

# Complex fluids in the GPU era

## Algorithms and simulations

Raúl P. Peláez

Universidad Autónoma de Madrid

May 9, 2022

Supervisor: Rafael Delgado-Buscalioni

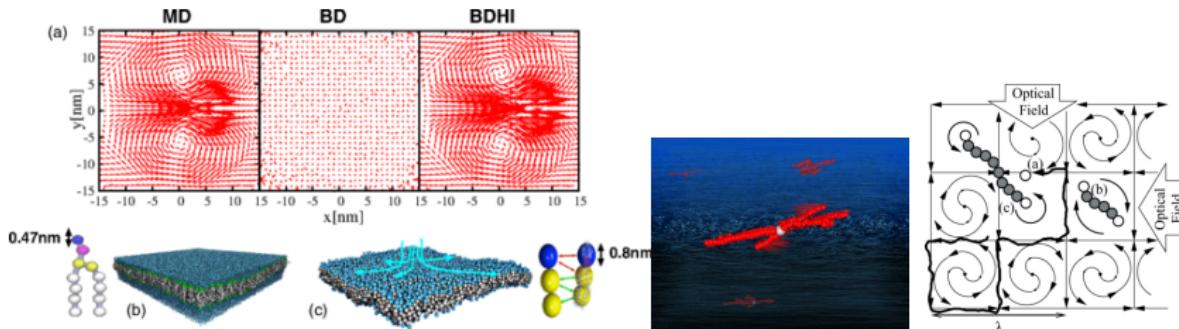


# Talk outline

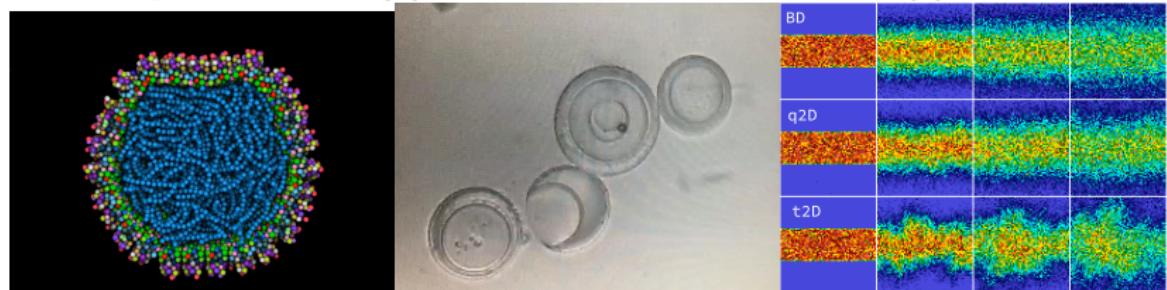
- 1 Introduction
- 2 Computational challenges
- 3 The Force Coupling Method
- 4 UAMMD
- 5 Examples

# Complex fluids

## The coexistence between a liquid and solid phase



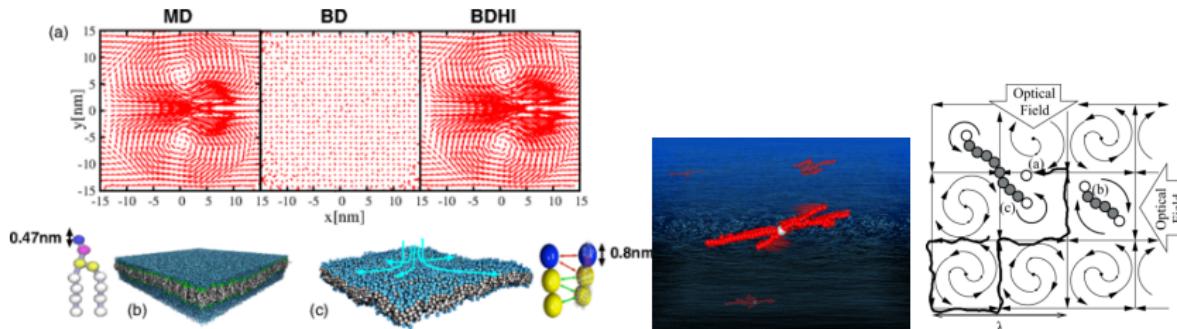
[Panzuela, S. and Delgado-Buscalioni, R. PRL 2018.] - [Raul P. Pelaez and Delgado-Buscalioni, R. Macromolecules 2020] - [Meléndez, M. et. al. PRE 2019.]



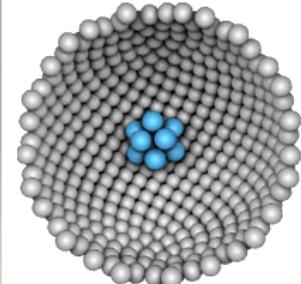
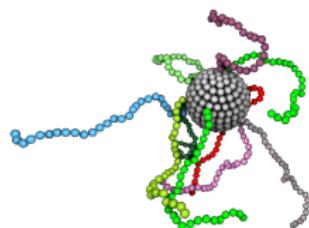
[Courtesy of Pablo Ibañez] ————— [Courtesy of Berta Tinao] ————— [Raul P. Pelaez et. al. JSTAT 2018.]

# Complex fluids

## The coexistence between a liquid and solid phase



[Panzuela, S. and Delgado-Buscalioni, R. PRL 2018.] - [Raul P. Pelaez and Delgado-Buscalioni, R. Macromolecules 2020] - [Meléndez, M. et. al. PRE 2019.]



[Courtesy of Pablo Palacios]

[Courtesy of Berta Tinao]

[Courtesy of Pablo Palacios]

# Complex fluids

The spatio-temporal landscape of numerical techniques.

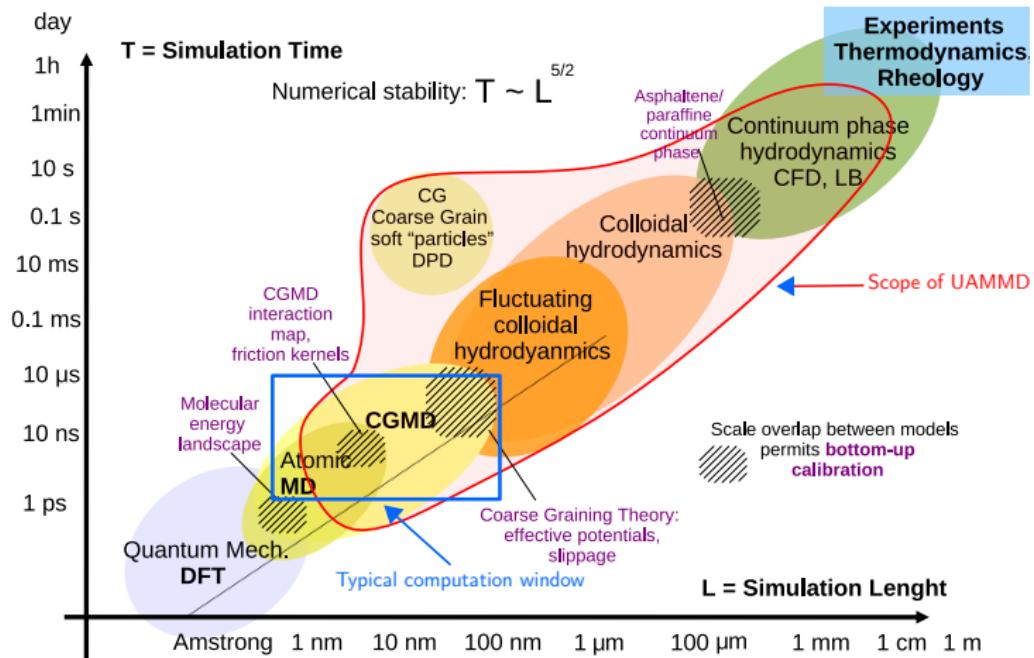


Figure: Figure courtesy of Rafael Delgado-Buscalioni.

# Complex fluids

The spatio-temporal landscape of numerical techniques.

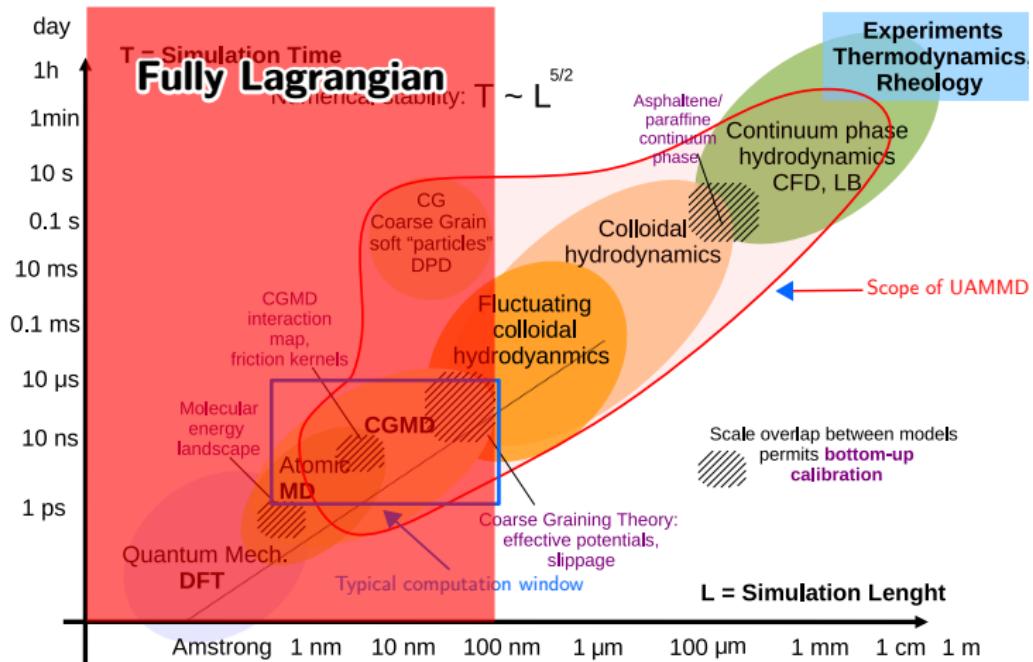
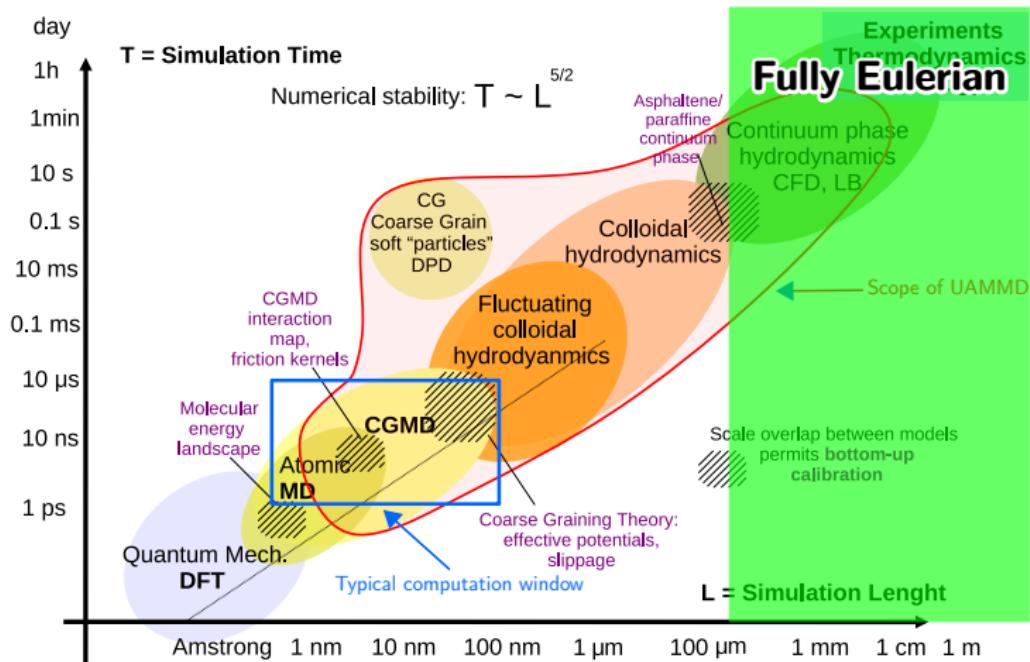


Figure: Figure courtesy of Rafael Delgado-Buscalioni.

