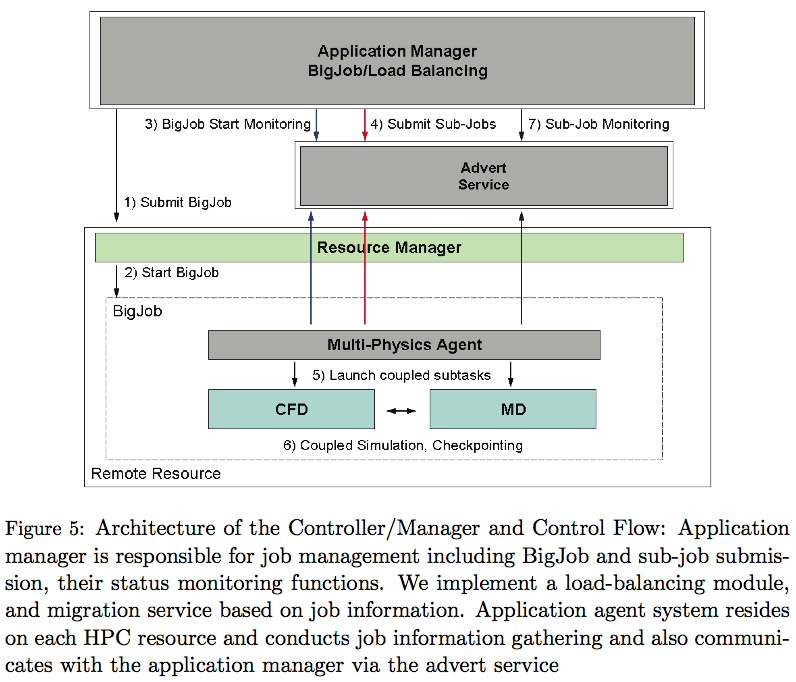


Fig. 5 shows the structure of BigJob and its operation flow. When a BigJob is submitted to the remote resource, the application manager monitors the status of this Pilot-Job through the advert service. When resources are allocated to the BigJob, the application manager allots the obtained resources to its sub-jobs and a coupled simulation starts under the control of a multi-physics agent in the remote resource. Advert service keeps on getting the status of a Pilot-Job from the queuing system and the status of sub-jobs from multi-physics agent. It also delivers this information to the application manager by a push-pull mechanism. The application manager watches the status of sub-jobs and decides the next event when the coupled simulation is finished. If an individual simulation is of interest, the manager closes a BigJob allocation when the simulation is finished: In cases of multiple replica simulations or load-balanced coupled simulation, the manager relaunches sub-jobs on the same BigJob allocation until all replicas or simulation loops are completed.

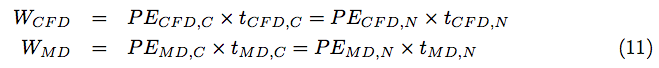


*3.2. Load-Balancing of Coupled Multi-Physics Simulation*

Load-balancing of a coupled simulation implies the flexibility to re-distribute resources to the individual task according to the performance of individual job. We will discuss the implementation and algorithm of a simple load balancer (LB) [33]; it is important to mention that the LB functions in the context of the SAGA-BigJob framework.

The idea is to assign more resources to heavier sub-jobs under the fixed resource allocation, until all sub-jobs elapse the same execution time. As it is impossible to predict the performance of each code in advance, we let the LB monitor the wall-clock time between information exchange points of coupled tasks and iteratively change the processor distribution until the load-balancing is achieved. As the individual solver is considered as black-box, each application code is assumed to have the ideal parallel efficiency: In case application codes are highly scalable, the LB can find the best condition after a few dynamic re-distributions. Also, all processors in one node are assigned to one single task to prevent the interference (and performance degradation) observed when multiple MPI tasks share the node.

Let the computation time (between exchanges) of the two sub-jobs be *tCFD* and *tMD* , and the number of processors assigned to each domain be *PECFD* and *PEMD* , respectively. Subscripts C and N denotes current and next states. Assuming ideal parallel efficiency, total load of each application remains the same after resource re-allocation,



In spite of the re-allocation, the total number of processors utilized remains the same:

:12.tiff

Our objective is to reduce the computation time of a sub-job to the point until the two application components show the same computation between the exchange points, i.e., *tCFD, N* = *tMD, N* . From Eqn. 11. and Eqn. 12. an optimal number of processors distributed for the CFD subtask would be:

:13.tiff

The MD simulation (sub-job) will follow a similar expression.

The above non-integer value proceed in discrete values expressed as the multiples of the number of CPU cores in a node. We choose the nearest discrete number to our load-balancing as the optimal number of processor on each application.

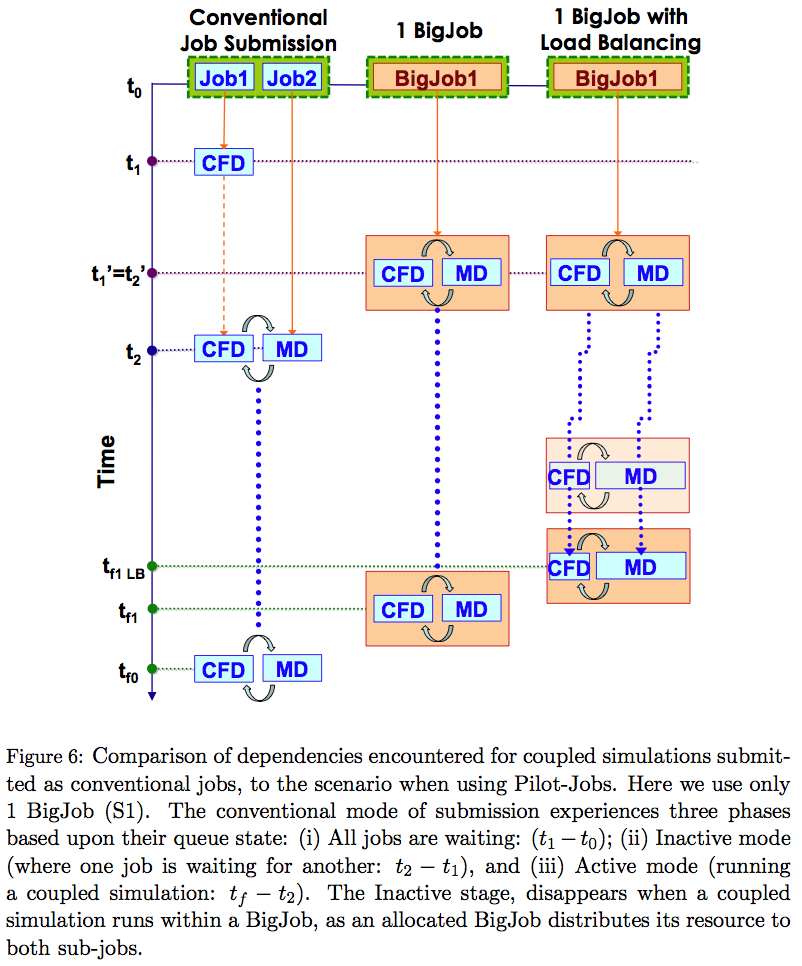
* 1. *Implementation of an Execution Framework and Application-level Corrections*

A hybrid CFD-MD framework is evaluated by implementing an application manager in Fig. 5., which is written in PYTHON script language. By default, an application manager calls a number of SAGA functions in sequence, to get allocated a vacant job, to run individual MPI simulation, to monitor its status, and to finalize the BigJob allocation.

