End-to-End Differentiable Fluid-Particle Simulations

Reiji Akashi[1], Fujita Daiki[1], Sam Turley[2], Mark P. Lynch[2], Mayank Dixit[1], Simon K. Schnyder[3], Takeshi Sato[4], Souta Miyamoto[1], Ryoichi Yamamoto[1], Matthew Turner[5], Takashi Taniguchi[1], John J. Molina[1*]

[1] Department of Chemical Engineering, Kyoto University, Kyoto, 615-8510, Japan
[2] Mathematics of Systems CDT, University of Warwick, Coventry, CV4 &AL, U.K.
[3] Institute of Industrial Science, The University of Tokyo, 4-6-1 Komaba, Meguro-ku, Tokyo, 153-8505, Japan
[4] Advanced Manufacturing Technology Institute, Kanazawa University, Kanazawa, 920-1192, Japan
[5] Department of Physics, University of Warwick, Coventry, CV4 &AL, U.K.
john@cheme.kyoto-u.ac.jp

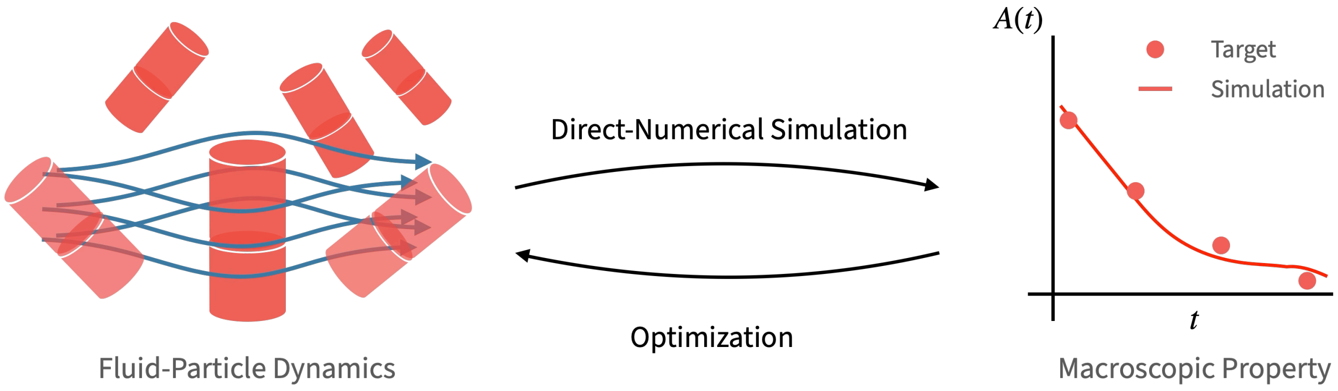

Fig. 1 Schematic representation of an end-to-end differentiable fluid-particle simulation setup. Direct numerical simulations for a complex particle dispersion are performed to measure system property $A(t)$. To obtain the desired/target results we seek to minimize a loss function of the form $L(\theta) = \sum (A^{\text{sim}}(\theta) - A^{\text{target}})^2$, with θ the system parameters. This optimization can be efficiently performed using a gradient-based method, with the gradients computed through the simulation via Automatic Differentiation.

Abstract

Particle dispersions (i.e., particles dispersed in a host fluid) are ubiquitous in the physical and biological sciences, as well as for engineering applications. These systems are characterized by **long-range many-body hydrodynamic interactions**, which are crucial to understand their macroscopic properties. Unfortunately, theoretical descriptions are limited to idealized systems (e.g., one or two particles), leaving computer simulations as the method of choice. To date, many computational methods have been developed, including Stokesian Dynamics, Lattice Boltzmann, Multi-Particle Collision Dynamics, Fluid Particle Dynamics, and the Smooth Profile Method (SPM)[1]. The **SPM** is able to directly solve the coupled fluid-particle dynamical equations; it can handle **multi-component non-Newtonian host fluids**, arbitrarily shaped particles, and it has no restriction on the Reynolds numbers. However, the standard formulation and implementation makes it unsuitable for solving inverse problems, e.g., to optimize flow conditions, fluid-particle affinity, or particle-particle interactions. To overcome this limitation, we aim to build a **composable and end-to-end differentiable fluid-particle simulator** that can be used to efficiently solve complex **fluid/particle optimization problems**.

