Lattice Boltzmann and Finite Volume Large Eddy Simulations

This document describes two C# classes, `LatticeBoltzmannLES` and `LargeEddySim`, designed for large eddy simulation (LES) of estuarine circulation, incorporating river inflow, tidal forcing, and salinity-driven density variations. Both classes use the Smagorinsky model for subgrid-scale turbulence and provide graphical user interfaces (GUIs) for simulation control and visualization.

LatticeBoltzmannLES

The `LatticeBoltzmannLES` class implements a 2D lattice Boltzmann method (LBM) with the D2Q9 model to simulate estuarine hydrodynamics, capturing velocity, salinity, and turbulence quantities.

Core Components and Initialization

The class is initialized with:

- Grid size: $n_x = 200$, $n_y = 50$ (lattice points)
- Spatial step: $\Delta x = 50\,\mathrm{m}$
- Time step: $\Delta t = 0.1\,\mathrm{s}$ (reserved for future scaling)
- River inflow: $u_{\text{river}} = 0.1\,\mathrm{m/s}$
- Tidal amplitude: $A_{\text{tidal}} = 1.0\,\mathrm{m}$, period: $T_{\text{tidal}} = 43200\,\mathrm{s}$
- Smagorinsky constant: $C_s = 0.1$
- Reynolds number: $Re = 1000$
- Salinity: $S_{\text{ocean}} = 35\,\mathrm{PSU}$, $S_{\text{river}} = 0\,\mathrm{PSU}$
- Ocean temperature: $T_{\text{ocean}} = 20\,^{\circ}\mathrm{C}$

Arrays include:

- Density: $\rho(i,j)$
- Salinity: $S(i,j)$
- Velocity distribution functions: $f(i,j,k)$, equilibrium $f^{\text{eq}}(i,j,k)$ (D2Q9, $k = 0,\ldots,8$)
- Salinity distribution functions: $g(i,j,k)$, equilibrium $g^{\text{eq}}(i,j,k)$
- Velocities: $u(i,j)$, $v(i,j)$; time-averaged: $u_{\text{avg}}(i,j)$, $v_{\text{avg}}(i,j)$
- Vorticity or Q-criterion: $\omega(i,j)$
- Local Smagorinsky constant: $C_s(i,j)$
- Local spatial step: $\Delta x_{\text{local}}(i,j)$, relaxation times: $\tau(i,j)$, $\tau_S(i,j)$

The GUI allows users to adjust parameters and select visualization modes (velocity, salinity, time-averaged velocity), refinement modes (None, River, Tidal, Both), filter width (Lattice Spacing, Cell Area), and vortex visualization (Vorticity Magnitude, Q-Criterion).

Functioning Logic

The simulation follows the LBM algorithm:

1. **Streaming**: Propagate $f(i,j,k)$ and $g(i,j,k)$ along D2Q9 directions:
$$f(i+c_{k,x}, j+c_{k,y}, k) = f(i,j,k), \quad g(i+c_{k,x}, j+c_{k,y}, k) = g(i,j,k) \tag{1}$$
where $c_k = (c_{k,x}, c_{k,y})$ are lattice directions.

2. **Collision**: Update distributions using the Bhatnagar-Gross-Krook (BGK) model:
$$f(i,j,k) \leftarrow f(i,j,k) + \frac{f^{\text{eq}}(i,j,k) - f(i,j,k)}{\tau(i,j)} \tag{2}$$
$$g(i,j,k) \leftarrow g(i,j,k) + \frac{g^{\text{eq}}(i,j,k) - g(i,j,k)}{\tau_S(i,j)} \tag{3}$$
where equilibrium distributions are:
$$f^{\text{eq}}(i,j,k) = w_k \rho \left[1 + 3\frac{\mathbf{u}\cdot\mathbf{c}_k \Delta x}{\Delta x_{\text{local}}} + 4.5\left(\frac{\mathbf{u}\cdot\mathbf{c}_k \Delta x}{\Delta x_{\text{local}}}\right)^2 - 1.5\frac{|\mathbf{u}|^2 \Delta x^2}{\Delta x_{\text{local}}^2}\right] \tag{4}$$
$$g^{\text{eq}}(i,j,k) = w_k S \rho \left[1 + 3\frac{\mathbf{u}\cdot\mathbf{c}_k \Delta x}{\Delta x_{\text{local}}} + 4.5\left(\frac{\mathbf{u}\cdot\mathbf{c}_k \Delta x}{\Delta x_{\text{local}}}\right)^2 - 1.5\frac{|\mathbf{u}|^2 \Delta x^2}{\Delta x_{\text{local}}^2}\right] \tag{5}$$
with weights $w_k = \{4/9, 1/9, \ldots, 1/36\}$.