In case the load-balancing capability is turned on, the situation becomes complicated. A single MPI job is not able to change its number of processors during the simulation. This implies that coupled codes should stop-and-restart to get assigned with changed number of processors. Thus, sub-jobs are scheduled to have multiple restarts from the previous checkpointing solution, which we denote 'simulation loop'. A LB is provided as a separate function in an application manager and is scheduled to run in between each restart of sub-jobs.

The efficient functioning of the LB is predicated on application codes being able to restart from their checkpointing data effectively. Application codes should also be equipped with generalized domain partitioning routine to run a simulation with any number of processors, without harming their parallel efficiency a lot. Another change implemented on application codes is the time checking routine. The runtime of each application is meaningless in running a LB since this runtime contains idle waiting on inter-domain information exchange as well as the individual simulation time. This actual runtime can be counted by putting two wall-time functions before and after the information exchange routine.

The generation of an application manager and the changes in application codes raise the possible simulation scenarios as given in Fig. 6. The first (leftmost) shows the time evolution of a coupled simulation under a conventional job submission (which we define to be scenario S0), and others using a BigJob (denoted as S1). For S0, individual tasks with resource requirements of *PECFD* and *PEMD* respectively, are independently submitted to the conventional queuing system and job scheduler recognizes these coupled tasks as two distinct jobs. Thus, they start at different times on average. In this case, both tasks wait on the queue when no job is allocated (waiting stage), the first allocated job idles to perform data exchange with its counterpart (idling stage), and the actual simulation starts when both jobs are allocated (running stage). On the other hand, for scenario S1, a BigJob of size *PECFD* + *PEMD* is submitted to the queue, and coupled simulation directly starts when the resource is assigned to this BigJob. Because of co-scheduling of sub-jobs, a BigJob is free from long inactive mode which is frequent in conventional job submission, while total runtime is the same if the resource distribution to sub-jobs is identical. However, eliminating inactive mode in itself does not guarantee a reduction in the total runtime, because a larger single allocation may result in a greater queue waiting time than two simulations requesting smaller number of processors each (but the total being the same). The same situation can arise for the load-balanced case with one BigJob (S1\lb). From the comparison between S1 and S0, we can estimate the performance gain by concurrent start of distinct coupled codes; S1\lb solution compared to other scenarios will demonstrate the benefit of a load-balancing function on coupled simulation.



1. *Multi-physics Flow Simulations in Various Flow Conditions*

A hybrid CFD-MD framework is employed to solve the multi-physics flow in nano-scale. Experimented problems are a sudden-start Couette flow and the oscillating boundary problem. We start from determining coupling parameters to the validation and verification of the hybrid simulation framework, then to the exploration of the moderate-speed flow simulation and time-accurate hybrid simulation.

*4.1. Problem Description and Coupling Conditions*

All applications we examine are internal flow fields filled with the liquid argon. Characteristic length of liquid argon is σ=3.405x10-10 and time scale is τ=2.2x10-12. Density is 0.81mσ-3, which means 0.81 atoms are included in the characteristic volume. The fluid domain is a channel system which consists of two parallel plates placed in vertical direction. Liquid argon particles are filled in the domain and both walls have artificial properties which is the same as those of liquid argon. The slip ratio between fluid and solid particles are set 0.6, to satisfy the linear velocity gradient along the vertical direction [7]. Channel is 52 σ in height, which is O(10-8) meters. Applications covered in this work are periodic systems in the perpendicular direction and the flow variation is derived by the horizontal motion of an upper plate.

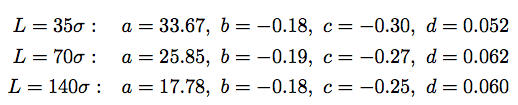
As has been proposed in Sec. 2.4.1., we solve a stationary flow in particle domain to determine coupling conditions. Table 2 shows the averaged velocity depending on the layer size and sampling duration in 10-8 meter domain. Experiments were conducted with different lengths of the domain, changing from 35 σ to 70 and 140 σ. Noises around 0.2 height above the wall are presented.

From the individual test, we find that mathematical expressions on the strength of noise according to the height of sampled layer and sampling duration [15, 22] do not coincide with our experiment. From the first table, increasing the height of averaged layer from 0.1 to 6.4 σ reduces the statistical error by 4 times when sampling duration is 1 τ and this ratio even gets worse as sampling duration increases. The same situation also happens on sampling duration. This is contradictory to previous mathematical expressions which has been introduced in Sec. 2.4. From the data analysis, we figure out that the magnitude of sampling noise depending on the layer size and sampling duration is far more complicated: *VN = a x SDb x LHc x SDd x ln(LH),* where *VN* represents the velocity magnitude of the noise, SD denotes sampling duration and LH is the height of the layer.

This equation is rewritten in simpler logarithmic formulation as,

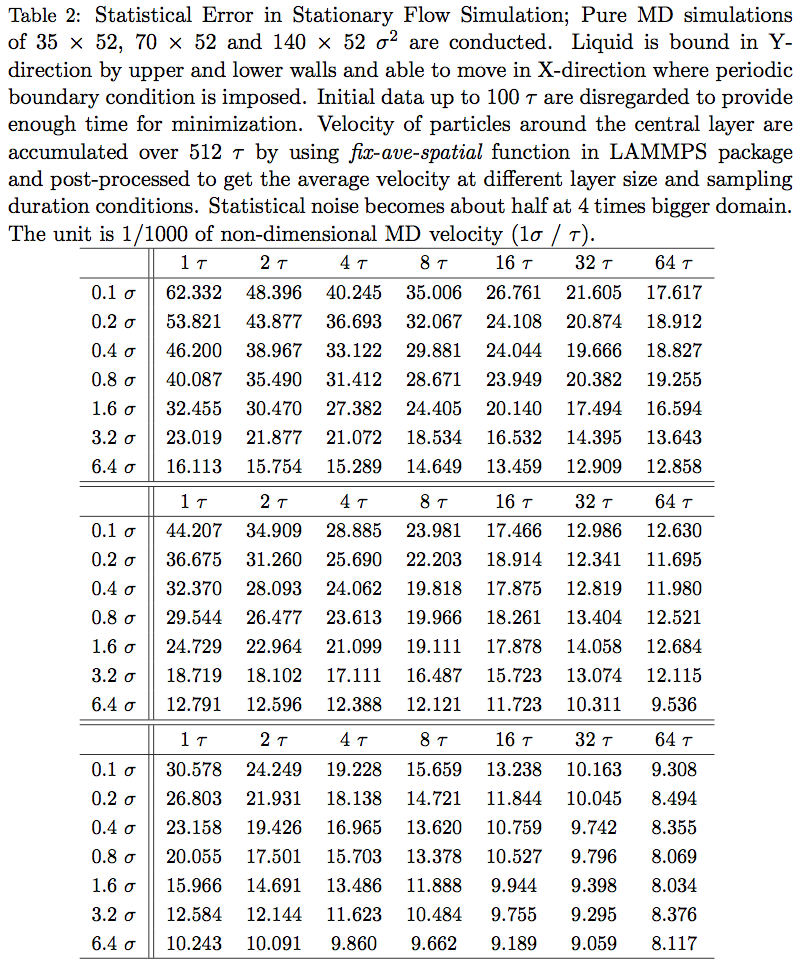
:eq36-1.tiff

and these coefficients in our specific case are as,



Another result we observe from the above mathematical expression is that increasing the system size into the periodic direction is more effective in reducing the noise. Setting SD and LH to 1 τ and 1 σ , we find that the magnitude of the noise reduces from 33.67 to 17.78 if system size is quadrupled. This result supports previous mathematical expressions on statistical error.