# Complex fluids

The spatio-temporal landscape of numerical techniques.



**Figure:** Figure courtesy of Rafael Delgado-Buscalioni.

# Complex fluids

The spatio-temporal landscape of numerical techniques.

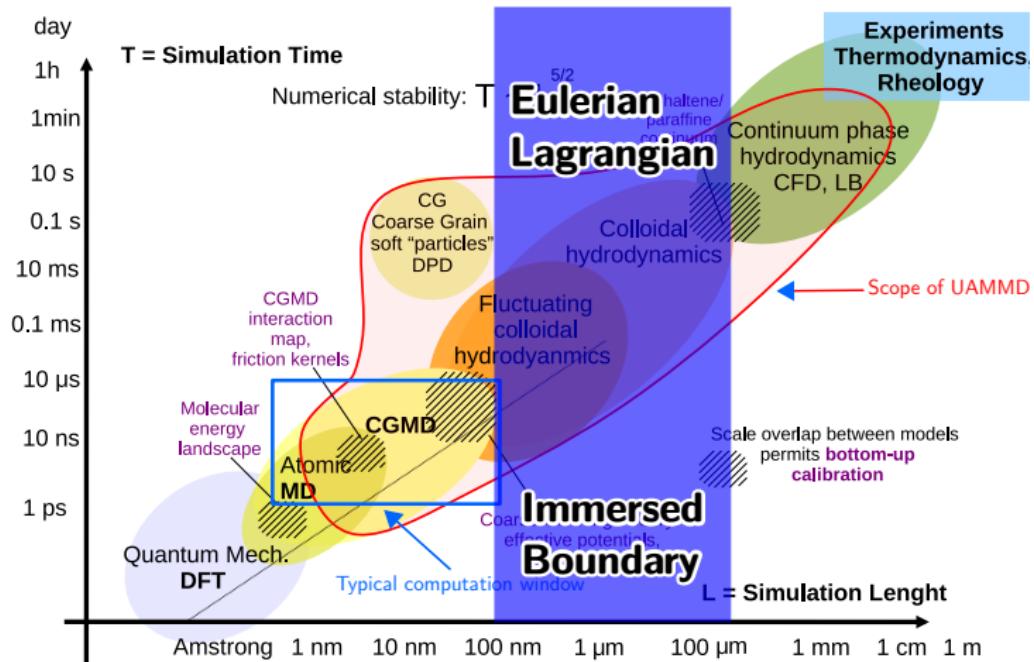


Figure: Figure courtesy of Rafael Delgado-Buscalioni.

# Complex fluids

The spatio-temporal landscape of numerical techniques.

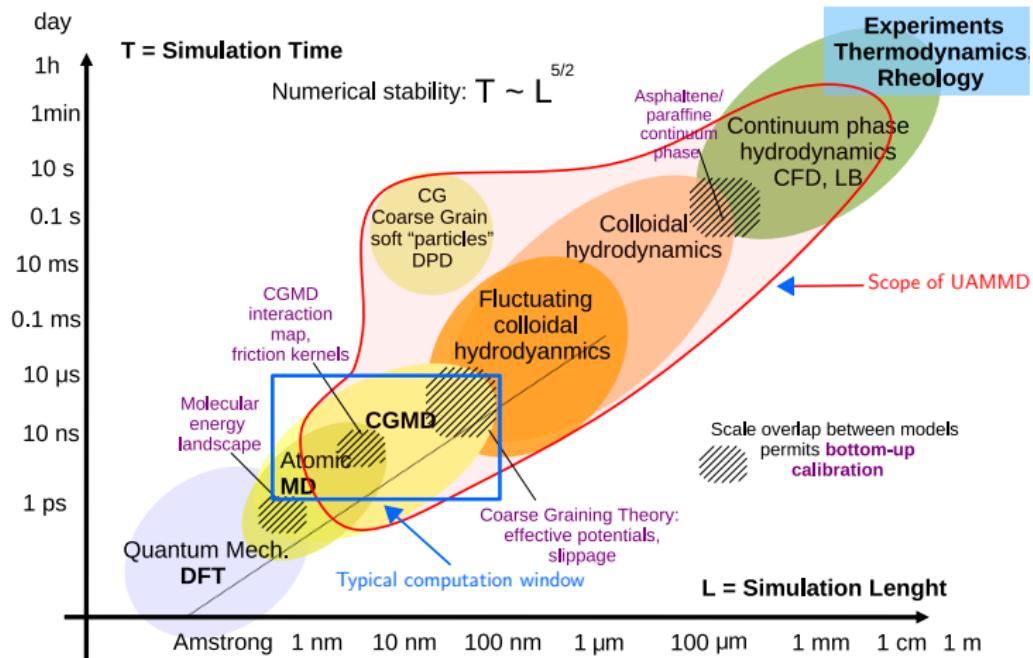
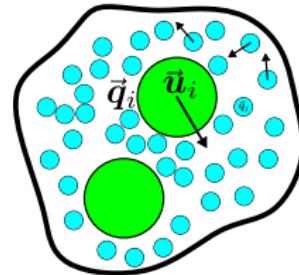


Figure: Figure courtesy of Rafael Delgado-Buscalioni.

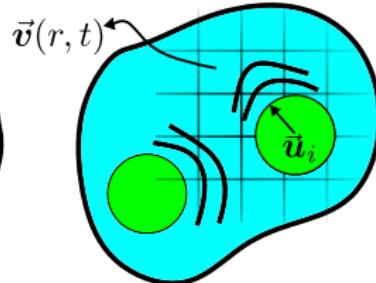
# Complex fluids

Usual levels of coarse-grained description

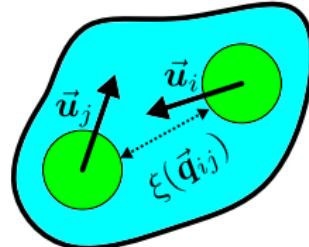
Microscopic



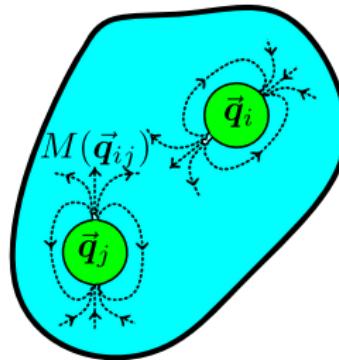
Hydrodynamic



Langevin



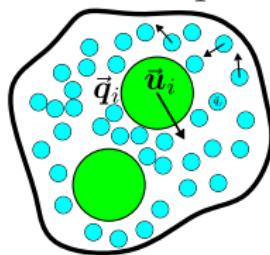
Smoluchowski



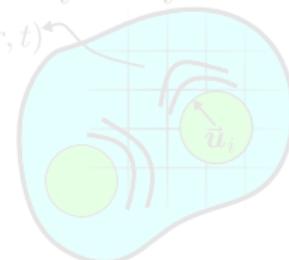
# Complex fluids

Usual levels of coarse-grained description

Microscopic



Hydrodynamic



Langevin

Smoluchowski

Molecular Dynamics (MD)

$$m\ddot{\vec{q}} = \vec{F}$$
$$\vec{u} = \dot{\vec{q}}$$

**Relevant variables:**

- $\vec{q}_i$ : Position of particle  $i$ .
- $\vec{u}_i$ : Velocity of particle  $i$ .

**Typical timescale:**

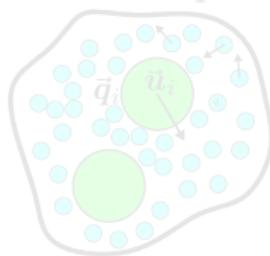
$$\tau \sim 10^{-12} s$$

Supervector notation  $\vec{q} := \{\vec{q}_1, \dots, \vec{q}_N\}$

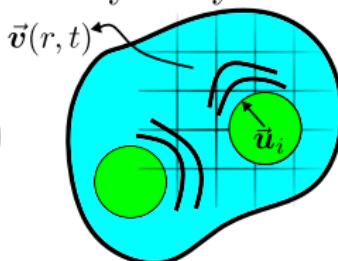
# Complex fluids

Usual levels of coarse-grained description

Microscopic



Hydrodynamic



**Relevant variables:**

- $\bar{q}_i$ : Position of particle  $i$ .
- $\bar{u}_i$ : Velocity of particle  $i$ .
- $\vec{v}(r, t)$ : Fluid velocity field.

## Fluctuating Hydrodynamics

$$\rho \partial_t \vec{v} = -\partial_r \cdot \boldsymbol{\sigma} + \mathbf{f} + \text{fluct}$$

$$\int_{V_p} \mathbf{f} dr = \mathbf{F}_i \quad \text{and} \quad \mathbf{u}_i = \int_{V_p} \mathbf{v} dr$$

**Typical timescale:**

$$\tau \sim 10^{-\{9-6\}} s$$

$$\partial_t := \frac{\partial}{\partial t} \rightarrow \partial_r := \nabla := (\partial_x, \partial_y, \partial_z)$$

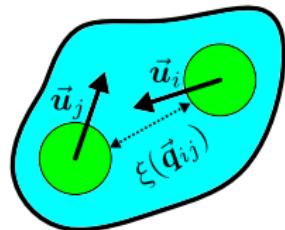
# Complex fluids

Usual levels of coarse-grained description

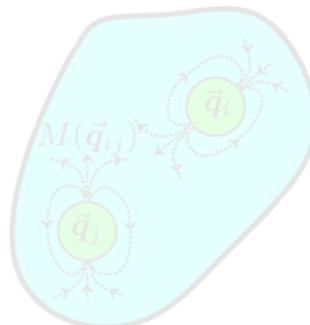
## Langevin Dynamics (LD)

$$mdu = \mathbf{F}dt - \xi u dt + \sqrt{2\xi k_B T} \tilde{\mathbf{W}}$$

Langevin



Smoluchowski



$\tilde{\mathbf{W}}$ : Wiener increments,  $\mathcal{N}(0, dt)$

**Relevant variables:**

- $\vec{q}_i$ : Position of particle  $i$ .
- $\vec{u}_i$ : Velocity of particle  $i$ .
- $\xi(\vec{q}_{ij})$ : Friction kernel.

**Typical timescale:**

$$\tau \sim 10^{-5} s$$

# Complex fluids

Usual levels of coarse-grained description

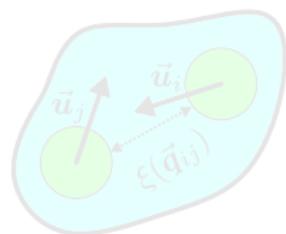
## Brownian Dynamics (BD)

$$dq = \mathcal{M}F dt + \sqrt{2k_B T \mathcal{M}} d\tilde{W} + k_B T \partial_q \cdot \mathcal{M} dt.$$

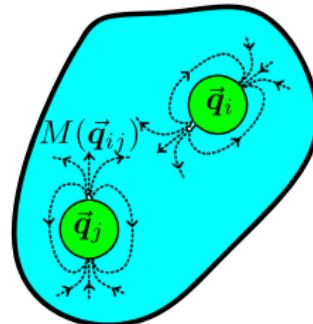
### Relevant variables:

- $q_i$ : Position of particle  $i$ .

Langevin



Smoluchowski



- $\mathcal{M}(q_{ij})$ : Mobility tensor.

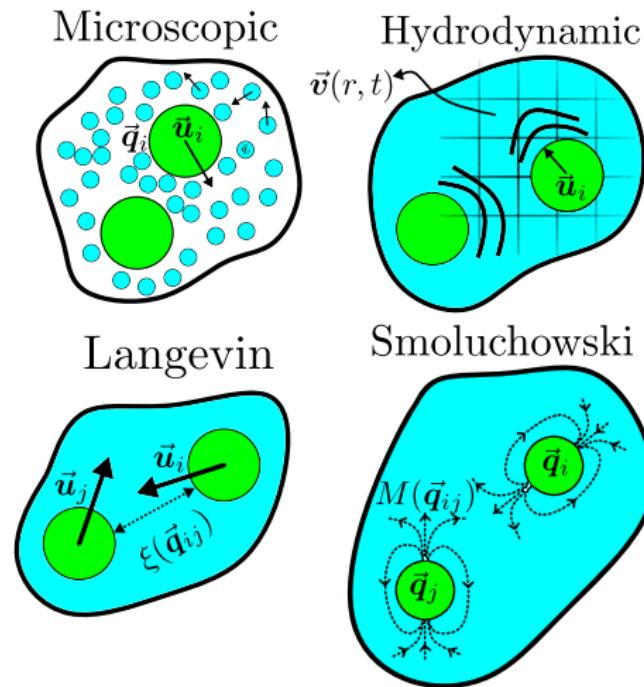
### Typical timescale:

$$\tau \sim 10^{-3} s$$

$d\tilde{W}$ : Wiener increments,  $\mathcal{N}(0, dt)$

# Complex fluids

Usual levels of coarse-grained description



## Relevant variables:

- $\vec{q}_i$ : Position of particle  $i$ .
- $\vec{u}_i$ : Velocity of particle  $i$ .
- $\xi(\vec{q}_{ij})$ : Friction kernel.
- $\vec{v}(\vec{r}, t)$ : Fluid velocity field.
- $\mathcal{M}(\vec{q}_{ij})$ : Mobility tensor.

