

# Different discretization schemes in OpenFOAM

Divyesh Variya

**Abstract**—In this study, the correlation of results due to different discretization in divSchemes in fvSchemes are explained. Selection of discretization in our solver is as important as meshing and deltaT. In this paper, results of few different discretization in divergence schemes are compared in the scalartransportFoam solver. Changes to be done in divSchemes inside fvSchemes with Cubic, upwind, limitedCubic, limitedLimitedLinear, limitedLinear, vanLeer, limitedVanLeer, linear, linearUpwind, QUICK, SFCD, UMIST. Oscillation effects and result accuracy are described.

**Keywords**—divSchemes, fvSchemes, scalartransportFoam discretization.

## I. INTRODUCTION

Over the past two decades there have been many research activities in the design and application of high order accurate numerical methods in computational fluid dynamics (CFD). High order methods are especially desirable for simulating flows with complicated solution structures. In this paper we give a review on the development and application of several classes of high order schemes in CFD, mainly concentrating on the simulation of basic flow. An important feature of the basic flow is the existence of shocks, interfaces and other discontinuities and often also complicated structure in the smooth part of the solution. This gives a unique challenge to the design of high order schemes to be non-oscillatory and yet still maintaining their high order accuracy. We will attempt to describe their main properties and their relative strength and weakness. We will also briefly review their developments and applications, concentrating mainly on the results over the past years.

## II. GEOMETRY

To understand and compare all discretization schemes available in OpnFOAMv6 a base case is prepared. For comparison, changes to be done in only one thing is divSchemes under fvSchemes. As shown in below image a 1D CAD geometry is done with blockMeshDict. The dimension of 1D CAD model is 10m long in X-direction 2m height in Y-direction and 1m deep in Z-direction.

```
convertToMeters 1;
```

```
vertices
(
  (-5 -1 -1)
  (5 -1 -1)
  (5 1 -1)
  (-5 1 -1)
  (-5 -1 0)
  (5 -1 0)
  (5 1 0)
  (-5 1 0)
);
```

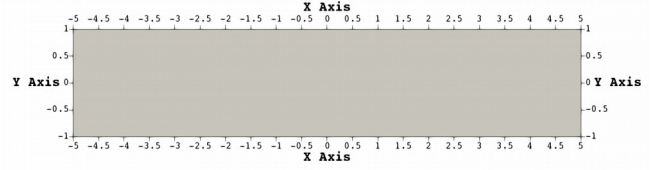


Figure 1. Base case geometry

## III. MESH

The mesh is executed with blockMeshDict, which is utility of OpenFoam itself. A structured mesh is created with code given below. It consists only hexahedras in the mesh, no other type of mesh geometry is used.

```
blocks
(
  hex (0 1 2 3 4 5 6 7) (1000 1 1) simpleGrading (1 1 1)
);
```

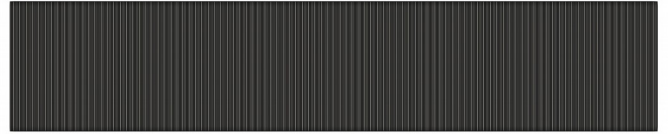


Figure 2. 1D Mesh in blockMeshDict

As per code shown above for the blockMeshDict, the 1D geometry decided with the meshing parameter. 1000 cells in X-direction shows that the geometry is 1 Dimensional. In Y and Z direction it is divided in 1 cells confirms the 1D geometry. Grading is simple type of grading with constant volume of each cells shows no upgrading or degrading with length..

Meshing is type of structural and quite smooth with respect to its length. So, it is cleared that super fine mesh is used to calculate accurate results in simulation. The result of checkMesh is given below.

Parameter	Value
Hexahedrons	1000
Max aspect ratio	200
Max skewness	8.88178e-15
Bounding box	(-5 -1 -1) (5 1 0)

TABLE 1. CHECKMESH RESULTS

## IV. LEVEL SET

After the meshing it is required to start setting up the simulations. In this case a scalar of temperature other then normal is going to translate in the given geometry with

specific velocity. No other conditions are used to setup the simulation other than velocity and temperature.