Model & Methods

Smooth Profile Method

We use the Smoothed Profile Method (SPM) to model the fluid-particle interaction [1]. The SPM replaces the sharp particle interface with a diffuse one, allowing particle quantities to be defined as fields. The coupled equations of motion are

Fluid

$$\dot{\boldsymbol{u}} = (\boldsymbol{u}\cdot\nabla)\boldsymbol{u} + \rho^{-1}\nabla\cdot\boldsymbol{\sigma} + \phi\boldsymbol{f}_p$$

$$\nabla\cdot\boldsymbol{u} = 0$$

Particles

$$\dot{\boldsymbol{R}}_i = \boldsymbol{V}_i \qquad M_i\dot{\boldsymbol{V}}_i = \boldsymbol{F}_i$$

$$\dot{\boldsymbol{Q}}_i = \text{skew}(\boldsymbol{\Omega}_i)\cdot\boldsymbol{Q}_i \qquad \dot{\boldsymbol{J}}_i = \boldsymbol{N}_i$$

ρ: density
$\boldsymbol{u}$: velocity
$\boldsymbol{\sigma}$: stress-tensor
$\phi\boldsymbol{f}_p$: constraint force
$\boldsymbol{R}$: position
$\boldsymbol{Q}$: orientation
$\boldsymbol{V}$: velocity
$\boldsymbol{\Omega}$: ang. velocity
$\boldsymbol{F}$: force
$\boldsymbol{N}$: torque
$\boldsymbol{J}$: ang. momentum
M: mass

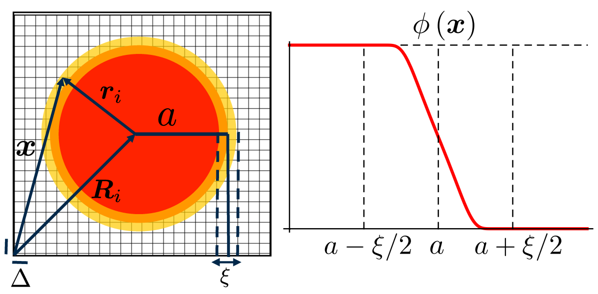

Fig. 2 Schematic representation for an SPM particle with a diffuse interface of thickness ξ.

where $\boldsymbol{u} = (1-\phi)\boldsymbol{u}_f + \phi\boldsymbol{u}_p$ is the *total* velocity field, which includes both the fluid ($\boldsymbol{u}_f$) and particle velocity fields ($\boldsymbol{u}_p$), with ϕ the phase field function that defines the particle domain (see Fig.2). The particle velocity field is defined as

$$\phi\boldsymbol{u}_p(\boldsymbol{x}) = \sum_i \phi_i(\boldsymbol{x})\left[\boldsymbol{V}_i + \boldsymbol{\Omega}_i\times\boldsymbol{r}_i\right]$$

The hydrodynamic forces/torques are computed by assuming momentum conservation, with the constraint force term $\phi\boldsymbol{f}_p$ introduced to maintain particle rigidity.

Automatic Differentiation

Automatic Differentiation (AD) is a form of program transformation that allows one to compute the derivative of an arbitrary function $f(x)$, as given by a program [2]. It forms the backbone of modern Machine-Learning (ML). Assuming that the function/program under consideration can be expressed as a composition of mappings μ_i between smooth manifolds

$$f = \mu_N \circ \mu_{N-1} \circ \cdots \circ \mu_1$$

the Jacobian of the f transformation is given by the (matrix) multiplication of the individual Jacobians

$$J = J_N \cdot J_{N-1} \cdot \cdots \cdot J_1$$

The JAX [3] library provides a framework for building Python/NumPy functions that can be arbitrarily composed and transformed, allowing us to automatically compute their derivatives.

