TEXAS A&M UNIVERSITY Engineering

High Performance Research Computing
A Resource for Research and Discovery

TEXAS A&M UNIVERSITY

Filtered Density Function(FDF) Solver

Development Plan

Vishal Indivar Kandala

Project Review: FDF Solver Development

TEXAS A&M UNIVERSITY Engineering

Overall Objective: Develop **data-driven closure models** for turbulent scalar transport using the **Filtered Density Function (FDF) approach**.

•**Current Focus**: Build a **scalar FDF solver** as a foundational step toward full model development.
•**Goal**: Establish a flexible, validated FDF framework as the basis for future **data-driven closure model training & evaluation**.

•**Approach**: Leverage the existing **LES solver** to provide:

- Resolved flow fields (velocity, pressure)
- Sub-grid-scale (SGS) fields (e.g., Reynolds stresses)

•**Key Tasks Ahead**:

- Develop robust **Eulerian–Lagrangian coupling** for scalar particle transport.
- Implement the **scalar transport Eulerian solver** compatible with curvilinear grids.

Master Plan & Status: (1/5) Grid & Geometry Infrastructure

1.1 Setup Core DMDA Structures ✅

- ✅ Create DMDA user->da (DOF=1).
- ✅ Create/Derive DMDA user->fda (DOF=3).
- ✅ Set parallel decomposition layout.
- ✅ Setup DM structures (DMSetUp).

1.2 Grid Definition ✅

- ✅ Read grid params from options (control.dat).
- ✅ Read grid params from file (grid.dat).
- ✅ Logic to choose input source.
- ✅ Determine & store global dimensions (IM, JM, KM).

1.3 Assign Curvilinear Coordinates ✅

- ✅ Get local coordinate vector Coor.
- ✅ Generate coordinates (uniform/stretched).
- ✅ Read coordinates from file.
- ✅ Broadcast/Distribute coordinates.
- ✅ Populate local coordinate vector Coor.
- ✅ Synchronize global coordinate vector gCoor.

1.4 Calculate Grid Metrics ➡️

- ➡️ Define data structures (Vecs) for metrics.
- ➡️ Implement numerical differentiation of Coor.
- ➡️ Implement calculation of Jacobian, basis vectors, etc.
- ➡️ Store metrics in Vecs.
- ➡️ Update ghost values for metric Vecs.

1.5 Walking Search Algorithm (Curvilinear) ⚠️

- ⚠️ Compute All rank Bounding Boxes (curvilinear)
- ✅ Compute Distance of Particle to Face Analyze
- ✅ Compute Signed Distances from all faces of a cell to particle.
- ✅ Search through Cells in a Rank Bounding Box.
- ✅ Determine cell indices (i,j,k) & weights (a1,a2,a3).
- ✅ Store results in DMSwarm_CellID and weight fields (mechanism exists).

1.6: Verify/Adapt Particle Migration ⚠️

- ✅ Existing logic identifies potential migrants using physical coords vs. axis-aligned bounding box of *owned* nodes.
- ✅ Existing logic identifies target rank by checking axis-aligned bounding boxes (bboxlist) of immediate Cartesian neighbors (user->neighbors).
- ⚠️ Analyze robustness: Evaluate if the current axis-aligned bounding box approach (based on owned nodes) is sufficient for identifying *all* necessary migrants and finding the *correct* target rank near highly curved inter-processor boundaries.
- ➡️ Develop/Implement robust condition (if needed): Based on analysis, potentially implement checks against neighbor ghost cells or use more sophisticated geometric tests instead of simple axis-aligned boxes.
- ➡️ Verify: Test particle transfer across representative curved boundaries using the chosen (existing or revised) logic.

✅ **Complete/Functional** | ⚠️ **Partial/Needs Adaptation/Verification** | ➡️ **Requires Implementation**

Master Plan & Status: (2/5) Eulerian Field Management

2.1 Setup Context and Parameters ✅

- ✅ Define UserCtx struct.
- ✅ Read standard simulation parameters.
- ➡️ Add reading for FDF parameters (Sc_t, C_mix).

2.3 Handle LES Data Input ⚠️

- ✅ Implement PETSc binary reader (ReadFieldData).
- ✅ Read Ucat using ReadFieldData.
- ➡️ Read Nu_t into user->nu_sgs_les.
- ➡️ Calculate D_sgs_grid = nu_sgs_les / Sc_t.
- ⚠️ Update ghost cell values (Needs cases for new fields: D_sgs_grid, <φ>).