Initially temperature scalar is setup of 1 m long at the center of geometry. The temperature of 0 set to all the geometry and 1 to the scalar. Changes to the velocity is none in the case. The code of setFieldsDict is given below.

---

```
defaultFieldValues
(
    volVectorFieldValue U (1 0 0)
    volScalarFieldValue T 0
);

regions
(
    boxToCell
    {
        box (-0.5 -1 -1) (0.5 1 1);
        fieldValues
        (
            volScalarFieldValue T 1
        );
    }
);
```

---

## V. SIMULATION

For simulation run OpenFoamv6 is used and post-processing is done in paraview.

### A. Boundary conditions

As discuss above, temperature and velocity boundary conditions used in the case. Temperature sets the change in scalar and velocity gives translation to the scalar, no other field is required to define as per case condition.

The list of abbreviation used in the following table are:

1. ZG = ZeroGradient
2. IO = InletOutlet

Boundary	U	T
Up	Z	IO
Down	ZG	IO
Left	ZG	IO
Right	ZG	IO
FrontAndBack	Empty	Empty

Table 2. Boundary condition

### B. Solver detail

The scalarTransportFoam is a basic solver which resolves a transport equation for a passive scalar, using a user-specified stationary velocity field.

The scalarTransportFoam solver implements and solves a convection-diffusion scalar transport equation without source terms.

The scalarTransportFoam solver uses a complete convection-diffusion equation, in the in-compressible form.

$$\frac{\partial \rho \phi}{\partial t} + \nabla \cdot (\rho \mathbf{U} \phi) - \nabla \cdot (D \nabla \phi) = S_\phi \quad (1)$$

- The system will be solved till time derivation is zero (steady state condition)
- Diffusion coefficient  $D = 0$
- In-compressible fluid and therefore no temperature-dependent density (constant)
- No source terms

Hence, Equation can be written as,

$$\nabla \cdot (\rho \mathbf{U} \phi) = 0 \quad (2)$$

The numerical methodology used to solve the scalar transport equation in scalarTransportFoam is based on the finite volume technique. In particular, all the terms of the equations are discretion implicitly, using the FVM family of operators.

### C. Controlling simulation

Simulation is been run with 0 start time and 0.005 deltaT. Complete time to run simulation is 5 seconds. At the end time scalar is reached to the end of volume of geometry.

---

```
application    scalarTransportFoam;
startFrom      startTime;
startTime      0;
stopAt         endTime;
endTime        5;
deltaT         0.005;
writeControl   runTime;
writeInterval   1;
purgeWrite     0;
writeFormat    ascii;
writePrecision  6;
writeCompression off;
timeFormat     general;
timePrecision  6;
runTimeModifiable true;
```

---

As discuss in above sections, the change are done in divSchemes in fvSchemes which is located in system directory. Code of the specific section is given below.