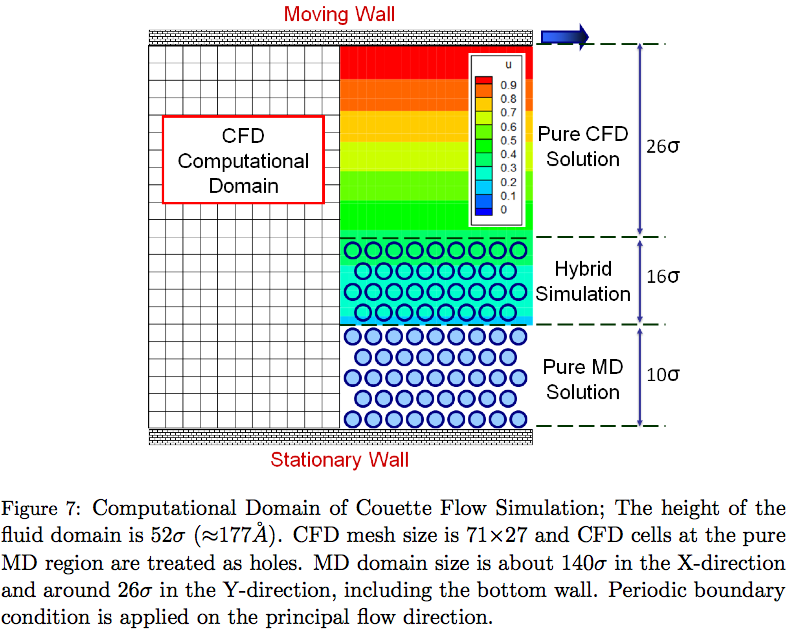
Following conclusions are deduced from the sampling noise analysis in the stationary flow. First, previous analyses on statistical error cannot be applied on wall-bounded nanoscale systems. As our empirical equation presents, the layer height and sampling duration are coupled together and noise reduction by handling these factors are less effective than have been reported. This emphasizes that the actual measurement of the sampling noise is inevitable to determine coupling parameters in the specific system. Second, increasing the size of sampling layer in the periodic direction is more effective on reducing the sampling noise. This leads us to an unfavorable conclusion that additional computational cost should be sacrificed to get the acceptable sampled solution.



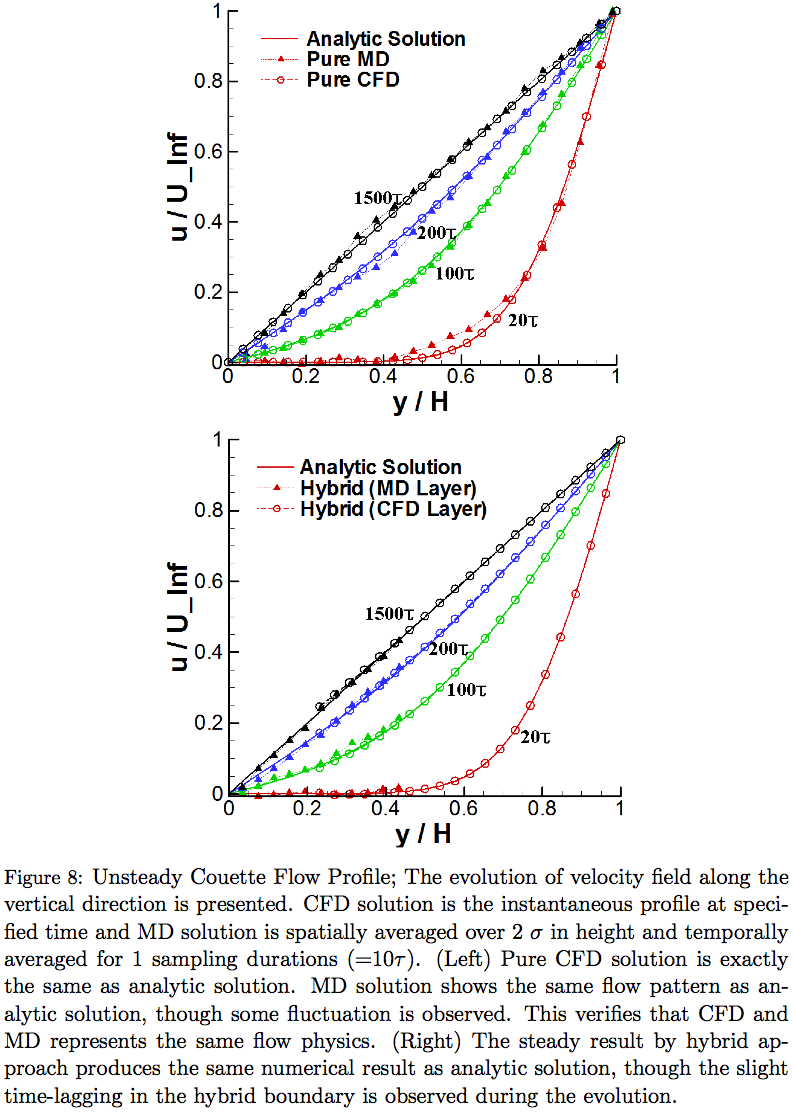
Finally, CFD and MD computational domains are generated based on the above experiment, as depicted in Fig. 7. Pure MD region is specified to be 10σ, which was reported to be sufficient to prevent strong fluctuation between fluid particles and wall materials from directly transported to CFD domains [11]. This implies that the steady-state velocity in the hybrid domain will be around 0.2 σ/τ. We design the strength of the statistical error not to exceed 5 percent (≅ 0.01 σ/τ) of steady-state velocity. This lead us to set the width of MD domain in the principal flow direction as 140 σ, the cell size of CFD mesh to be 2σ in Y-direction, and sampling duration to be 10 τ. Two layer boundary zones from particle to continuum domain is placed ahead of pure MD region, from 10 to 14σ. Two layers of continuum to particle boundary is positioned from 18 to 22σ and external force region is place at the top of hybrid region, from 24 to 26σ.

*4.2. A Sudden-start Couette Flow*

A sudden-start Couette flow is simulated to verify the accuracy of the current framework with multiple sampling approach for the moderate-speed flow simulation. This application has been in wide use for the validation of a hybrid CFD-MD solver. [7, 11] The flow is initially set stationary and the upper wall starts moving by a constant velocity (1 σ/τ). This physical boundary condition of the continuum domain governs the evolution of the whole flow field. Figure 8 presents a sudden-start Couette flow profile by CFD, MD and hybrid simulations. Pure CFD produces identically the same result as analytic solution and MD simulations also describes the same flow physics though the slight fluctuation of the solution is observed. This verifies the accuracy of the individual solver. The hybrid solution also shows the slight deviation from the analytic solution, which is the fluctuation of the sampled MD solution. Nevertheless, the hybrid simulation succeeds in demonstrating the same flow physics as the analytic solution. This variation can even be diminished if the solution is visualized over a longer temporal range, which is observed in many articles. This proves that the current hybrid framework accurately analyzes the steady flow profile in nano-scaled systems.



The flow condition in above experiment is rather unrealistic, which motivates us to apply the hybrid scheme to the analysis of moderate velocity flow. As has been expressed in Sec. 2.4.2., hybrid simulation of the low-speed flow field is mathematically possible but technically bound by the computing capacity, since the computational domain becomes u2 times bigger in solving 1/u velocity field. We instead run u2 independent simulations with the initial system size and different random number seeds in LAMMPS package.