2.2 Setup Eulerian Field Vectors ⚠️

- ✅ Create Vecs for Ucat, P, Nvert, etc.
- ➡️ Create Vec user->nu_sgs_les (on da).
- ➡️ Create Vec user->D_sgs_grid (on da).
- ➡️ Create Vec user->euler_phi_mean (for <φ>, on da).
- ➡️ Create Vecs for FDF stats (phi_fdf_mean, phi_fdf_variance, on da).
- ➡️ Create Vec for FDF source term (if needed).

Master Plan & Status: (3/5) Lagrangian Particle Solver

TEXAS A&M UNIVERSITY Engineering

3.1 Setup DMSwarm ✅

- ✅ Create DMSwarm object.
- ✅ Register standard fields (position, velocity, pid, CellID, weight).

3.2 Add FDF Particle Fields ➡️

- ➡️ Register "phi_scalar" field.
- ➡️ Register "D_sgs_interp" field.
- ➡️ Register temporary fields if needed (e.g., "phi_fluct_sq").

3.3 Initialize Particle Properties ⚠️

- ✅ Initialize positions.
- ✅ Initialize velocity (to zero/interpolated).
- ➡️ Initialize "phi_scalar" based on initial condition.

3.4 Implement Stochastic Differential Equation (SDE) ➡️

- ➡️ Access required fields (pos, vel, D_sgs_interp).
- ➡️ Setup/Verify parallel PetscRandom.
- ➡️ Implement sqrt(2*D_sgs) term (handle non-positive D_sgs)**.
- ➡️ Implement Wiener increment dW generation**.
- ➡️ Implement Euler-Maruyama update step**.
- ➡️ Refactor UpdateAllParticlePositions.

3.5 Implement Scalar Mixing Model (IEM)** ➡️

- ➡️ Create new function UpdateParticleScalars_IEM.
- ➡️ Access particle fields & interpolated <φ>.
- ➡️ Calculate cell size Δ.
- ➡️ Calculate mixing timescale τ_mix.
- ➡️ Implement Euler forward update for phi_scalar.
- ➡️ Integrate call into AdvanceSimulation.

3.6 Implement Particle Boundary Handling ⚠️

- ✅ Check/Remove particles leaving physical domain.
- ➡️ Implement periodic boundary conditions (if needed).
- ➡️ Implement/Verify reflective boundary conditions for curvilinear walls.

Master Plan & Status: (4/5) Eulerian-Lagrangian Coupling

4.1 Grid-to-Particle Interpolation ⚠️

- ✅ Interpolate Ucat (nodal fda) to particle "velocity" field (Mechanism exists).
- ⚠️ Verify Ucat interpolation accuracy on curvilinear grid (PRIORITY 1c).
- ➡️ Implement D_sgs_grid (cell da) to particle "D_sgs_interp" interpolation.
- ➡️ Implement grid <φ> (from phi_cell_mean_grid on da) back to particles (for IEM).

4.2 Particle-to-Grid Statistics (Moments) ⚠️

- ✅ Calculate particle count per cell (user->ParticleCount).
- ✅ Implement scatter/accumulate framework.
- ✅ Implement normalization by count framework.
- ⚠️ Implement scatter/normalize for cell-mean <φ> (user->phi_cell_mean_grid, used for IEM).
- ➡️ Implement calculation & scatter/normalize for variance <φ'^{2}>_fdf (user->phi_fdf_variance).
- ⚠️ Implement scatter/normalize for final FDF mean <φ>_fdf (user->phi_fdf_mean).

Master Plan & Status: (5/5) Post Processing, Validation, Performance

TEXAS A&M UNIVERSITY
Engineering

5 Post Processing and I/O ⚠️

- ✅ Write Eulerian/Lagrangian fields to PETSc binary.
- ➡️ Write FDF statistics fields (<φ>_fdf, <φ'²>_fdf).
- ➡️ Write particle scalar field (phi_scalar).
- ⚠️ Adapt VTK output for new fields (FDF stats, particle scalar, maybe metrics).
- ➡️ Implement particle PDF calculation/export.
- ➡️ Implement ML-ready data output (HDF5/CSV).

7 Parallel Performance & Scalability ⚠️

- ✅ Particle migration framework exists.
- ⚠️ Verify particle migration on curvilinear grids (see Task 1.6).
- ✅ PETSc parallel structures used.
- ➡️ Verify parallel RNG correctness.
- ➡️ Implement strong/weak scaling tests.
- ➡️ Implement profiling of key components.

