UserGuide for UnDiFi-2D-V2

This document serves as a detailed, step-by-step guide to the newly developed Python scripts, specifically designed for integration with the NEO solver.

1. **Geometry creation and mesh generation**

   The software utilizes a triangular mesh. To generate this mesh, an SU2 mesh file with properly defined boundary conditions is required. This SU2 mesh can then be converted to the required format using the Python script `SU2_triangle.py`.

2. **Boundary Definition**

   Boundary conditions are specified in the `type.dat` file using predefined boundary markers. It is important to ensure that each boundary is correctly defined according to the intended flow configuration. For detailed guidance on boundary definitions, please consult the associated research paper.

3. **Run the simulation using the Shock-Capturing method**

   The second step is to run the simulation using the shock-capturing method. Please refer to the original user guide for detailed instructions on executing the simulation in shock-capturing mode.

4. **shock extraction and .node file updation**

   After the solution has converged, a file named `vvvv.dat` will be generated in the `NEO_data/output` directory. Use the Python script `UpdateNodeDetectShock.py` to extract the shock points and update the corresponding `.node` file with the updated solution. Replace the initial `.node` file with the updated version to proceed with the shock-fitting simulation.

5. **Generating Shock Equation and Refined shock points**

   After extracting the shock points, a shock curve is generated using curve-fitting methods. This fitted curve is then used to compute the refined shock points. For hypersonic flow over a cylinder, use the script `Hypersonic_Cylinder.py`; for transonic airfoil cases, use `Airfoil.py`. In cases where the shock is approximately linear, such as oblique shocks, a straight-line equation can be used to represent the shock curve and compute the corresponding refined shock points.

6. **Shock data generation**

   After generating the refined shock points, the next step is to compute the shock data, which includes the upstream and downstream flow properties across the shock front. This is accomplished using the Python script `shock_data.py`, which generates an Excel file containing the shock data. Use the values from this file to manually create the shock data file `sh00.dat` following the required format, and ensure that the definitions of special points are included appropriately. For details on `sh00.dat` format, please refer to the associated research paper.

7. **Input Parameters**

   The `input.dat` file contains the values of essential parameters required to run the shock-fitting code. To compute `DXCELL`, a critical parameter for the remeshing process, use the Python script `dxcell.py`. For details on other parameters, please refer to the associated research paper.

8. **Run the simulation using the Shock-Fitting method**

   Once all the necessary parameters are configured and the required files are placed in the working directory, the simulation can be run using the shock-fitting method. For detailed instructions on executing the simulation in shock-fitting mode, please refer to the original user guide.

9. **Surface data extraction**

   For cases such as an airfoil or a cylinder, where surface data is required, it can be extracted from the `vvvv.dat` file, located in the `step` folder, using the Python script `Surface_data.py`.

Note:
To aid understanding, users are encouraged to run the `Circular_cylinder_M81` test case available in the `tests` folder. All required files for each task are provided in their respective folders within the directory `Circular_cylinder_M8_180_20mm/Fitting`, along with a `readme.txt` file that offers reference and detailed instructions.