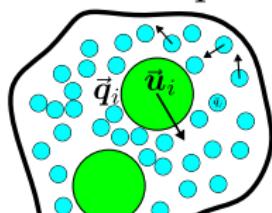
## Timescale range:

$$\tau \sim [10^{-12}, 10^{-3}] s$$

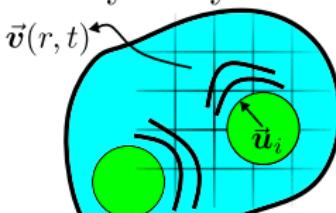
# Complex fluids

Usual levels of coarse-grained description

Microscopic



Hydrodynamic

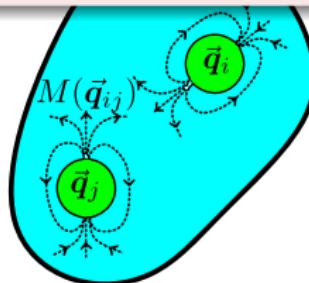
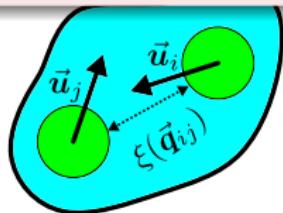


**Relevant variables:**

- $\vec{q}_i$ : Position of particle  $i$ .
- $\vec{u}_i$ : Velocity of particle  $i$ .

We always have

Interacting *particles* with a *state* that evolves.



- $\mathcal{M}(\vec{q}_{ij})$ : Mobility tensor.

**Timescale range:**

$$\tau \sim [10^{-12}, 10^{-3}] s$$

# Computational challenges

- ① Short range interactions
- ② Particle-grid coupling

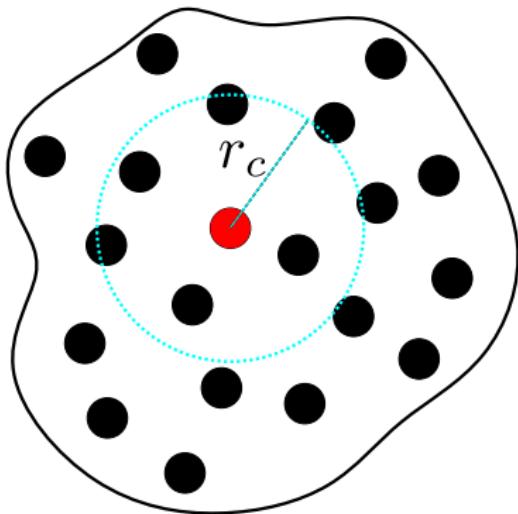
# Computational challenges

## 1 Short range interactions

## 2 Particle-grid coupling

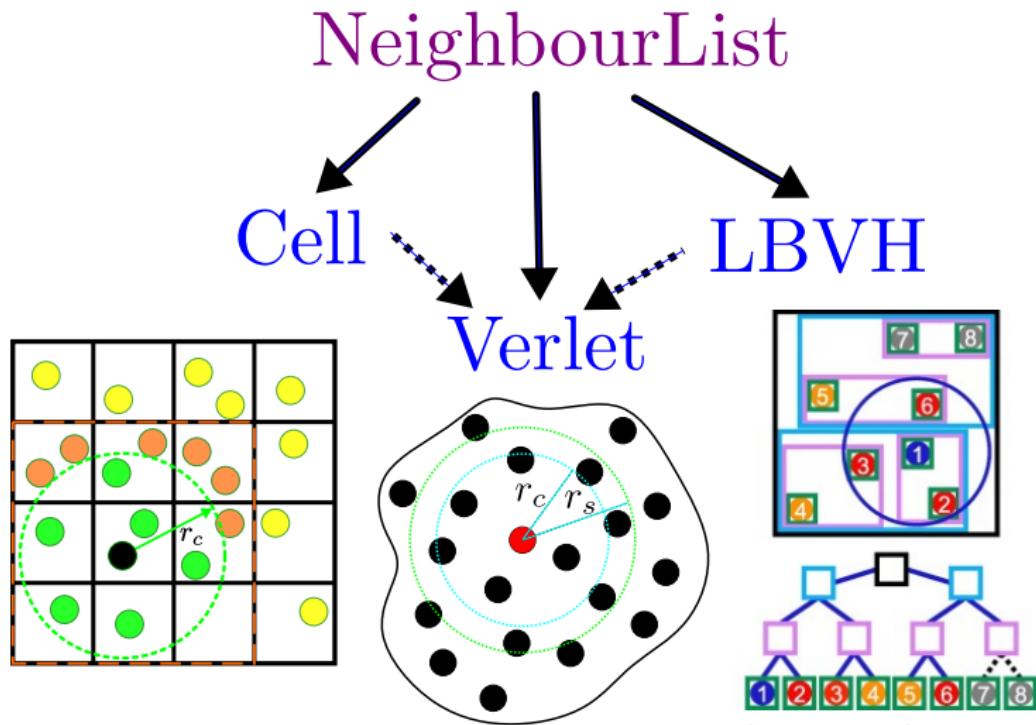
Example: Lennard-Jones potential

$$U_{LJ}(r) = \begin{cases} 4\epsilon \left[ \left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^6 \right] & r < r_c \\ 0 & r \geq r_c \end{cases}$$



# Short range interactions

## Neighbour lists



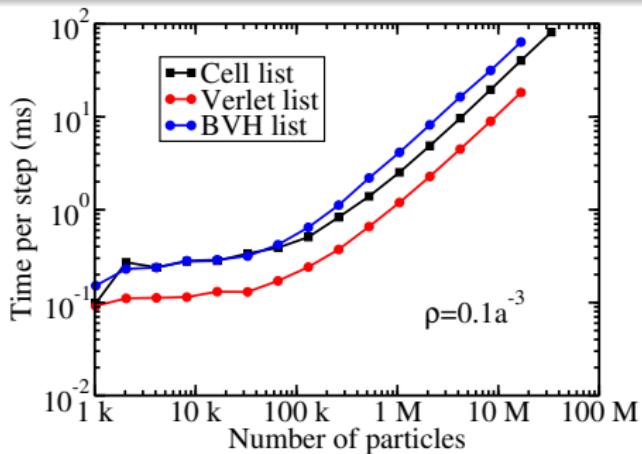
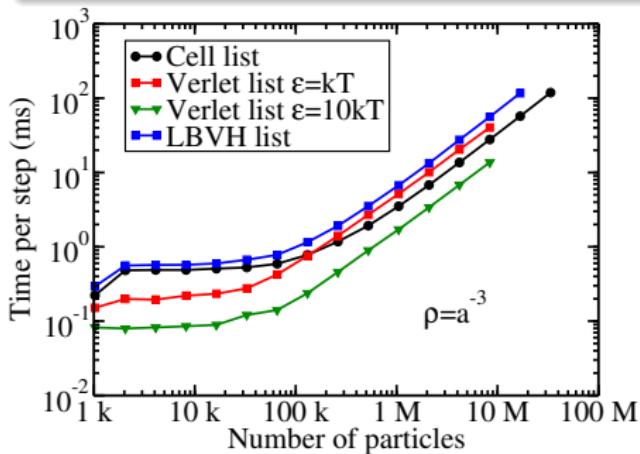
[M.P. Howard et al. 2019]

# Short range interactions

Neighbour lists: Performance

Example: Lennard-Jones potential,  $r_c = 2.5\sigma$  and  $\varepsilon = kT$ .

$$U_{LJ}(r) = \begin{cases} 4\varepsilon \left[ \left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^6 \right] & r < r_c \\ 0 & r \geq r_c \end{cases}$$



Ran with a single RTX2080Ti GPU.

# Computational challenges

1 Short range interactions

2 Particle-grid coupling

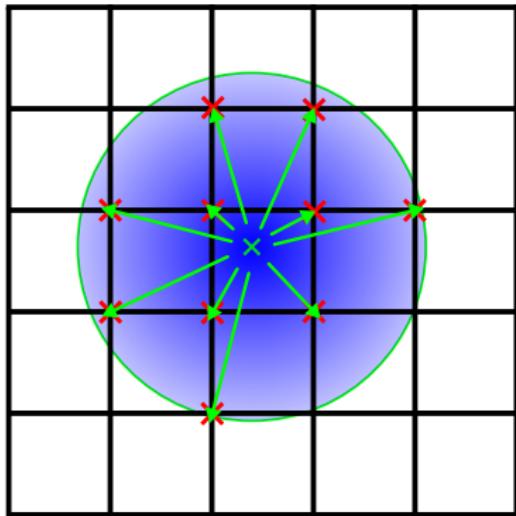
# Computational challenges

1 Short range interactions

2 Particle-grid coupling

Spreading ( $\mathcal{S}$ ):

$$\mathbf{f}(\mathbf{r}) = \mathcal{S}(\mathbf{r})\mathbf{F} := \sum_i \mathbf{F}_i \delta_a(\mathbf{r} - \mathbf{q}_i)$$



Supervector notation  $\mathbf{F} := \{\mathbf{F}_1, \dots, \mathbf{F}_N\}$

# Computational challenges

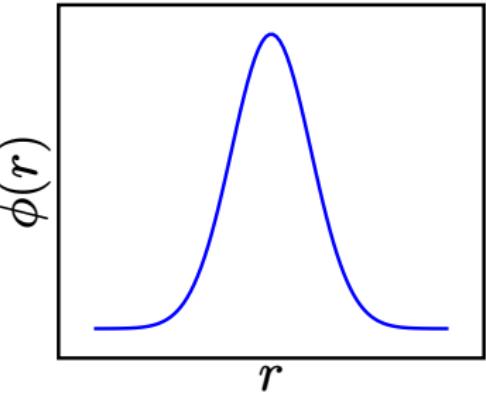
## 1 Short range interactions

## 2 Particle-grid coupling

Spreading ( $\mathcal{S}$ ):

$$\mathbf{f}(\mathbf{r}) = \mathcal{S}(\mathbf{r})\mathbf{F} := \sum_i \mathbf{F}_i \delta_a(\mathbf{r} - \mathbf{q}_i)$$

$$\delta_a(\mathbf{r}) := \phi(r_x)\phi(r_y)\phi(r_z) \rightarrow \text{Smeared delta}$$



Example:  $\phi(r) \propto e^{-\frac{r^2}{2\sigma^2}}$

# Computational challenges

## 1 Short range interactions

## 2 Particle-grid coupling

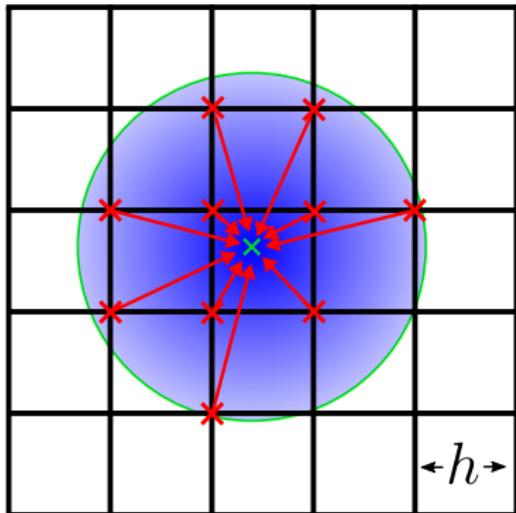
Spreading ( $\mathcal{S}$ ):

$$\mathbf{f}(\mathbf{r}) = \mathcal{S}(\mathbf{r})\mathbf{F} := \sum_i \mathbf{F}_i \delta_a(\mathbf{r} - \mathbf{q}_i)$$