The Couette flow profile with different numbers of samples are presented in Fig.~ 9. All configurations and parameters are identical to the above validation problem, except the upper plate velocity of 0.25 σ/τ. Changing the velocity to 1/4 implies that averaging 16 samples are required to get the acceptable numerical solution. The result supports the above supposition. The solution becomes very accurate when 16 individual runs are sampled. The reduction of statistical noise by multiple samplings is clearly verified by the graph in Fig. 10. The noise is roughly seen to reduce by half with 4 times more samples and the solution of sampling 16 runs shows about 5% of the noise compared to the analytic solution profile.

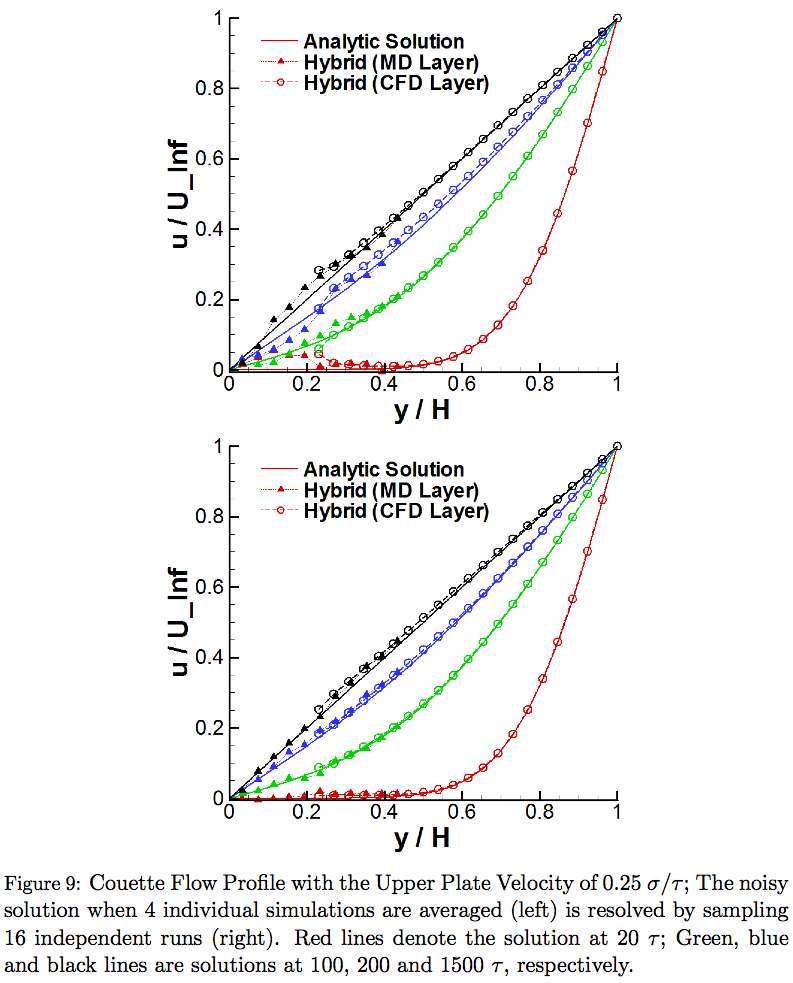
The solution by multiple sampling is compared with the solution in increased system domain. Figure 11 shows the Couette flow profile with 16 times bigger system size in horizontal direction and the comparison of velocity variation in the middle of the overlapping region. From the result, both ways (multiple sampling and increasing system size) produce acceptable numerical solutions compared to the analytic solution. Interestingly, the scale of the noise compared to the analytic solution is very similar in both ways, which verifies that multiple sampling approach can replace the increase of system size to reduce the statistical error.

Collectively, results by multiple samples show the same order of accuracy compared to the increase of the system size. Especially in cases the physical system is large, multiple sampling is more effective than directly increasing the system size, because excessively large-scaled jobs (in view of wall-time limit or number of resources requested) are very hard to get allocated. In addition, the labor of manually submitting multiple independent runs and managing data sets can be relieved by adopting a BigJob framework.

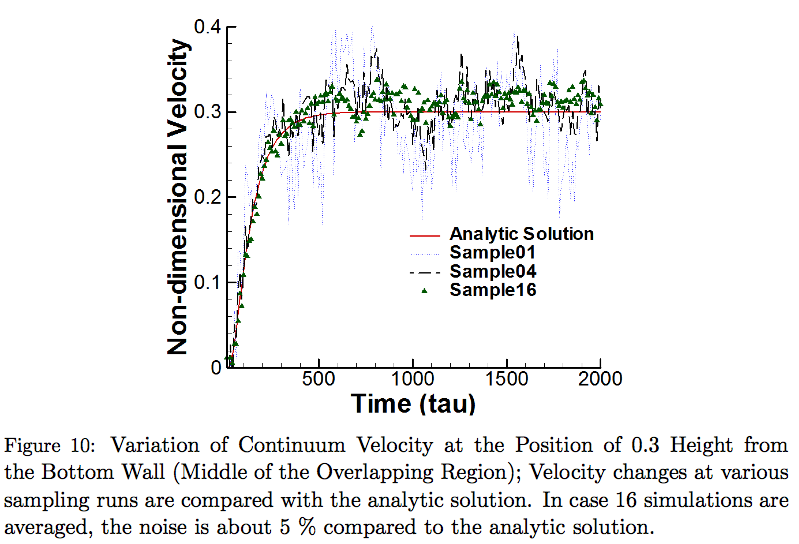
*4.3. A Physically Unsteady Flow Field: Oscillating Boundary Problem*

The accuracy of designed temporal coupling scheme is verified by solving an unsteady oscillating boundary problem. In Couette flow simulation, which converges to a steady-state flow profile, the minor inaccuracy during the flow evolution can be eventually recovered. On the other hand, the inaccurate solution at an one instance harms the flow field afterwards in this physically unsteady problem. Therefore, the accuracy of temporal coupling scheme becomes more important. Also, velocity in the hybrid region becomes far slower than the Couette flow profile, so that the influence of noise from MD side is concerned to be more critical in the current flow simulation. The computational domain and coupling conditions are the same as the above Couette flow simulation. In this case, the upper wall boundary condition changes from the fixed velocity to oscillatory wall, which is uwall(t) = (σ/τ) x sin(2πt/T). Period T is set to be 200τ.