---

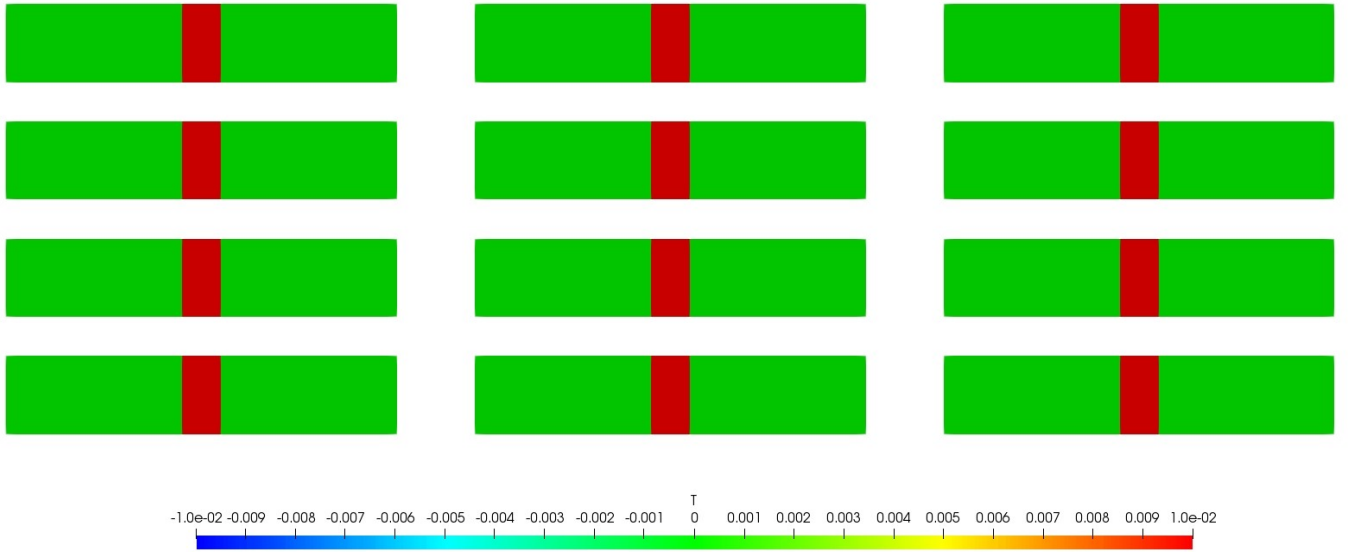
```
divSchemes
{
    default      none;
    div(phi,T)   scheme;
}
```

---

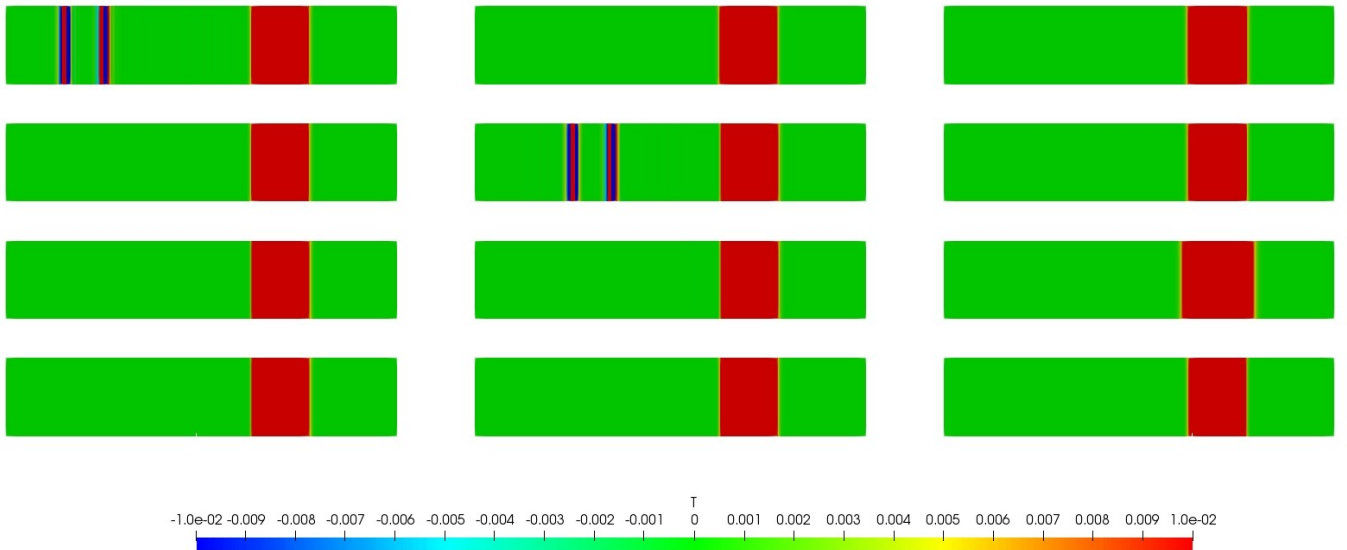
Here, scheme is replaced by the different schemes. Like linear, cubic, Quick, vanLeer, upwind and many more...