$\delta_a(\mathbf{r}) := \phi(r_x)\phi(r_y)\phi(r_z) \rightarrow$  Smeared delta

Interpolation ( $\mathcal{J}$ ):

$$\mathbf{u}_i = \mathcal{J}_{\mathbf{q}_i} \mathbf{v}(\mathbf{r}) = \int \mathbf{v}(\mathbf{r}) \delta_a(\mathbf{r} - \mathbf{q}_i) d\mathbf{r}$$



# Computational challenges

1 Short range interactions

2 Particle-grid coupling

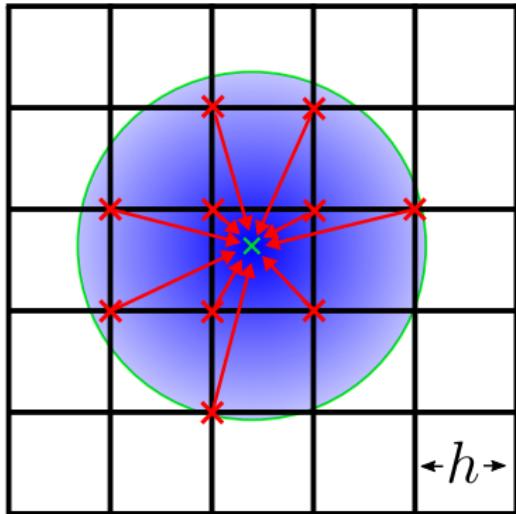
Spreading ( $\mathcal{S}$ ):

$$\mathbf{f}(\mathbf{r}) = \mathcal{S}(\mathbf{r})\mathbf{F} := \sum_i \mathbf{F}_i \delta_a(\mathbf{r} - \mathbf{q}_i)$$

$\delta_a(\mathbf{r}) := \phi(r_x)\phi(r_y)\phi(r_z) \rightarrow$  Smeared delta

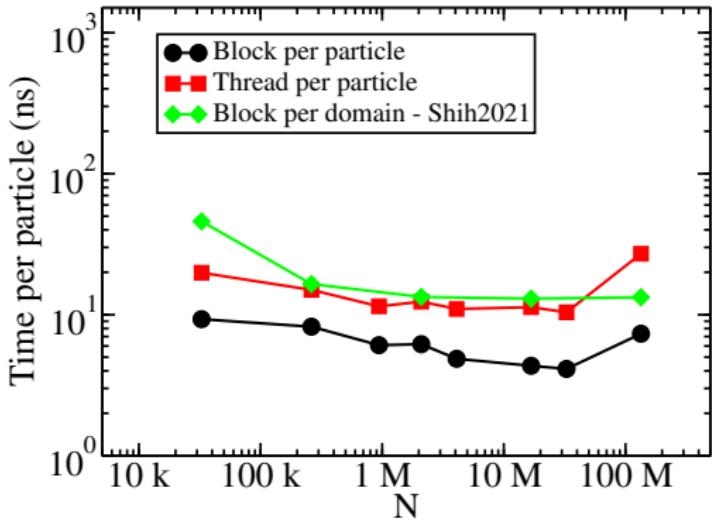
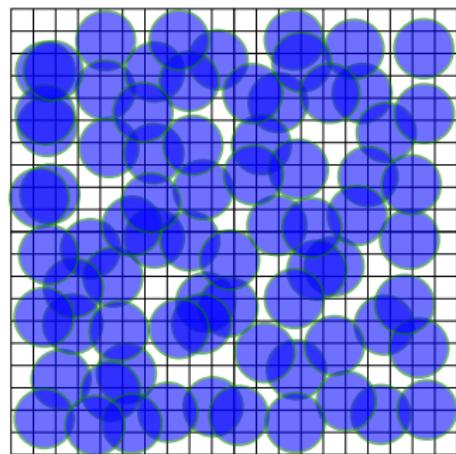
Interpolation ( $\mathcal{J}$ ):

$$\mathbf{u}_i = \mathcal{J}_{\mathbf{q}_i} \mathbf{v}(\mathbf{r}) \approx \sum_n \mathbf{v}_n \delta_a(\mathbf{r}_n - \mathbf{q}_i) h^3$$



# Particle-grid coupling

## Performance



3D random distribution. Ran with a single RTX2080Ti GPU.

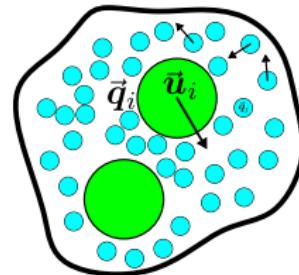
# Talk outline

- 1 Introduction
- 2 Computational challenges
- 3 The Force Coupling Method
- 4 UAMMD
- 5 Examples

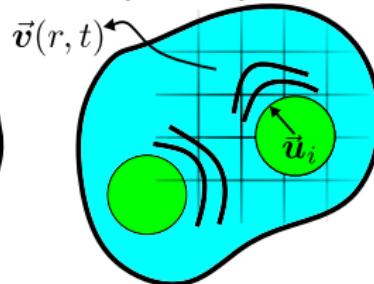
# Hydrodynamics

Algorithms for Brownian Dynamics with Hydrodynamic Interactions (BDHI)

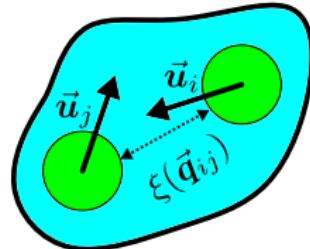
Microscopic



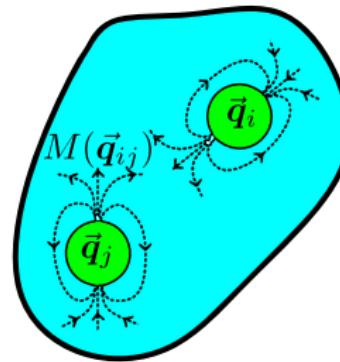
Hydrodynamic



Langevin



Smoluchowski



# Complex fluids

The spatio-temporal landscape of numerical techniques.

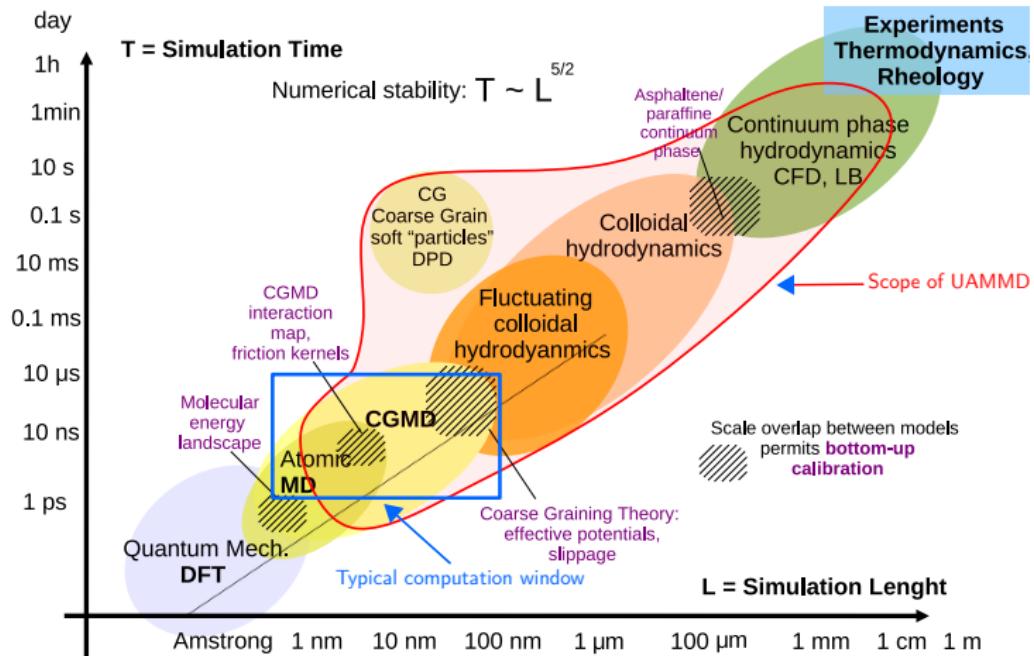
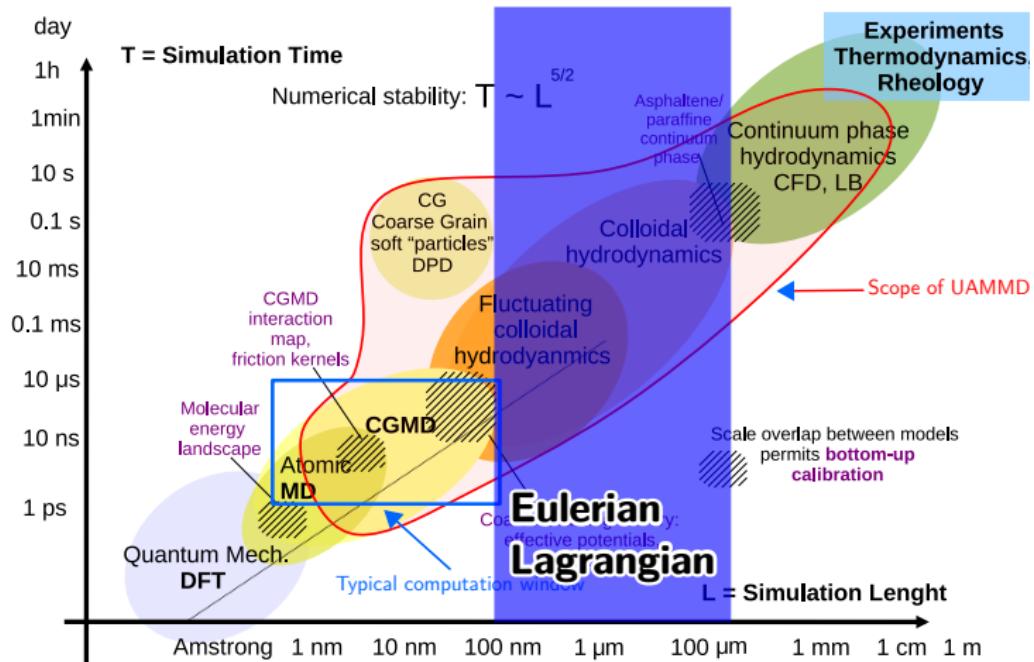


Figure: Figure courtesy of Rafael Delgado-Buscalioni.

# Complex fluids

The spatio-temporal landscape of numerical techniques.



**Figure:** Figure courtesy of Rafael Delgado-Buscalioni.

# Smoluchowski level

Brownian Dynamics with Hydrodynamics Interactions (BDHI)

$$dq = \mathcal{M}F dt + \sqrt{2k_B T \mathcal{M}} d\tilde{W} + k_B T \partial_q \cdot \mathcal{M} dt.$$

- $\mathcal{M}(q)$ :  
Mobility tensor. Configuration- and BC-dependent

# Smoluchowski level

Brownian Dynamics with Hydrodynamics Interactions (BDHI)

$$dq = \mathcal{M}F dt + \sqrt{2k_B T \mathcal{M}} d\tilde{W} + k_B T \partial_q \cdot \mathcal{M} dt.$$

- $\mathcal{M}(q)$ :  
Mobility tensor. Configuration- and BC-dependent
- $d\tilde{W}$ : Brownian motion

# Smoluchowski level

Brownian Dynamics with Hydrodynamics Interactions (BDHI)

$$dq = \mathcal{M}F dt + \sqrt{2k_B T \mathcal{M}} d\tilde{W} + k_B T \partial_q \cdot \mathcal{M} dt.$$

- $\mathcal{M}(q)$ :  
Mobility tensor. Configuration- and BC-dependent
- $d\tilde{W}$ : Brownian motion
- $k_B T \partial_q \cdot \mathcal{M} dt$ :  
Thermal drift. Zero depending on the BCs

# Smoluchowski level

Brownian Dynamics with Hydrodynamics Interactions (BDHI)

$$dq = \mathcal{M}F dt + \sqrt{2k_B T \mathcal{M}} d\tilde{W} + k_B T \partial_q \cdot \mathcal{M} dt.$$

## Problems

# Smoluchowski level

Brownian Dynamics with Hydrodynamics Interactions (BDHI)

$$dq = \mathcal{M}F dt + \sqrt{2k_B T \mathcal{M}} d\tilde{W} + k_B T \partial_q \cdot \mathcal{M} dt.$$

## Problems

- $\mathcal{M}F$ : Matrix-vector multiplication is  $O(N^2)$

$$dq = \mathcal{M}F dt + \sqrt{2k_B T \mathcal{M}} d\tilde{W} + k_B T \partial_q \cdot \mathcal{M} dt.$$

## Problems

- $\mathcal{M}F$ : Matrix-vector multiplication is  $O(N^2)$
- $\sqrt{\mathcal{M}}$ : Quite expensive in general.  $O(N^{[2.25-3]})$

$$dq = \mathcal{M}F dt + \sqrt{2k_B T \mathcal{M}} d\tilde{W} + k_B T \partial_q \cdot \mathcal{M} dt.$$

## Problems

- $\mathcal{M}F$ : Matrix-vector multiplication is  $O(N^2)$
- $\sqrt{\mathcal{M}}$ : Quite expensive in general.  $O(N^{[2.25-3]})$
- $\mathcal{M}$  might be unknown/not analytical.

