Middle East Technical University
Department of Mechanical Engineering
ME 485-CFD using FVM
Fall 2024
Homework 2

Your task is to integrate diffusion computation methods into the *mefvm* code. You are going to solve a parabolic equation given as

$$\frac{\partial q}{\partial t} = \nabla \left(\mu \nabla q \right)$$

for constant coefficient μ and using the second-order cell-centered finite volume method.

Steps to Complete the Homework

1. **Retrieve Code:**
   - Download the initial code version from the GitHub repository: GitHub Link.
   - Ensure you work within the branch **HW2**.
2. **Complete Missing Implementations:**
   - Navigate to the *solvers/parabolic* folder.
   - Implement the following in the *ParabolicElements* class (located in *elements.py*):
     - *_make_compute_norm*: Compute the L_2 norm of error if it is known or the residual of the solution field if not.
     - *_make_compute_fpts*: Assign cell center values to element faces.
     - *_make_grad*: Compute the gradient of solution field. Note that the required operator is known.
   - Implement the following in the *ParabolicIntInters* and *ParabolicBCInters* classes (located in *inters.py*):
     - *_make_flux*: Compute the diffusive flux with the flipped sign (note that the - sign below) i.e.

       $$-\int_S \mu \nabla q \cdot \mathbf{n} dS$$

       using *minimum* correction, *orthogonal* correction and *over_relaxed* correction approaches. Note that these options can be given in the INI file.

- *_make_grad_at_face*: Compute the gradient at the face center using inverse distance weighted average of cell center gradients i.e.

$$(\nabla q)_{\text{face}} = w_l \, (\nabla q)_{\text{left element}} + w_r \, (\nabla q)_{\text{right element}}$$

3. **Run and Test the Code:**
   - Check the provided *parabolic.ini* file (in the *unittests* folder) and the sample scripts *parabolic_test_mesh.py* and *kershaw_mesh.py* to generate sample test meshes.
   - Execute the parabolic solver using the driver script *parabolic_test.py*.
   - Perform tests across:
     - Mesh types: Triangles, quadrilaterals, structured, unstructured.
     - Different Kershaw meshes with various distortion levels i.e. just modify α term in *kershaw_mesh.py* where $0 < \alpha < 0.5$.
     - Boundary conditions: Different configurations.

4. **Error Computation and Reporting:**
   - Compute the L_2 error norm between exact and approximate solutions. You can provide any exact steady or unsteady solutions to measure the errors.
   - Determine the performance of different correction approaches on various skewed mesh structures.
   - Document your implementation process, results, and findings in a detailed report.
   - Ensure the report includes:
     - Implementation details.
     - Observations from tests on various meshes and fields.

5. **Submit Your Work:**
   - Provide the **complete code** (all folders) to allow replication of results.
   - Submit a **PDF report** (other formats are not accepted).

Good luck!