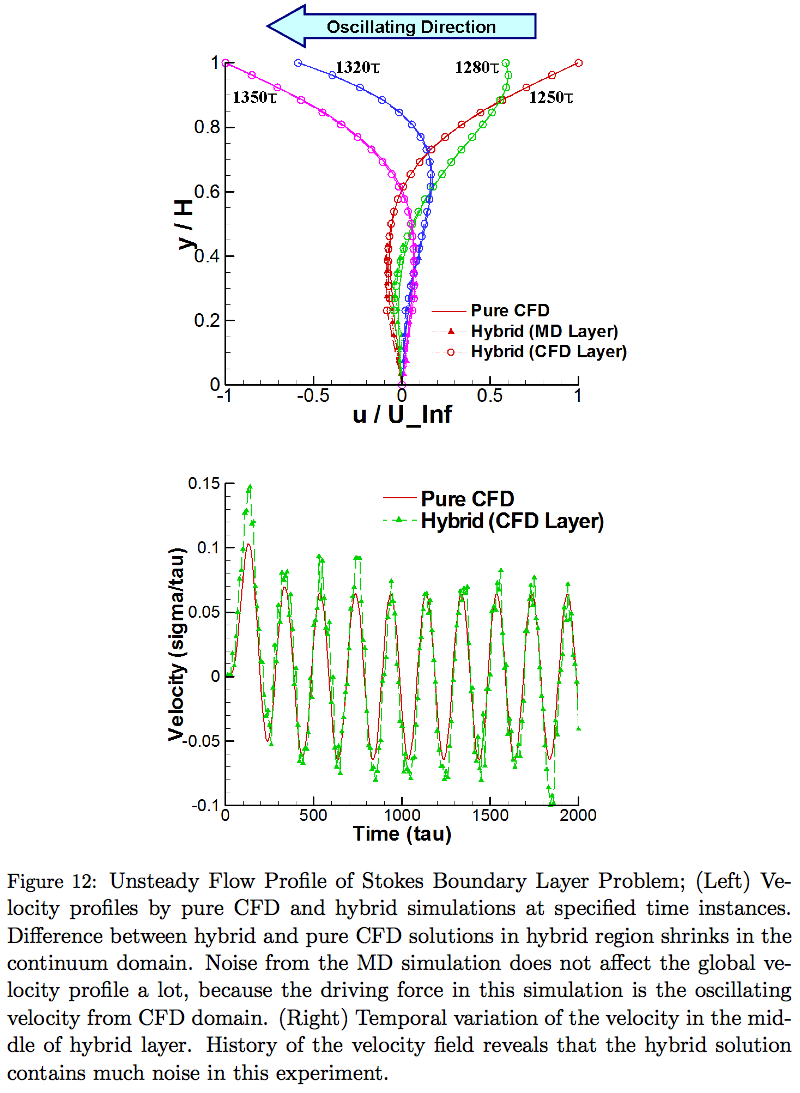
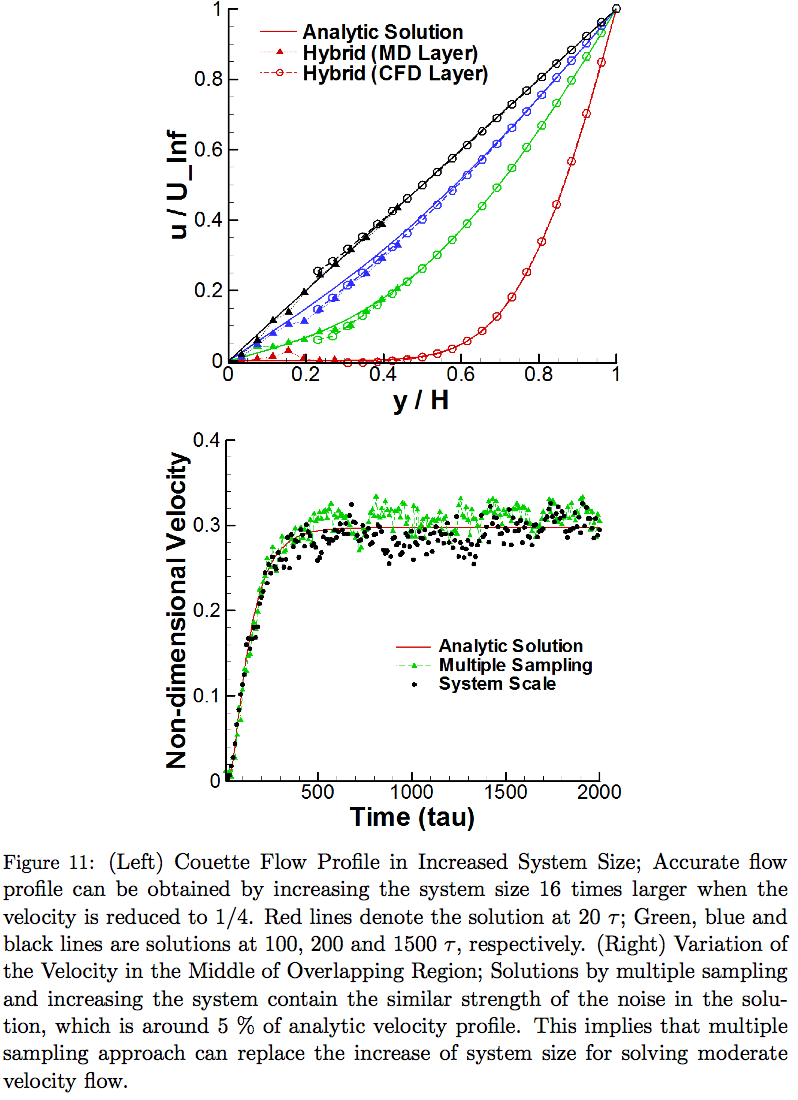
Figure 12 shows the oscillatory velocity profile by pure CFD and hybrid simulations. From the left figure, velocity profiles at each time instance are roughly the same between the pure CFD and hybrid simulations in magnitude. This provides that the hybrid simulation approach is also applicable to time-varying flow simulations. Meanwhile, the temporal variation of the horizontal velocity in the middle of hybrid region expresses that the noise in hybrid solution is not negligible considering the ratio between continuum and hybrid velocities. We claim that this noisy solution is caused by the combination of sampling error and inaccurate temporal coupling scheme. To clearly examine the effect of temporal coupling scheme, we sampled multiple independent experiments and compared solutions by different temporal coupling schemes.