# Hydrodynamics

## Fluctuating Stokes equations

### Stokes equations

$$\nabla \pi - \eta \nabla^2 \mathbf{v} = \mathbf{f} + \nabla \cdot \mathcal{Z}$$

$$\nabla \cdot \mathbf{v} = 0$$

# Hydrodynamics

## Fluctuating Stokes equations

### Stokes equations

$$\nabla \pi - \eta \nabla^2 \mathbf{v} = \mathbf{f} + \nabla \cdot \mathcal{Z}$$
$$\nabla \cdot \mathbf{v} = 0$$

### Fluid forcing

$$\mathbf{f}(\mathbf{r}) = \mathcal{S}(\mathbf{r}) \mathbf{F}$$

### Particle velocity

$$\mathbf{u}_i = \mathcal{J}_{q_i} \mathbf{v}(\mathbf{r})$$

# Hydrodynamics

Fluctuating Stokes equations

Stokes equations

$$\begin{aligned}\nabla \pi - \eta \nabla^2 \mathbf{v} &= \mathbf{f} + \nabla \cdot \mathcal{Z} \\ \nabla \cdot \mathbf{v} &= 0\end{aligned}$$

Fluid forcing

$$\mathbf{f}(\mathbf{r}) = \mathcal{S}(\mathbf{r}) \mathbf{F}$$

Particle velocity

$$\mathbf{u}_i = \mathcal{J}_{q_i} \mathbf{v}(\mathbf{r})$$

## Green formalism

$$\mathbf{v}(\mathbf{r}) = \mathcal{G} \tilde{\mathbf{f}}$$

# Hydrodynamics

Fluctuating Stokes equations

## Stokes equations

$$\begin{aligned}\nabla \pi - \eta \nabla^2 \mathbf{v} &= \mathbf{f} + \nabla \cdot \mathcal{Z} \\ \nabla \cdot \mathbf{v} &= 0\end{aligned}$$

## Fluid forcing

$$\mathbf{f}(\mathbf{r}) = \mathcal{S}(\mathbf{r}) \mathbf{F}$$

## Particle velocity

$$\mathbf{u}_i = \mathcal{J}_{q_i} \mathbf{v}(\mathbf{r})$$

## Green formalism

$$\mathbf{v}(\mathbf{r}) = \mathcal{G} \tilde{\mathbf{f}} := \int \mathcal{G}(\mathbf{r}, \mathbf{r}') \tilde{\mathbf{f}}(\mathbf{r}') d\mathbf{r}'$$

# Hydrodynamics

## Fluctuating Stokes equations

### Stokes equations

$$\begin{aligned}\nabla \pi - \eta \nabla^2 \mathbf{v} &= \mathbf{f} + \nabla \cdot \mathcal{Z} \\ \nabla \cdot \mathbf{v} &= 0\end{aligned}$$

### Fluid forcing

$$\mathbf{f}(\mathbf{r}) = \mathcal{S}(\mathbf{r}) \mathbf{F}$$

### Particle velocity

$$\mathbf{u}_i = \mathcal{J}_{q_i} \mathbf{v}(\mathbf{r})$$

## Green formalism

$$\mathbf{v}(\mathbf{r}) = \mathcal{G} \tilde{\mathbf{f}} := \int \mathcal{G}(\mathbf{r}, \mathbf{r}') \tilde{\mathbf{f}}(\mathbf{r}') d\mathbf{r}'$$

### Force Coupling Method (FCM)

$$\frac{d\mathbf{q}_i}{dt} = \mathbf{u}_i = \mathcal{J}_{q_i} \mathcal{G}(\mathcal{S} \mathbf{F} + \nabla \cdot \mathcal{Z})$$

# Hydrodynamics

## Fluctuating Stokes equations

### Stokes equations

$$\begin{aligned}\nabla \pi - \eta \nabla^2 \mathbf{v} &= \mathbf{f} + \nabla \cdot \mathcal{Z} \\ \nabla \cdot \mathbf{v} &= 0\end{aligned}$$

### Fluid forcing

$$\mathbf{f}(\mathbf{r}) = \mathcal{S}(\mathbf{r}) \mathbf{F}$$

### Particle velocity

$$\mathbf{u}_i = \mathcal{J}_{q_i} \mathbf{v}(\mathbf{r})$$

## Green formalism

$$\mathbf{v}(\mathbf{r}) = \mathcal{G} \tilde{\mathbf{f}} := \int \mathcal{G}(\mathbf{r}, \mathbf{r}') \tilde{\mathbf{f}}(\mathbf{r}') d\mathbf{r}'$$

### Force Coupling Method (FCM)

$$\frac{d\mathbf{q}_i}{dt} = \mathbf{u}_i = \mathcal{J}_{q_i} \mathcal{G}(\mathcal{S} \mathbf{F} + \nabla \cdot \mathcal{Z})$$

$$\mathcal{M} = \mathcal{J} \mathcal{G} \mathcal{S}$$

$$\mathcal{M}^{1/2} = \mathcal{J} \mathcal{G} \nabla \cdot$$

$$\partial_q \cdot \mathcal{M} = 0 \leftarrow \text{Incompressible}$$

# Hydrodynamics

## Fluctuating Stokes equations

### Stokes equations

$$\begin{aligned}\nabla \pi - \eta \nabla^2 \mathbf{v} &= \mathbf{f} + \nabla \cdot \mathcal{Z} \\ \nabla \cdot \mathbf{v} &= 0\end{aligned}$$

### Fluid forcing

$$\mathbf{f}(\mathbf{r}) = \mathcal{S}(\mathbf{r}) \mathbf{F}$$

### Particle velocity

$$\mathbf{u}_i = \mathcal{J}_{q_i} \mathbf{v}(\mathbf{r})$$

## Green formalism

$$\mathbf{v}(\mathbf{r}) = \mathcal{G} \tilde{\mathbf{f}} := \int \mathcal{G}(\mathbf{r}, \mathbf{r}') \tilde{\mathbf{f}}(\mathbf{r}') d\mathbf{r}'$$

### Force Coupling Method (FCM)

$$\frac{d\mathbf{q}_i}{dt} = \mathbf{u}_i = \mathcal{J}_{q_i} \mathcal{G}(\mathcal{S} \mathbf{F} + \nabla \cdot \mathcal{Z})$$

$$\mathcal{M}_{ij} = \iint \delta_a(\mathbf{r} - \mathbf{q}_j) \mathcal{G}(\mathbf{r}, \mathbf{r}') \delta_a(\mathbf{q}_i - \mathbf{r}') d\mathbf{r}' d\mathbf{r}$$

# Hydrodynamics

## Fluctuating Stokes equations

### Stokes equations

$$\begin{aligned}\nabla \pi - \eta \nabla^2 \mathbf{v} &= \mathbf{f} + \nabla \cdot \mathcal{Z} \\ \nabla \cdot \mathbf{v} &= 0\end{aligned}$$

### Fluid forcing

$$\mathbf{f}(\mathbf{r}) = \mathcal{S}(\mathbf{r}) \mathbf{F}$$

### Particle velocity

$$\mathbf{u}_i = \mathcal{J}_{q_i} \mathbf{v}(\mathbf{r})$$

## Green formalism

$$\mathbf{v}(\mathbf{r}) = \mathcal{G} \tilde{\mathbf{f}} := \int \mathcal{G}(\mathbf{r}, \mathbf{r}') \tilde{\mathbf{f}}(\mathbf{r}') d\mathbf{r}'$$

### Force Coupling Method (FCM)

$$\frac{d\mathbf{q}_i}{dt} = \mathbf{u}_i = \mathcal{J}_{q_i} \mathcal{G}(\mathcal{S} \mathbf{F} + \nabla \cdot \mathcal{Z})$$

$$\mathcal{G} := -\eta^{-1} \nabla^{-2} \left( \mathbb{I} - \nabla \nabla^{-2} \nabla \right)$$

# Hydrodynamics

## Fluctuating Stokes equations

### Stokes equations

$$\begin{aligned}\nabla \pi - \eta \nabla^2 \mathbf{v} &= \mathbf{f} + \nabla \cdot \mathcal{Z} \\ \nabla \cdot \mathbf{v} &= 0\end{aligned}$$

### Fluid forcing

$$\mathbf{f}(\mathbf{r}) = \mathcal{S}(\mathbf{r}) \mathbf{F}$$

### Particle velocity

$$\mathbf{u}_i = \mathcal{J}_{q_i} \mathbf{v}(\mathbf{r})$$

## Green formalism

$$\mathbf{v}(\mathbf{r}) = \mathcal{G} \tilde{\mathbf{f}} := \int \mathcal{G}(\mathbf{r}, \mathbf{r}') \tilde{\mathbf{f}}(\mathbf{r}') d\mathbf{r}'$$

### Force Coupling Method (FCM)

$$\frac{d\mathbf{q}_i}{dt} = \mathbf{u}_i = \mathcal{J}_{q_i} \mathcal{G}(\mathcal{S} \mathbf{F} + \nabla \cdot \mathcal{Z})$$

$$\mathcal{G} := -\eta^{-1} \nabla^{-2} \left( \mathbb{I} - \nabla \nabla^{-2} \nabla \right)$$

### Open boundaries

$$\mathcal{G} \rightarrow \mathcal{O}(\mathbf{r}) = \frac{1}{8\pi\eta r} \left( \mathbb{I} - \frac{\mathbf{r} \otimes \mathbf{r}}{r^2} \right)$$

# Hydrodynamics

## Triply periodic FCM

$$\mathbf{u}_i = \mathcal{J}_{\mathbf{q}_i} \mathcal{G}(\mathcal{S}\mathbf{F} + \nabla \cdot \mathcal{Z})$$

# Hydrodynamics

Triply periodic FCM

$$\mathbf{u}_i = \mathcal{J}_{\mathbf{q}_i} \mathcal{G}(\mathcal{S}\mathbf{F} + \nabla \cdot \mathcal{Z})$$



$\mathfrak{F}$ : Fourier transform

$$\mathbf{u}_i = \mathcal{J}_{\mathbf{q}_i} \quad \mathcal{G}( \mathcal{S}\mathbf{F} + \nabla \cdot \mathcal{Z} )$$

# Hydrodynamics

Triply periodic FCM

$$\mathbf{u}_i = \mathcal{J}_{\mathbf{q}_i} \mathcal{G}(\mathcal{S} \mathbf{F} + \nabla \cdot \mathcal{Z})$$



$\mathfrak{F}$ : Fourier transform

$$\mathbf{u}_i = \mathcal{J}_{\mathbf{q}_i} \mathfrak{F}^{-1} \widehat{\mathcal{G}}(\mathfrak{F} \mathcal{S} \mathbf{F} + \mathbf{k} \cdot \mathcal{Z}_k)$$

### Force Coupling Method

$$\boldsymbol{u}_i = \mathcal{J}_{\boldsymbol{q}_i} \mathfrak{F}^{-1} \widehat{\mathcal{G}} (\mathfrak{F} \mathcal{S} \boldsymbol{F} + \boldsymbol{k} \cdot \mathcal{Z}_{\boldsymbol{k}})$$