3. **Macroscopic Variables**: Compute density, velocity, and salinity:
$$\rho(i,j) = \sum_{k=0}^{8} f(i,j,k), \quad u(i,j) = \frac{1}{\rho}\sum_{k=0}^{8} f(i,j,k)c_{k,x}, \quad v(i,j) = \frac{1}{\rho}\sum_{k=0}^{8} f(i,j,k)c_{k,y}, \quad S(i,j) = \frac{1}{\rho}\sum \tag{6}$$

4. **Turbulence Modeling**: Use Smagorinsky LES:
$$\nu_t = (C_s \Delta)^2 |\mathbf{S}|, \quad \tau(i,j) = 3(\nu + \nu_t)c_s^2 \frac{\Delta x}{\Delta x_{\text{local}}} + 0.5 \tag{7}$$
where Δ is the filter width (Δx_{local} or $\sqrt{\Delta x_{\text{local}}^2}$), $|\mathbf{S}|$ is the strain rate magnitude, and $c_s^2 = 1/3$.

5. **Vorticity/Q-Criterion**: Compute vorticity ($|\partial v/\partial x - \partial u/\partial y|$) or Q-criterion:
$$Q = \frac{1}{2}\left(\|\Omega\|^2 - \|\mathbf{S}\|^2\right), \quad \Omega_{xy} = \frac{1}{2}\left(\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y}\right) \tag{8}$$

6. **Boundary Conditions**: Apply river inflow ($u = u_{\text{river}}, S = S_{\text{river}}$) at $x = 0$ and tidal velocity ($u = A_{\text{tidal}} \sin(2\pi t/T_{\text{tidal}})/\Delta x, S = S_{\text{ocean}}$) at $x = n_x - 1$.

7. **Energy Spectrum**: Compute FFT of velocities at mid-y plane every 100 timesteps, outputting power spectral density to `energy_spectrum.txt`.

Visualization

The GUI displays:

- Velocity, salinity, or time-averaged velocity in a color-coded panel (red-blue gradient).
- Vorticity or Q-criterion isosurfaces in a separate panel, colored by velocity or vorticity.
- Console output with time, average velocity, salinity, TKE, dissipation, and vorticity.

LargeEddySim

The `LargeEddySim` class implements a 2D finite volume method for LES, solving Navier-Stokes equations with salinity and temperature effects.

Core Components and Initialization

The class is initialized with:

- Grid size: $n_x = 50$, $n_z = 20$
- Estuary dimensions: $L = 10000\,\text{m}$, $h = 10\,\text{m}$
- Spatial steps: $\Delta x = L/n_x$, $\Delta z = h/n_z$
- Time step: $\Delta t = 1.0\,\text{s}$ (adjusted by CFL condition)
- Smagorinsky constant: $C_s = 0.1$
- Kinematic viscosity: $\nu = 10^{-6}\,\text{m}^2/\text{s}$
- River inflow: $Q_{\text{river}} = 0.1\,\text{m}^3/\text{s}$
- Tidal amplitude: $A_{\text{tidal}} = 1.0\,\text{m}$, period: $T_{\text{tidal}} = 43200\,\text{s}$
- Ocean salinity: $S_{\text{ocean}} = 35\,\text{PSU}$
- Ocean temperature: $T_{\text{ocean}} = 20\,°\text{C}$

Arrays include:

- Velocities: $u(i, j)$, $w(i, j)$
- Salinity: $S(i, j)$, temperature: $T(i, j)$
- Vorticity: $\omega(i, j)$
- Eddy viscosity: $\nu_e(i, j)$

The GUI allows parameter adjustments and selection of time integration schemes (RK4 or Crank-Nicolson).