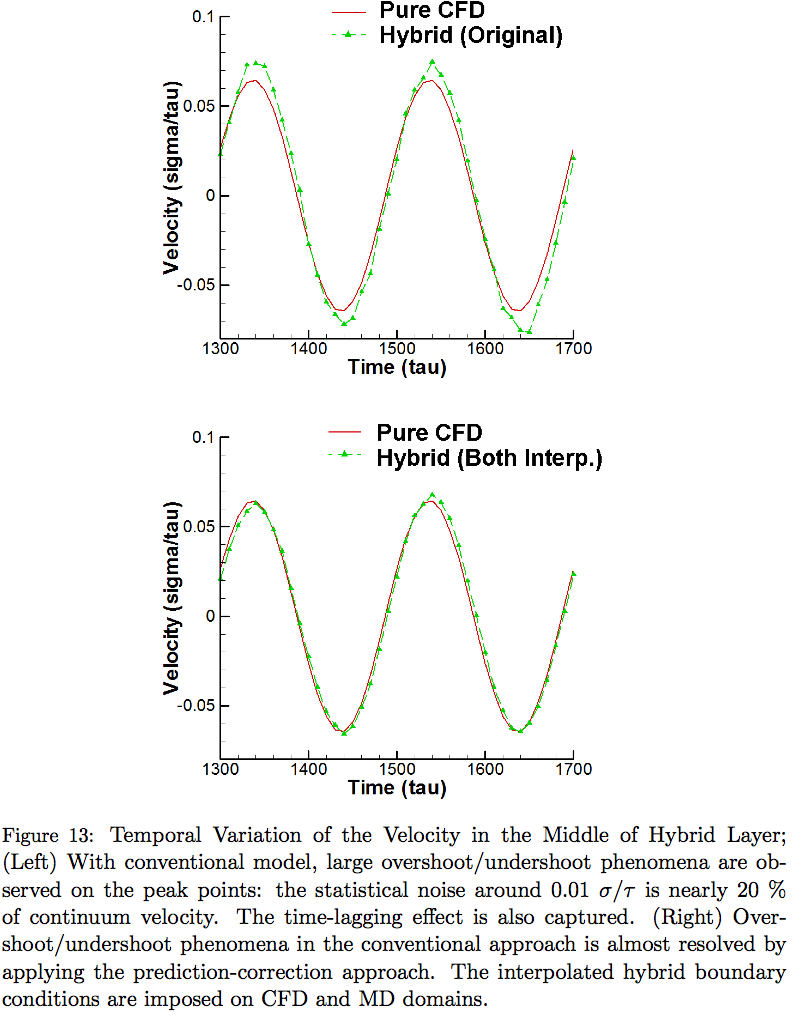


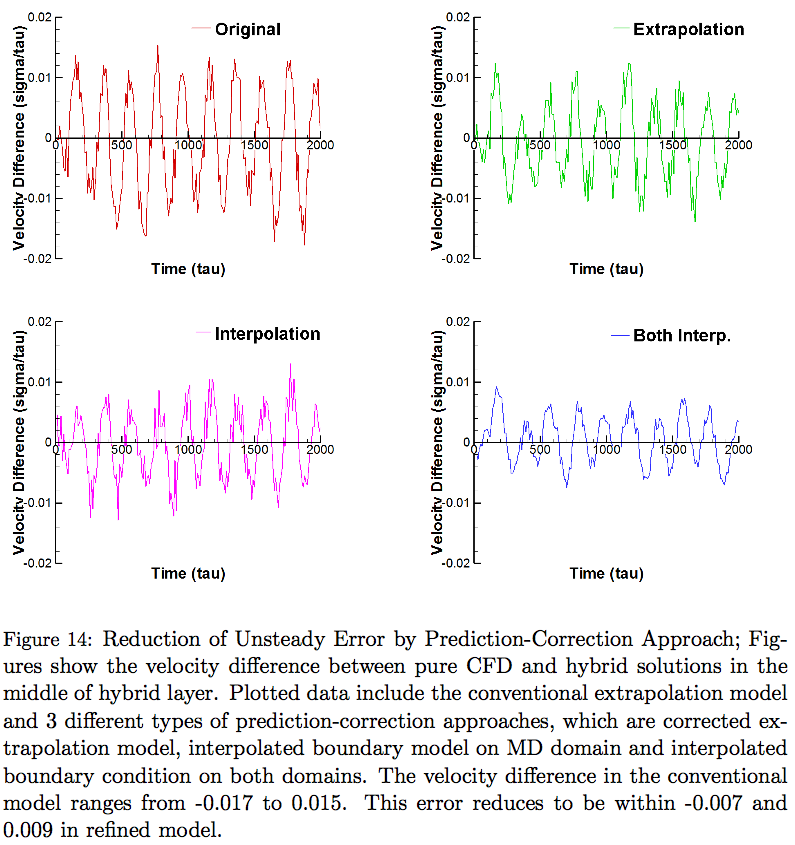
Comparing temporal evolutions of the velocity field by different coupling schemes in Fig. 13, the clear difference is the resolution of the overshoot/undershoot phenomena by the prediction-correction approach. Sufficient numbers of independent experiments are sampled to reduce the statistical error and prediction time scale of 2.5 sampling intervals is chosen on prediction-correction approach to impose the interpolated boundary conditions on both domains. On conventional model, the maximal error is seen at peak points. This is a natural characteristics of the linear extrapolation that it fails to predict the correct values around the strong variation. This inaccuracy is resolved by using a prediction-correction approach and applying interpolated boundary condition from predicted flow properties. However, another numerical inaccuracy of time-lagging pattern in conventional model is also seen in the prediction-correction model. This necessitates the implementation of higher-order interpolation schemes.



Plots in Fig. 14 quantifies the scale of inaccuracy according to the hybrid boundary condition imposition. In the conventional model, the velocity difference ranges from -0.017 to 0.015 σ/τ. The magnitude of this error reduces as we apply the prediction-correction approach and increase the prediction time scale from 0.5 (corrected extrapolation) to 1.5 (MD interpolation) and 2.5 (both interpolation) sampling intervals. The clear difference between the conventional extrapolation model and corrected extrapolation in prediction-correction approach demonstrates that the extrapolation from two ''previous'' properties contains a lot more error compared to extrapolating from the current value. The similar variation between extrapolation and MD interpolation in prediction-correction approach emphasizes that the accuracy of boundary condition in CFD domain is as important as the boundary condition in MD domain. Lastly, the solution error form the conventional model is reduced by half with the imposition of interpolated boundary conditions on both domains.







1. *Performance Analysis of a Multi-physics Simulation Framework*

The BigJob framework is expected to save total runtime of a coupled multi-task simulation than conventional (and direct) job submissions. Logical bases are as follows:

-1 Many supercomputing centers adopt the queueing policy of assigning higher priority on bigger jobs. This increases the probability that a bigger-sized task gets allocated faster than the smaller one(s).

-2 Independently submitted multiple tasks are usually allocated at different physical time. So, requested wall-time limit should be sufficiently large to cover both maximal waiting time and computation time in case of coupled simulations. On the other hand, co-scheduling of coupled tasks is inherently guaranteed in a packaged job. So wall-time limit can be determined according to the actual computation time.

-3 Load-balancing among multiple tasks can be dynamically achieved by changing the resource allocation to individual task during the simulation. It can be possible only when coupled yet distributed tasks are scheduled in the umbrella of a packaged job.

In this section, we present our numerical experiments to verify the above logics.

*5.1. Waiting Time according to the Size and Wall-time Limit of an Individual Job*

Many factors effect the waiting times, arguably the most important of them are the existing job requests on the resource at the instant of submission, requested wall-time limit and also the number of processors requested. Two other factors that affect this are the backfilling capability (which allows the running of a small job in between higher priority jobs with larger and longer resource requests) and the changes in the priority of the test job when a particular higher priority job joins the queue. Of these factors, number of requested resources and wall-time limits are what can be handled by users and can be systematically accounted for.

We designed experiments to determine if running larger and/or longer simulations effects the actual waiting time on the queue. We submitted jobs of different sizes and different wall-time limits at the same time. Each time we submitted a job, we gathered the actual waiting time on the queue. We performed our experiments on a sufficiently large and crowded system -- around 65000 px. We argue that the use of this large and crowded system increases the credibility of our experiments: it reduces the possibility of self-interference between our job submissions (i.e., our requests are allocated in a succeeding pattern). Considering the internal queueing policy of supercomputing centers, such as credential, fairshare, resource and service priority, each user account has the restricted number of concurrent job submission and the allocation history affects the priority of next submitted job. So, we submitted different wall-time limit jobs from different user accounts in the same group allocation, and we put enough idling time between individual experiments to recover the priority of each account to the initial level. Results from 10 independent experiments are averaged for measurement.