### Triply Periodic

$$\widehat{\mathcal{G}}_{3D} := \eta^{-1} k^{-2} \left( \mathbb{I} - \frac{\boldsymbol{k} \otimes \boldsymbol{k}}{k^2} \right)$$

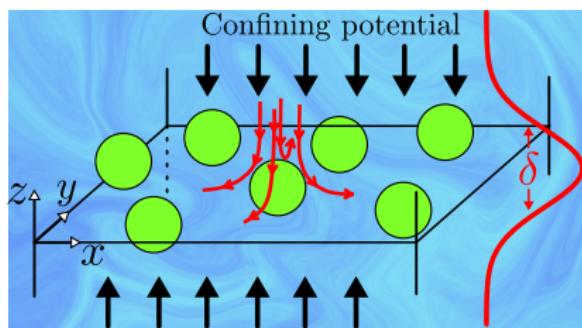
# Hydrodynamics in confined geometries

New algorithms for Brownian Dynamics with Hydrodynamic Interactions (BDHI)

## Force Coupling Method

$$\mathbf{u}_i = \mathcal{J}_{\mathbf{q}_i} \mathfrak{F}^{-1} \hat{\mathcal{G}}_{3D} (\mathfrak{F} \mathcal{S} \mathbf{F} + \mathbf{k} \cdot \mathcal{Z})$$

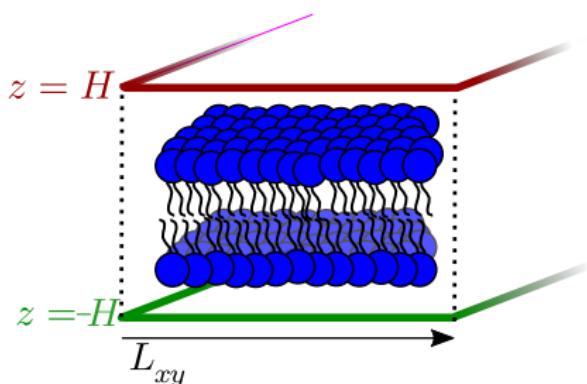
- Quasi two-dimensional (q2D)



q2D:  $\delta \rightarrow 0$ .

3D fluid, 2D particles

- Doubly Periodic Stokes (DPStokes)



# Triply Periodic Electrostatics

Force Coupling Method for the Poisson equation

## Poisson

$$\varepsilon_0 \nabla^2 \phi = -f(\mathbf{r}) = -\mathcal{S}(\mathbf{r})\mathbf{Q}$$

## Spreading

$$\mathcal{S}(\mathbf{r})\mathbf{Q} = \sum_i Q_i \delta_a(\mathbf{q}_i - \mathbf{r})$$

$$\mathbf{E}(\mathbf{r}) = \partial_r \phi \rightarrow \hat{\mathbf{E}} = i\mathbf{k}\hat{\phi}$$

$$\mathbf{F}_i = Q_i \mathcal{J}_{\mathbf{q}_i} \mathbf{E}(\mathbf{r})$$

## Gaussian sources

$$\delta_a(\mathbf{r}) \propto e^{-r^2/(2a^2)}$$

## Charges to forces

$$\mathbf{F}_i = Q_i \mathcal{J}_{\mathbf{q}_i} \mathfrak{F}^{-1} i\mathbf{k} \hat{\mathcal{G}}_P \mathfrak{F} \mathcal{S} \mathbf{Q}$$

Supervector notation. Charges  $\mathbf{Q} := \{Q_1, \dots, Q_N\}$

$$\hat{\mathcal{G}}_P = (\varepsilon_0 k^2)^{-1}$$

[O. Maxian, Raul P. Pelaez et. al. J. Chem. Phys. 2021.]

# Talk outline

1 Introduction

2 Computational challenges

3 The Force Coupling Method

4 UAMMD

- Basic code structure

5 Examples

# Universally Adaptable Multiscale Molecular Dynamics

What makes UAMMD stand out

## UAMMD: A CUDA/C++ library for complex fluids.

---

- Header only and library-like

# Universally Adaptable Multiscale Molecular Dynamics

What makes UAMMD stand out

## UAMMD: A CUDA/C++ library for complex fluids.

---

- Header only and library-like
- Lightweight with minimal dependencies

# Universally Adaptable Multiscale Molecular Dynamics

What makes UAMMD stand out

## UAMMD: A CUDA/C++ library for complex fluids.

---

- Header only and library-like
- Lightweight with minimal dependencies
- Hackable

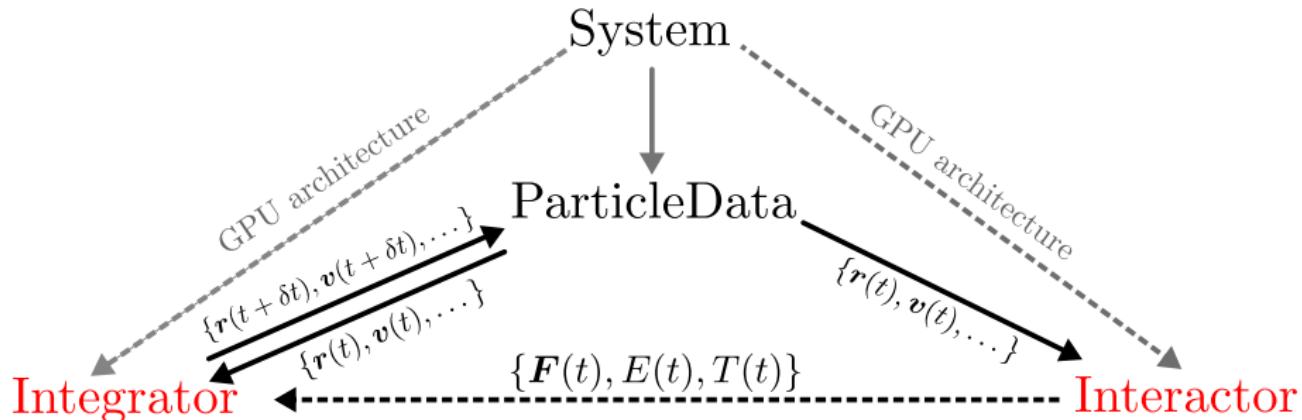
# Universally Adaptable Multiscale Molecular Dynamics

What makes UAMMD stand out

## UAMMD: A CUDA/C++ library for complex fluids.

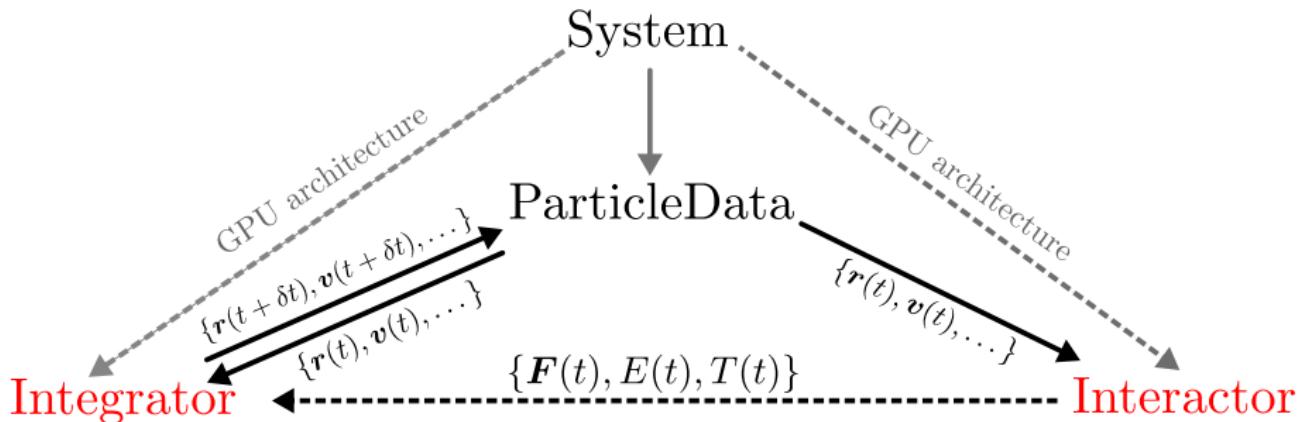
---

- Header only and library-like
- Lightweight with minimal dependencies
- Hackable
- Focus on hydrodynamics

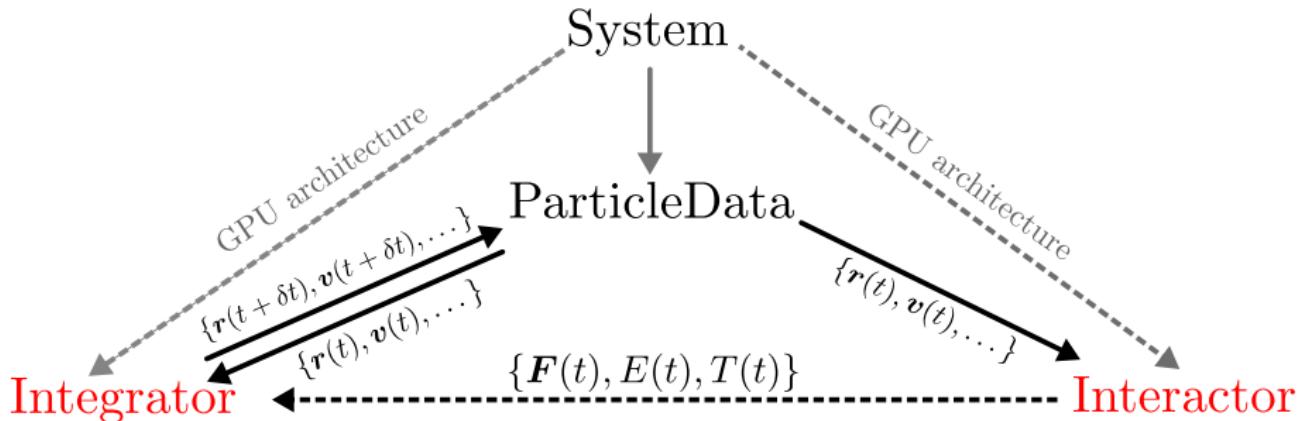


Remember

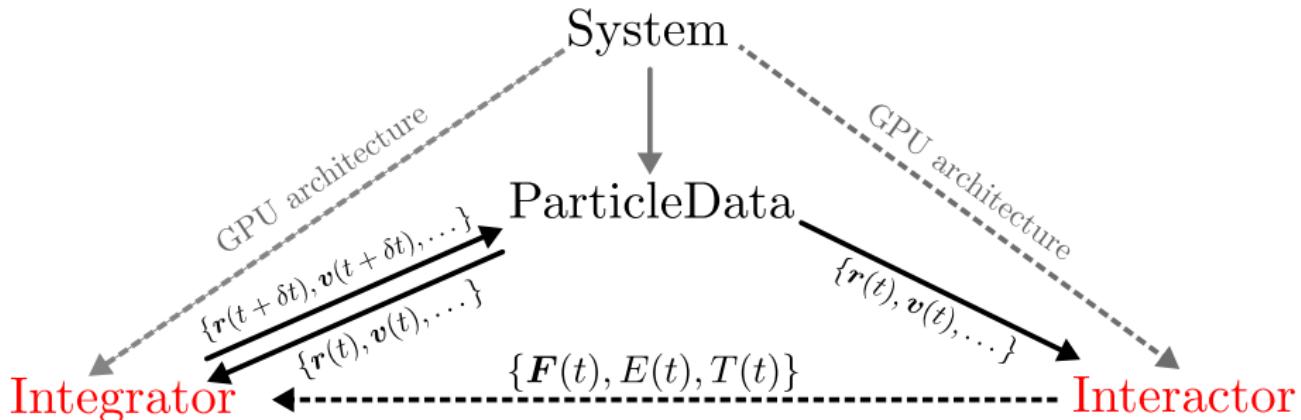
“Interacting *particles* with a state that *evolves*”