Research Plan

Our research is divided into three main themes / components:

(A) **SPM implementation** : Implement the SPM in JAX with jit and grad support. Incrementally add support for ellipsoidal particles, arbitrarily shaped particles, complex fluids, etc.

(B) **Optimization** : Parallelize the JAX code to target multi-node / multi-GPU systems.

(C) **Flow optimization** : Use the new simulator to solve inverse flow design problems, e.g., optimizing processing flows and inferring particle-particle interaction potentials.

Preliminary Results

Flow Optimization / Inference

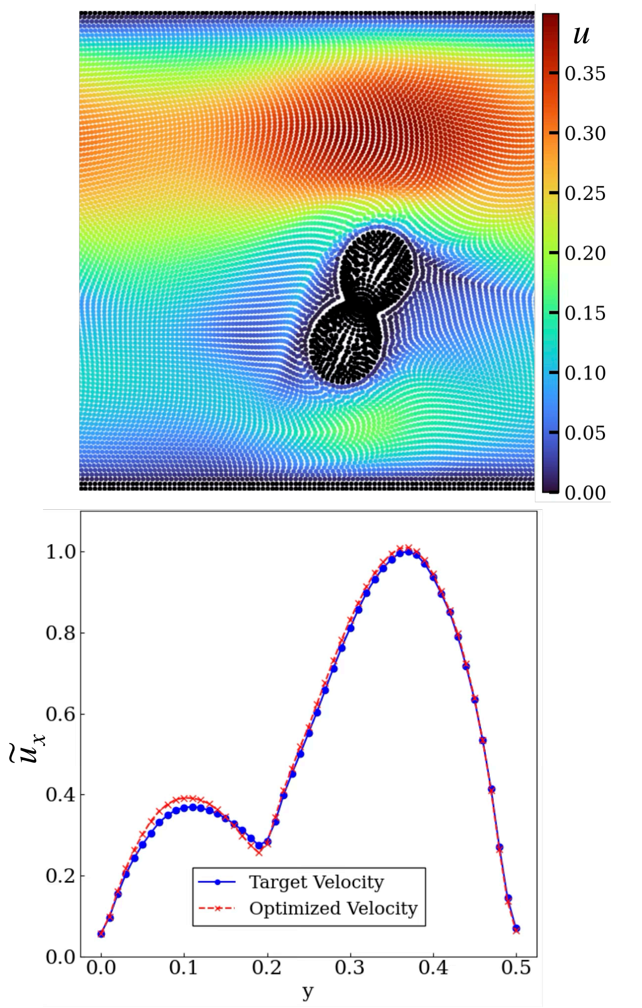

Fig. 3 Sample optimization problem for 2D pressure-driven flow in a channel with a particle inclusion. The particle position and shape are optimized to produce a target flow velocity at the outlet. Figure adapted from Ref. [4].

Acknowledgements

Support by the Japan Society for the Promotion of Science (JSPS) KAKENHI Grant No. 25H01536, 25H01476, and 25K03185.

References

[1] R. Yamamoto, J. J. Molina, Y. Nakayama, *Smoothed profile method for direct numerical simulations of hydrodynamically interacting particles*, Soft Matter **17**, 4226-4253 (2021).

[2] A. G. Baydin, B. Pearlmutter, A. A. Radul, J. Siskind, *Automatic differentiation in machine learning: a survey*. Journal of Machine Learning Research **18**, 1-43 (2018).

[3] J. Bradbury, R. Frostig, Peter Hawking et al., *JAX: composable transformations of Python+NumPy programs*, http://github.com/jax-ml/jax (2018).

[4] Akashi Reiji, *Undergraduate Thesis*, Kyoto University (2025).