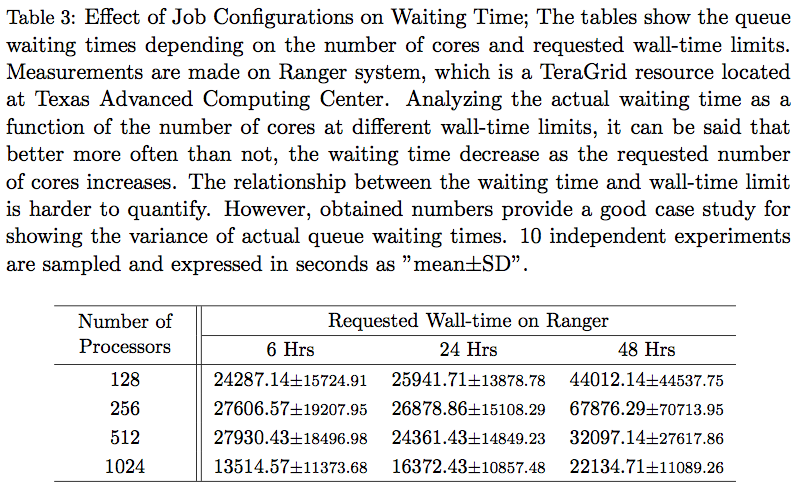
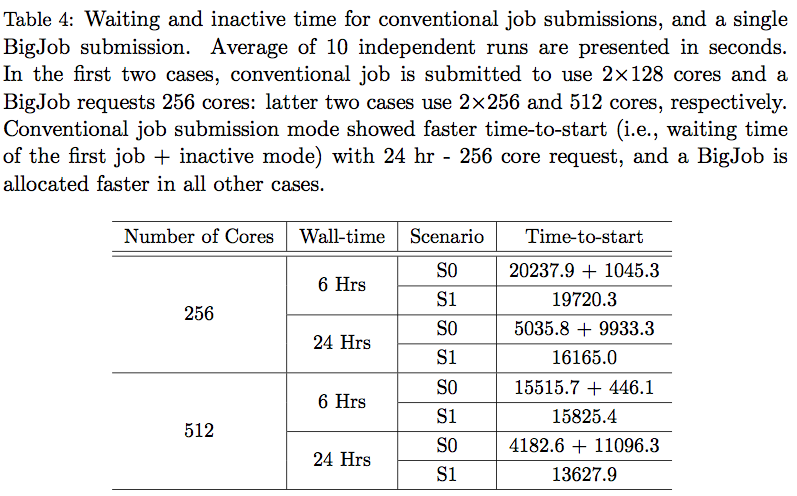


Table 3 demonstrates the queue waiting time according to the size and wall-time limit of the job. Regarding the influence of job sizes on waiting time, jobs with larger processor counts have typically lower wait times, with the exception at 128 cores which is presumably affected by the backfilling capability. Waiting time with 1000+ cores shows the stiff decrease particularly because it is granted with the higher priority in this queueing policy. The same applies to the increase of waiting time with the wall-time limit of 48, which is administrated by the different queue. The effect of wall-time limit on waiting time is not clear in our experiment, in case they are on the same queue (6- and 24-hour jobs). Collectively, submitting a BigJob for the coupled simulation instead of individually submitting multiple smaller tasks at least provides the comparable waiting time on the queue even when small jobs are "ideally" allocated (i.e., all jobs are allocated at the same time).

*5.2. Waiting Time for a Coupled Simulation*

Looking back on the scenario maps in Fig. 5. the result at Sec. 5.1 provides the waiting time between a BigJob and a first-allocated conventional task. We expect more performance gain by eliminating the inactive idling time (i.e., the difference between the waiting time of the two jobs) through the use of a BigJob.

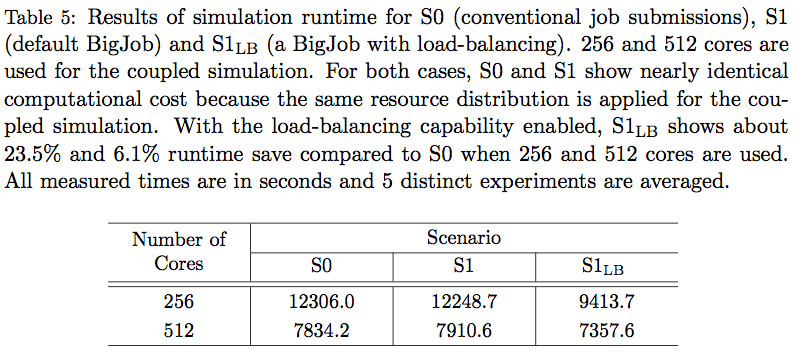
Table 4 presents the waiting time of a BigJob with size 2X and two conventional job submissions with size X each. We experimented on two different wall-time limits (6 and 24 hours) and two different processor requests (256 and 512 cores in total). A BigJob submission shows faster allocation except the small number of processors are requested for a long time, although the waiting time for the first-to-start job was smaller in the conventional job submission mode (S0) than the BigJob (S1). Interestingly, the inactive mode in conventional job submissionis observed to become quite larger as the wall-time limit increases: users get to waste more allocation time as they increase wall-time limit. From the result, bigger and longer jobs tend to start faster (comparing individual experiment), but it is hard to affirm this tendency as the general phenomenon because individual experiment has been performed at different time.



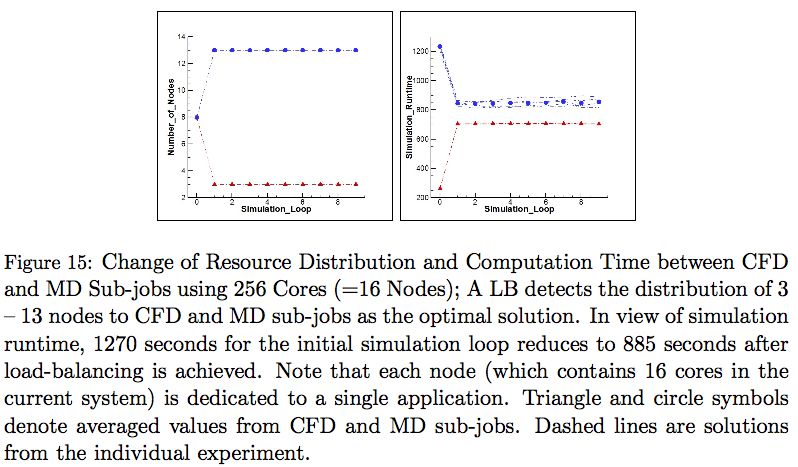
*5.3. Performance Gain through Load Balancing*

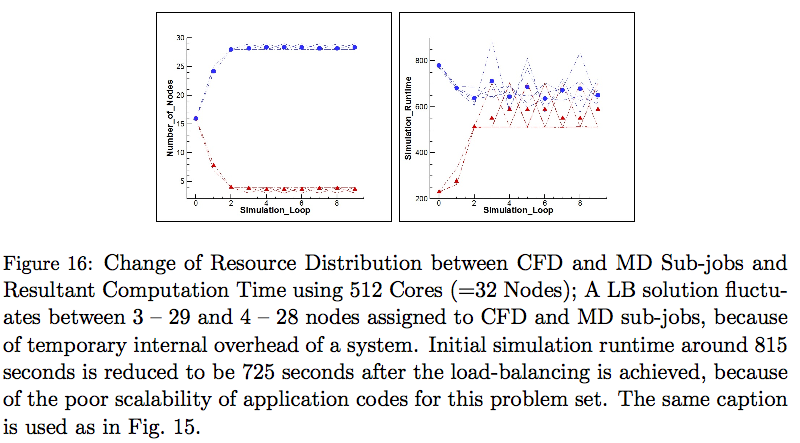
A BigJob framework in itself does not provide any performance gain during the simulation: a load balancer (LB) in a BigJob framework provides the better parallel performance by redistributing processors to individual task within a context of packaged job. LB measures the performance of individual task during the temporary stop and evolves application codes with changed number of processors. So, each codeis scheduled to stop-and-restart several times for the complete simulation.It necessitates the capability of the application-level checkpointing and generic domain partitioning routine in application codes (both of which are incorporated capacities in most legacy codes).

The benefit of a LB in a BigJob framework is that it can be applied to any types of coupled applications. On the other hand, it is not highly required for the current application in two reasons: (1) in a hybrid CFD-MD simulation, MD usually requires far more computing power that most resources are dedicated to molecular dynamic simulation; (2) computational cost for the current application in Sec. 4.1 is so small that it is not necessary to endeavour to apply the load-balancing capability. This leads us to increase the computational cost of each simulation, for the purpose of evaluating the LB in a BigJob framework. In this experiment, CFD domain size is increased by 500 times and MD domain is increased by 20 times. CFD and MD codes are scheduled to have 10 simulation loops: initially start with tne same number of processors and experience nine stop-and-restarts with changed resource allocation according to the result of a LB.