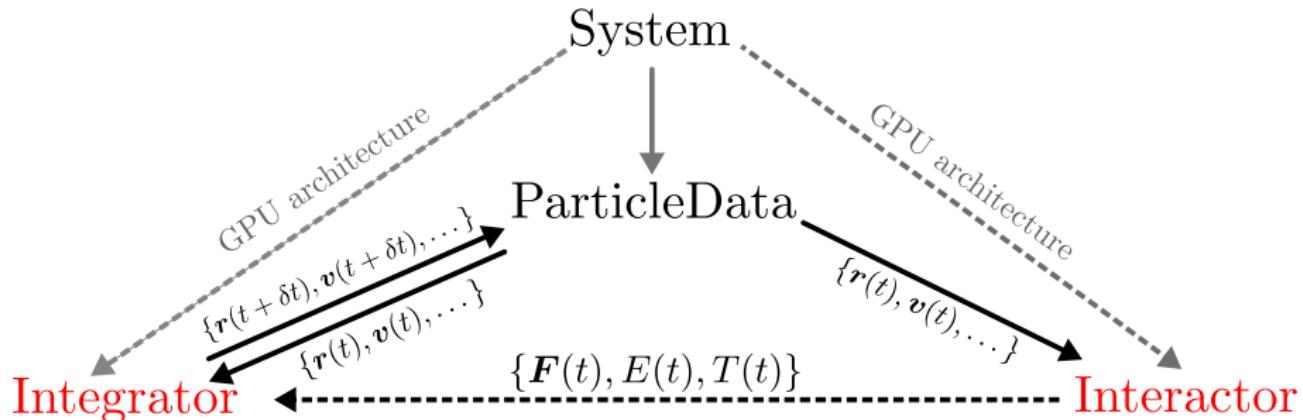
```
#include <uammd.cuh>
int main(int argc, char* argv[]){
    auto sys = std::make_shared<System>(argc, argv);
    ...
}
```



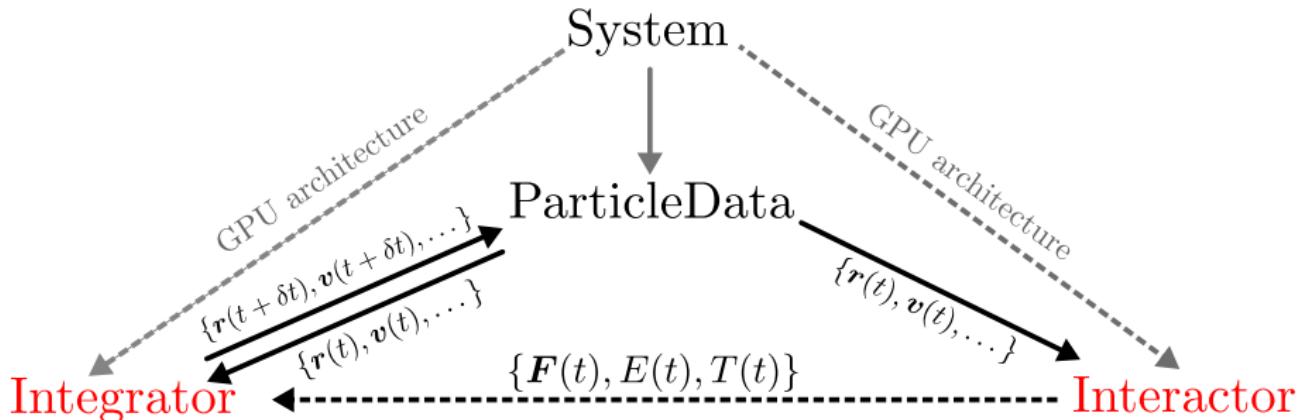
```
#include <uammd.cuh>
int main(int argc, char* argv[]){
    auto sys = std::make_shared<System>(argc, argv);
    const int nP = 1e6; //Number of particles
    auto pd = std::make_shared<ParticleData>(nP, sys);
    ...
}
```



```
...
auto pos = pd->getPos(access::cpu, access::write);
pos[0] = {1,1,1,0}; // x,y,x, color (type)
...
```

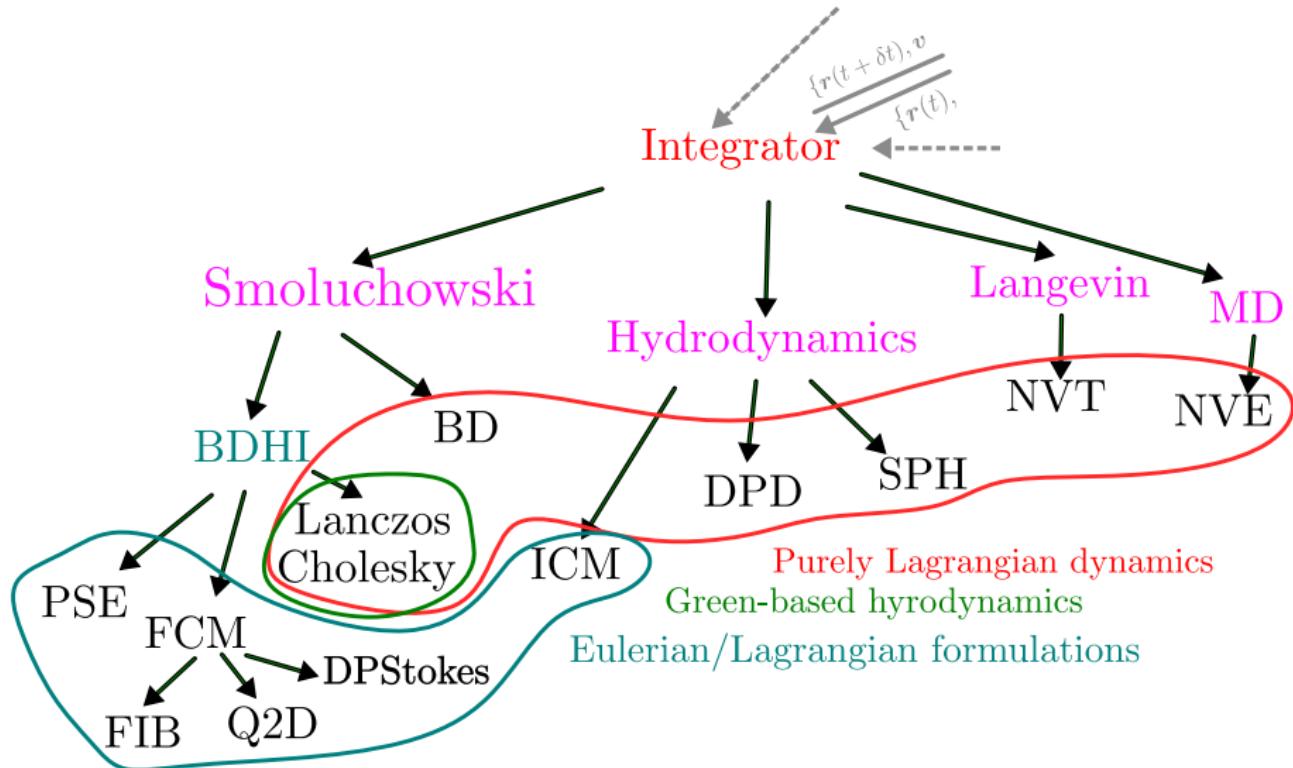


```
...
auto bd = std::make_shared<BDHI::FCM>(pd, /*Parameters*/);
bd->forwardTime(); //Exposed by every Integrator
...
```

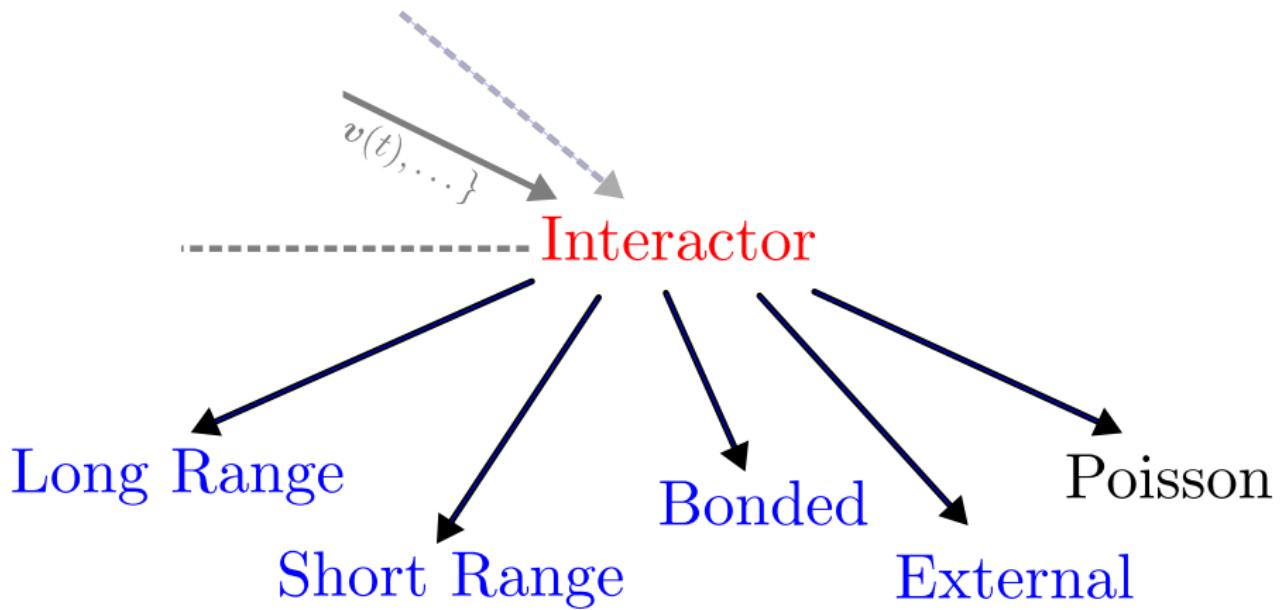


```
...
auto elec = std::make_shared<Poisson>(pd, /*Parameters*/);
elec->sum({.force=true, .energy=false, .virial=false});
//Exposed by every Integrator
bd->addInteractor(elec);
bd->forwardTime();
...
```

# Available solvers



## Available interactions



# UAMMD's online presence

## Git repository

<https://github.com/RaulPPelaez/uammd>

The screenshot shows the GitHub repository page for 'RaulPPelaez / UAMMD'. The repository is public and has 11 branches and 0 tags. The 'Code' tab is selected. A list of commits is shown, with the most recent being a merge from 'docs' into 'v2.x' by RaulPPelaez 3 days ago. Other commits include updates to .res, docs, examples, extensions, src, test, .gitignore, .gitmodules, CHANGELOG.txt, LICENSE.txt, README.md, and compile\_flags.txt. The 'About' section describes UAMMD as a CUDA project for Molecular Dynamics, Brownian Dynamics, and Hydrodynamics, intended to simulate a very generic system using modules. It includes tags for cuda, molecular-dynamics, hydrodynamics, and cuda-molecular-dynamics. The 'Readme' section links to the file, and the 'License' section shows it uses the GPL-3.0 license. The repository has 25 stars, 4 watchers, and 7 forks. The 'Releases' section indicates no releases have been published, with a link to 'Create a new release'. The 'Packages' section shows no packages have been published, with a link to 'Publish your first package'. The footer features the text 'Universally Adaptable Multiscale Molecular'.