6 Validation & Test Cases ➡️

- ➡️ Curvilinear grid verification (VTK, metrics).
- ➡️ Curvilinear particle location validation.
- ➡️ Curvilinear interpolation/advection validation.
- ➡️ (Standalone Eulerian solver validation Deferred)
- ➡️ SDE particle dispersion test.
- ➡️ FDF test case(s) (e.g., scalar decay/channel flow using LES data).
- ➡️ Implement grid convergence tests.
- ➡️ Implement FDF diagnostics (PDF plots).

Priority 1: Ensuring Curvilinear Grid Compatibility

Key Focus Areas & Tasks (Phase 1)

1. Verify/Adapt Particle Location Algorithm (Task 1.5):

- Ensure robust mapping: physical (x,y,z) → logical (i,j,k, a1,a2,a3).
- Debug/Refine LocateParticleInGrid using coordinate/metric data.

Status: Framework exists [✅], Curvilinear logic needs verification/implementation [⚠️ ➡️]

2. Verify/Adapt Particle Migration Logic (Task 1.6):

- Ensure correct particle identification & transfer across curved processor boundaries.
- Evaluate robustness of current axis-aligned bounding box approach.

Status: Framework exists [✅], Curvilinear robustness needs analysis/verification [⚠️ ➡️].

3. Verify Interpolation Accuracy (Task 4.1.1 / 6.1.3):

- Assess accuracy of InterpolateEulerFieldToSwarm (for U_LES) on distorted grid cells.

Status: Framework exists [✅], Accuracy on specific grid needs verification [⚠️ ➡️].

Priority 2: FDF Core Physics Implementation

TEXAS A&M UNIVERSITY Engineering

1. Ingest SGS Data & Calculate Diffusivity (Task 2.3):

- Read LES Nu_t (cell-centered) into user->nu_sgs_les.
- Calculate D_sgs = Nu_t / Sc_t on the grid (user->da).

2. Add Particle Scalar & Interpolate D_sgs (Tasks 3.2, 3.3, 4.1):

- Add phi_scalar & D_sgs_interp fields to DMSwarm.
- Implement cell-to-particle interpolation for D_sgs.

3. Implement Stochastic Position Update (SDE) (Task 3.4):

- Modify UpdateAllParticlePositions: dX = U_LES dt + sqrt(2*D_sgs)dW.
- Requires parallel Random Number Generation.

4. Implement Mixing Model (IEM) (Task 3.5)**:

- Calculate conditional mean <φ|X_p> (scatter/interpolate).
- Calculate mixing timescale τ_mix.
- Update phi_scalar.

5. Calculate FDF Statistics (Task 4.2):

- Compute Mean <φ>_fdf (scatter/normalize phi_scalar).
- Compute Variance <φ'²>_fdf (scatter/normalize fluctuations) (If time permits).

** If time permits

Proposed 2-Month Plan & Mid-August Goals

TEXAS A&M UNIVERSITY Engineering

Phase 1: Curvilinear Foundation Verification (~3 Weeks)

Focus: Tasks 1.5, 1.6, 4.1.1 (Location, Migration, Interp. Accuracy).
Deliverable: Verified particle tracker components on curvilinear grid.

Phase 2: Core A Posteriori FDF Implementation (~5 Weeks)

Focus: Tasks 2.3 (Nu_t, D_sgs), 3.2/3.3 (phi_scalar), 4.1.2 (D_sgs interp.), 3.4 (SDE). Stretch: 3.5 (IEM), 4.2 (Stats).
Deliverable (Depends on Goal Level): Basic FDF prototype.

Focus: Prioritize robust curvilinear handling (Phase 1) above all else.

Mid-August (~2 Months) Goal Scenarios:

- **Ambitious Goal:** Phase 1 verified + FDF prototype with SDE, IEM, Mean & Variance calculation functional for a simple test case.
- **Moderate Goal (Primary Target):** Phase 1 verified + FDF core with SDE implemented & tested. phi_scalar added. D_sgs read & interpolated. (Maybe) FDF Mean calculated.
- **Conservative Goal (If Challenges Arise):** Phase 1 fully verified & robust (Curvilinear tracker foundation complete). Initial steps of Phase 2 started (e.g., Nu_t reading).

Dependencies : Phase 2 heavily relies on successful completion and verification of Phase 1. Plan flexibility is key.