Runtimes of the coupled simulation with a single BigJob is given on Table 5. In all scenarios, the same number of processors are initally assigned to CFD and MD simulations. Processors distribution in S1LB is changing during the simulation according to the result of a LB function. For both simulations, the time difference between S0 and S1 are within 1%. This explains that the possible overhead of a BigJob due to the communication with the advert server (for minitoring the status of subjobs) are negligible. In cases of load-balanced BigJob simulations, there is a significant reduction in the runtime compared to the default BigJob application when 256 processors are used -- 23.1%. For larger problem set, the performance gain through the application of a LB is relatively small (7.0%), whose reason is discussed below.





The validity of a LB can be discussed by the change of processor distribution between sub-jobs throughout the simulation. For the result of a 256 px (=16 nodes) simulation in Fig. 15, both CFD and MD subtasks are assigned with 8 nodes initially. From the next simulation loops, the processor distribution converges to 3 to 13 nodes between CFD and MD respectively and this ratio is maintained throughout the simulation. CFD and MD computation times change from 260 -- 1235 seconds at initial loop to 705 -- 850 seconds after load-balancing. Simulation runtimes are measured to be about 35 seconds longer than the slower simulation (MD simulation) per each simulation loop, which are the overhead by the initialization, I/O, and communication between coupled applications.

The result of computation time evolution in 512 px (=32 nodes) case is presented in Fig. 16. Compared to the above experiment when smaller px are used, the load-balancing solution shows noisier pattern. In detail, a LB fails to find the optimal solution at the first simulation loop, and the node allocation after load-balancing fluctuates between 3 -- 29 and 4 -- 28 nodes to each application. This is caused by two reasons. First, individual code has poor scalability. Initial simulation time for CFD and MD applications are measured around 230 and 780 seconds and the LB proposes the node distribution from 16 -- 16to 8 -- 24. However, the computation time at the next simulation loop are measured to be 275 and 680 seconds, respectively. Therefore, the LB has to search for the optimal solution one more time. This is the limit of load-balancing functions which are incorporated in schedulers, since they have to manage black-box applications with unknown problem size based on restricted information. Another reason for the fluctuation is the momentary overhead in computing system. In our experiment, MD simulation times when 28 nodes are used vary from 610 seconds to 880 seconds, depending on the magnitude of internal overhead. This evaluates the necessity of a LB as a self-correcting tool in response to the instability of a system.

Two things remained for open discussions are how to effectively measure the actual simulation time of individual task and how to determine the number of simulation loops. We put time checking routines in between inter-domain communication routine and accumulate the time for running iteration loop of each code. Based on our knowledge, there is no way of systemically gathering the individual simulation time except this manual deployment. Regarding the number of simulation loops, excessive number of loops increase the I/O-related overhead (storing checkpointing solution and restarting from that); insufficient number of loops increases the simulation time at initial imbalanced configuration and degrades the capability of a LB for adapting to the unpredicted internal overhead.

1. *Next Step: Further Refinement*

The empirical mathematical equation on sampling noise has been presented in Sec. 4.1. We argue that this is a refined model compared to previous mathematical expressions. However, the formulation has not been verified by various systems and conditions: coefficients will be changing according to the distance from the wall, characteristics of fluid and solid elements, etc. More rigorous research is required to address a globally acceptable equation of sampling noise.

Unsteady flow simulation in Sec. 4.3 verifies that the prediction-correction approach provides more accurate solution than conventional temporal coupling scheme. The new approach is especially powerful in resolving the unfavorable overshoot/undershoot phenomena. On the other hand, the slight time-lagging effect in the conventional model has not been improved by the prediction-correction approach. We expect that this phenomenon can be resolved by applying higher-order extrapolation/interpolation on hybrid boundary regions.

1. *Conclusions*

Accurate and efficient multi-scale flow simulations by a hybrid CFD-MD simulation framework have been presented in this paper. Constrained Lagrangian dynamics equations of motion and file-based hybrid interfaces are implemented on a highly-reliable LAMMPS molecular dynamics package and a verified in-house incompressible CFD code. They are virtually integrated as a single BigJob framework.

A number of numerical issues which harm the accuracy of a hybrid solution have been explored. First, quantifying the sampling noise from a stationary flow has been proposed as a way of determining coupling parameters. We argue that our simple and intuitive idea unveils the influence of individual coupling parameter on the magnitude of statistical error and is very cost-effective in contrast to traditional trial-and-error approach. Moreover, the empirical equation derived from the stationary flow simulation describes that well-know mathematical expressions on statistical error are not accurate on nano-scale wall-bounded systems. Second, sampling multiple independent replicas has been introduced to refine the sampling noise of an individual solution and to explore to the low-speed flow regimes. This approach is superior to simulating a single large-scale problem set which is technically bound by computing capacity. The application to a Couette flow simulation in O(10) m/s velocity field is the first successful report of a moderate-speed flow simulation using a hybrid CFD-MD approach. Last, a prediction-correction approach has been designed for the accurate unsteady simulation. This approach acquires better solution by enabling the imposition of interpolated hybrid boundary conditions. The application to the oscillating boundary problem expresses that the current approach diminishes the overshoot/undershoot phenomena in the conventional methods.

Along with numerical issues, computational issues for the efficient coupled simulations have been also discussed. We introduced a BigJob framework and this directly solves the co-scheduling problem among logically separated sub-tasks. A simple load-balancing function is also implemented on a BigJob framework, to achieve the load-balancing among those separated-yet-coupled codes. From numerical experiments, we evaluate that a BigJob is very powerful in reducing the waiting time of the coupled simulation. Also, a simple load-balancing function employed in a BigJob is effective in reducing the simulation runtime.

We emphasize that above numerical investigations ease the challenge to the hybrid simulation and broaden the application area. Also, our computational experiments contribute on how to efficiently conduct coupled simulations.

1. Sampling interval is designed less than *1/100th* of hydrodynamic characteristic time; The location of sampling layer is placed *O(10)* nanometers above the solid obstacle. [↑](#footnote-ref-1)
2. The height of the domain can be reduced by placing a specular wall on the top, in case the system is sufficiently large; The length of the domain along the periodic direction can be arbitrarily chosen. The optimal length is further determined by the relation between the strength of noise and number of particles, i.e., *Vnoise∝ 1/√N*. [↑](#footnote-ref-2)
3. Data collection starts as soon as the relaxation process is finished. Temporal history of averaged velocity from the smallest layer size with shortest sampling interval is stored. [↑](#footnote-ref-3)
4. Produced dataset around the location of sampling layer is spatially and temporally averaged to produce the spatial and temporal variation of the sampled velocity. [↑](#footnote-ref-4)
5. The noise is compared with the expected macroscopic velocity at that position, considering the linear velocity gradient from the wall to the far field. A paired condition which produces sufficiently small portion of noise and whose temporal duration is less than designed sampling interval is chosen to be the layer size and sampling duration of this hybrid simulation.

   [↑](#footnote-ref-5)
6. If no condition is satisfactory, the acceptable condition can be obtained by either increasing the length of computational domain based on *Vnoise∝ 1/√N* or changing the position of sampling layer and repeating data processing. The one condition which generates the smallest MD domain in the hybrid simulation is chosen. [↑](#footnote-ref-6)