RaulPPelaez Merge branch 'docs' into v2.x 6ea244c 3 days ago 798 commits

.res Add images to README.md 4 years ago

docs Docs: Update sketch image 3 days ago

examples Examples: Update CUDA root detection in Makefiles last month

extensions @ b8f6346 Add preamble.h to extensions 6 months ago

src Merge branch 'docs' into v2.x 3 days ago

test Update tests to new interfaces last month

.gitignore Update gitignore 2 months ago

.gitmodules Change default uammd-extensions submodule to https 12 months ago

CHANGELOG.txt Update CHANGELOG 3 months ago

LICENSE.txt Add LICENSE.txt 4 years ago

README.md Add Doc badge to README.md 2 months ago

compile\_flags.txt Refractor the new Integrator/Interactor creation into the wh... 5 months ago

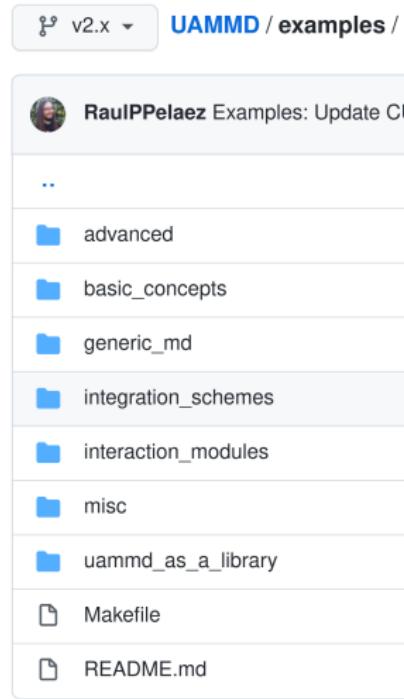
README.md

Universally Adaptable Multiscale Molecular

# UAMMD's online presence

Git repository

<https://github.com/RaulPPelaez/uammd>



A screenshot of a GitHub repository page for 'RaulPPelaez Examples: Update CI'. The page shows a list of files and folders:

- ..
- advanced
- basic\_concepts
- generic\_md
- integration\_schemes
- interaction\_modules
- misc
- uammd\_as\_a\_library
- Makefile
- README.md

# UAMMD's online presence

Git repository

<https://github.com/RaulPPelaez/uammd>

The image shows two side-by-side screenshots of GitHub repository pages for "UAMMD / examples". Both pages have a dropdown menu set to "v2.x".

**Left Repository:** "RaulPPelaez Examples: Update CUDA root directory" (private)

- ..
- 1-system.cu
- 10-initial\_configuration.cu
- 11-measuring\_things.cu
- 2-hello\_world.cu
- 3-more\_system.cu
- 4-uammd\_types.cu
- 5-particle\_data.cu
- 6-particle\_data2.cu
- 7-moving\_particles.cu
- 8-interacting\_particles.cu
- 9-reading\_parameters.cu
- Makefile
- README.md

**Right Repository:** "RaulPPelaez Examples: Update CI" (private)

- ..
- advanced
- basic\_concepts
- generic\_md
- integration\_schemes
- interaction\_modules
- misc
- uammd\_as\_a\_library
- Makefile
- README.md

# UAMMD's online presence

Git repository

<https://github.com/RaulPPelaez/uammd>

The image shows two GitHub code editor panes side-by-side. Both panes have a 'v2.x' dropdown at the top left and a 'History' button at the top right.

**Left Pane:** The title is 'UAMMD / examples / basic\_concepts /'. It lists several files: 1-system.cu, 10-initial\_configuration.cu, 11-measuring\_things.cu, 2-hello\_world.cu (which is selected), 3-more\_system.cu, 4-uammd\_types.cu, 5-particle\_data.cu, 6-particle\_data2.cu, 7-moving\_particles.cu, 8-interacting\_particles.cu, 9-reading\_parameters.cu, Makefile, and README.md.

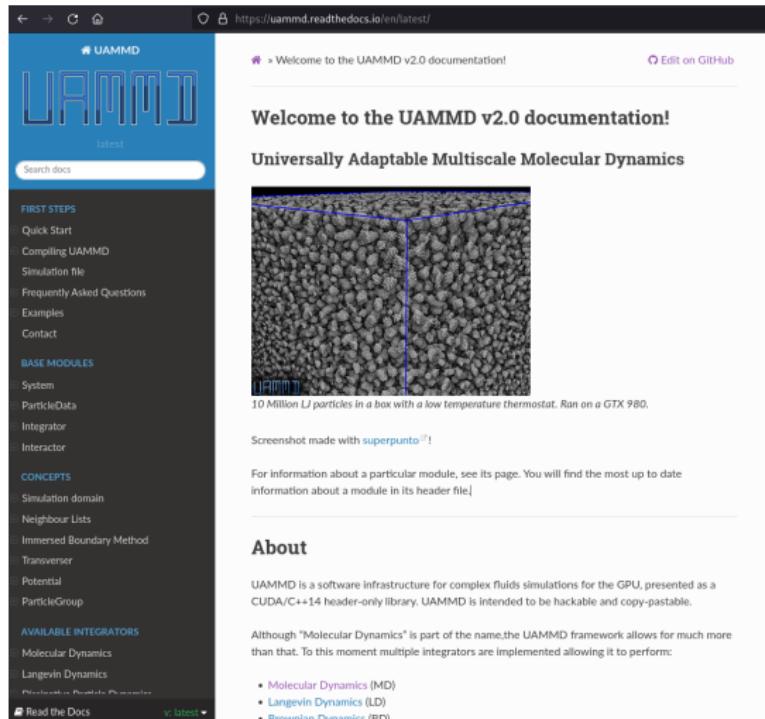
**Right Pane:** The title is 'UAMMD / examples / basic\_concepts / 2-hello\_world.cu'. The code content is as follows:

```
36 lines (35 sloc) | 2.2 KB
1 // Raul P. Pelaez 2021
2 // Hello world with UAMMD
3 // We will start from the previous example and see the first utility provided by System, this
4 // System::log allows to, well, log messages with several levels of priority.
5 // Internally UAMMD modules use this utility to print everything ranging from informational
6 // In this example we will use System::log to print a simple message along with the date.
7 */
8
9 //uammd.cuh is the basic uammd include containing, among other things, the System struct.
10 #include "uammd.cuh"
11 #include <ctime> //For time and ctime
12 using namespace uammd;
13 //The main function will initialize an UAMMD environment, print a message, then destroy it.
14 int main(int argc, char* argv[]){
15     //Initialize System
16     auto sys = std::make_shared<System>(argc, argv);
17     //Unless something goes wrong System creation logs messages using level MESSAGE, which prints
18     //There are a lot more levels to choose from, each associated with a number. From highest
19     //CRITICAL=0, ERROR, EXCEPTION, WARNING, MESSAGE, STDERR, STDOUT, DEBUG1, DEBUG2, DEBUG3
20     //For example, MESSAGE is associated with log level number 5.
21     //Let's print something using the MESSAGE level:
22     sys->logSystem("Hello from UAMMD");
23     //Let's also print today's date, this time as a WARNING:
24     auto currentTime = time(nullptr);
25     //Notice that System::log works as C's printf, with a format string and then arguments.
26     sys->logSystem(WARNING,"Current time is: %s",ctime(&currentTime));
27     //The maximum log level printed can be controlled through the MAXLOGLEVEL compile macro (which is
28     //CRITICAL=0, so 5, which will print up to MESSAGE.
29     //The special level CRITICAL will terminate the execution of the program with an error code.
30     //Unless the log level STDOUT is used, all messages will be issued to stderr.
31     //Since it is known at compile time, any log calls with levels above the maximum one will
32     //no performance penalty
33     //Destroy the UAMMD environment and exit
34     sys->finish();
35     return 0;
36 }
```

# UAMMD's online presence

## Documentation

<https://uammd.readthedocs.io>



The screenshot shows the homepage of the UAMMD v2.0 documentation. At the top, there is a navigation bar with icons for back, forward, search, and refresh, followed by the URL <https://uammd.readthedocs.io/en/latest/>. Below the URL is the UAMMD logo and the word "latest". To the right of the logo is a "Search docs" input field. On the far right of the header is a "Edit on GitHub" link.

The main content area has a blue header with the text "Welcome to the UAMMD v2.0 documentation!" and "Universally Adaptable Multiscale Molecular Dynamics". Below this is a large image showing a simulation of 10 million LJ particles in a box with a low temperature thermostat, run on a GTX 980. A blue rectangular selection box highlights a portion of the particle distribution.

Below the image, the text reads: "10 Million LJ particles in a box with a low temperature thermostat. Run on a GTX 980." and "Screenshot made with superpunto<sup>®</sup>!"

Further down, there is a section titled "About" which contains the following text: "UAMMD is a software infrastructure for complex fluids simulations for the GPU, presented as a CUDA/C++14 header-only library. UAMMD is intended to be hackable and copy-pastable." and "Although "Molecular Dynamics" is part of the name, the UAMMD framework allows for much more than that. To this moment multiple integrators are implemented allowing it to perform:" followed by a bulleted list: • Molecular Dynamics (MD) • Langevin Dynamics (LD) • Brownian Dynamics (BD)

On the left side of the page, there is a sidebar with a dark background containing a list of links under categories: "FIRST STEPS", "BASE MODULES", "CONCEPTS", and "AVAILABLE INTEGRATORS".

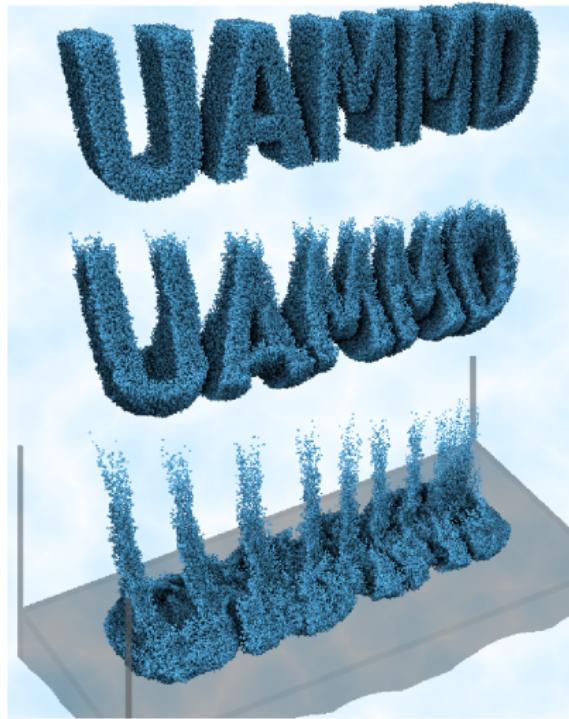
# Talk outline

- 1 Introduction
- 2 Computational challenges
- 3 The Force Coupling Method
- 4 UAMMD
- 5 Examples

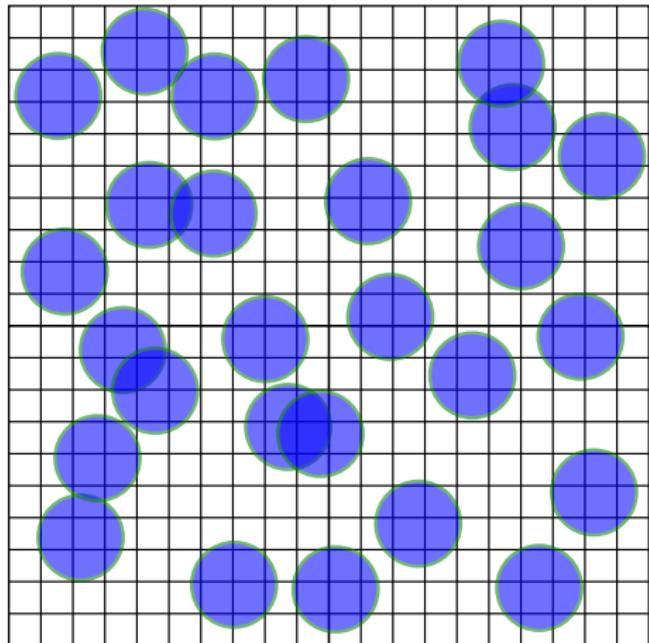
# Lets clone

```
$ git clone git@github.com:RaulPPelaez/2022Bilbao
```

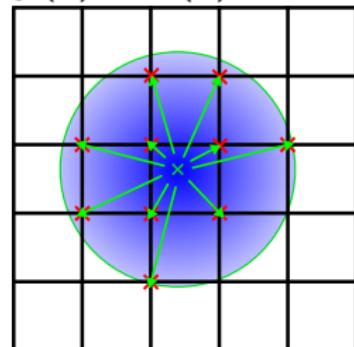
# The falling logo



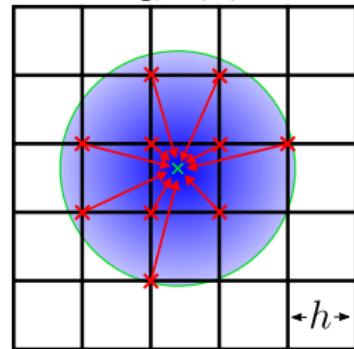
# Particle-grid coupling



$$\mathbf{f}(\mathbf{r}) = \mathcal{S}(\mathbf{r})\mathbf{F}$$



$$\mathbf{u}_i = \mathcal{J}_{q_i} \mathbf{v}(\mathbf{r})$$



# Thanks for your attention!