Functioning Logic

The simulation follows:

1. **Density and Pressure Gradient**: Compute density using the equation of state:

$$\rho(i,j) = \text{EqOfState}(S(i,j), T(i,j), P), \quad P = \rho_0 g(h - j\Delta z)/10^4 \tag{9}$$

   Pressure gradient: $\partial P/\partial x = g(\rho_{i+1,j} - \rho_{i-1,j})/(2\Delta x)$.

2. **Eddy Viscosity**: Use Smagorinsky model:

$$\nu_e = (C_s\Delta)^2|\mathbf{S}|, \quad \Delta = \sqrt{\Delta x \Delta z}, \quad |\mathbf{S}| = \sqrt{2\left(\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial w}{\partial z}\right)^2\right) + \left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\right)^2} \tag{10}$$

3. **Tendencies**: Compute time derivatives for velocity, salinity, and temperature:

$$\frac{\partial u}{\partial t} = -\left(u\frac{\partial u}{\partial x} + w\frac{\partial u}{\partial z}\right) + \frac{1}{\rho_0}\frac{\partial}{\partial x}\left(\nu_e\frac{\partial u}{\partial x}\right) + \frac{1}{\rho_0}\frac{\partial}{\partial z}\left(\nu_e\frac{\partial u}{\partial z}\right) - \frac{1}{\rho_0}\frac{\partial P}{\partial x} \tag{11}$$

$$\frac{\partial w}{\partial t} = -\left(u\frac{\partial w}{\partial x} + w\frac{\partial w}{\partial z}\right) + \frac{1}{\rho_0}\frac{\partial}{\partial x}\left(\nu_e\frac{\partial w}{\partial x}\right) + \frac{1}{\rho_0}\frac{\partial}{\partial z}\left(\nu_e\frac{\partial w}{\partial z}\right) \tag{12}$$

$$\frac{\partial S}{\partial t} = -\left(u\frac{\partial S}{\partial x} + w\frac{\partial S}{\partial z}\right) + \frac{\partial}{\partial x}\left(\nu_e\frac{\partial S}{\partial x}\right) + \frac{\partial}{\partial z}\left(\nu_e\frac{\partial S}{\partial z}\right) \tag{13}$$

$$\frac{\partial T}{\partial t} = -\left(u\frac{\partial T}{\partial x} + w\frac{\partial T}{\partial z}\right) + \frac{\partial}{\partial x}\left(\nu_e\frac{\partial T}{\partial x}\right) + \frac{\partial}{\partial z}\left(\nu_e\frac{\partial T}{\partial z}\right) \tag{14}$$

4. **Time Integration**: Use RK4 or Crank-Nicolson (CN):

   - **RK4**: Four-stage explicit method:

$$\phi^{n+1} = \phi^n + \frac{\Delta t}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{15}$$

   - **CN**: Implicit for diffusion, explicit for advection, solved using tridiagonal matrix algorithm.

5. **Boundary Conditions**: River inflow ($u = Q_{\text{river}}/h, S = 0$) at $x = 0$, tidal velocity ($u = A_{\text{tidal}}\cos(2\pi t/T_{\text{tidal}})$) at $x = L$, no-slip at bottom, and free surface.

6. **Vorticity**: Compute as:

$$\omega = \frac{\partial w}{\partial x} - \frac{\partial u}{\partial z} \tag{16}$$

Visualization

The GUI displays:

- Vorticity field (blue-red gradient).
- Velocity vectors (black arrows).
- Status with simulation time, scheme, and Δt.

Comparison and Physical Models

Both classes model estuarine circulation with:

- **Navier-Stokes Equations**: Solved via LBM (D2Q9) in `LatticeBoltzmannLES` or finite volume in `LargeEddySim`.
- **Smagorinsky Model**: Eddy viscosity:

$$\nu_e = (C_s \Delta)^2 |\mathbf{S}| \tag{17}$$

- **Density**: $\rho = 1000(1 + 0.0008S - 0.00007(T - 20))$ in `LatticeBoltzmannLES`; custom `EqOfState` in `LargeEddySim`.
- **Boundary Conditions**: River inflow and tidal forcing drive the flow.

`LatticeBoltzmannLES` uses a lattice-based approach with adaptive grid refinement and energy spectrum output, while `LargeEddySim` uses a finite volume method with RK4 or Crank-Nicolson integration, suitable for structured grids.