#### IV. RESULTS

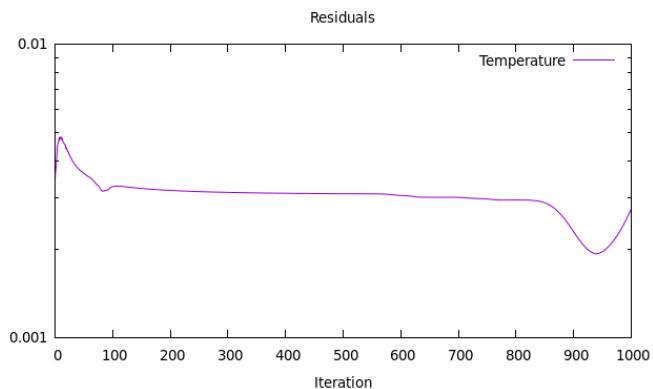
In the figure 1, time zero values shows different discretization results. At the beginning of the simulation, all results look same, because data set by setFieldsDict are not calculated yet. It will start calculating by time 0.005 as describe in controlling of the simulation section. For more accurate results in oscillation, data range is set by -0.01 to +0.01. The results shows in picture are name by cubic, limitedVanLeer, SFCD, limitedcubic, linear, UMIST, limitedlimitedlinear, limitedUpwind, Upwind, limitedLinear, QUICK, vanLeer respectively from upper left corner to right side.



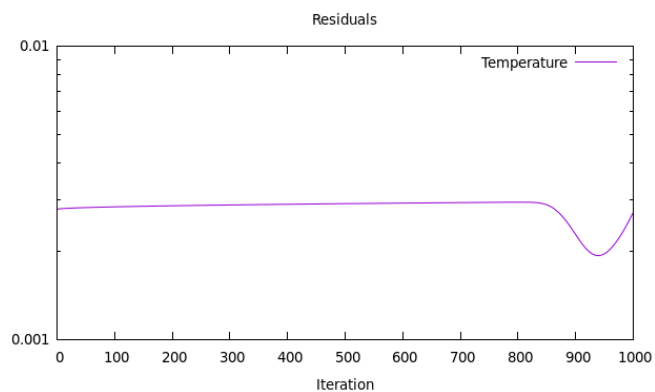
At time 2 seconds results shown below. When the simulation starts, it shows different results in all the schemes values. This happens because of different discretization method. Every discretization has its own equation to solve the main equation value. Taylor series with first order, second order, third order, and forth order discretization order define all the schemes. Some values gives oscillation to the results, some gives faded results and some gives very conjugated results. While simulating the problem in CFD its very crucial part to choose discretization scheme for accurate results. Meshing, deltaT and discretization scheme are all the important to choose for accurate and precise results.



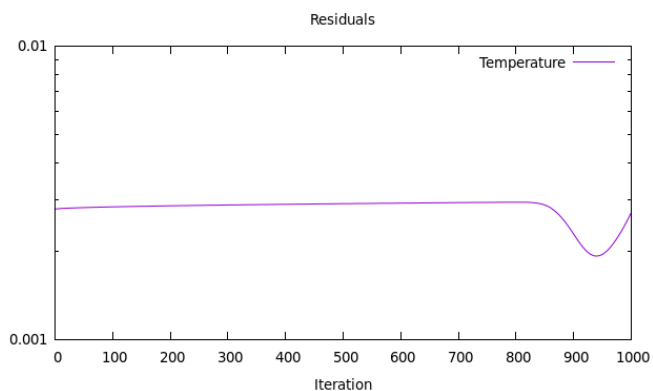
To visualize result data for all discretization scheme, graph are plotted with the help of gnuplot. Those graphs are listed below. Residual data of temperature result calculation is shown in all the graphs with different discretization.



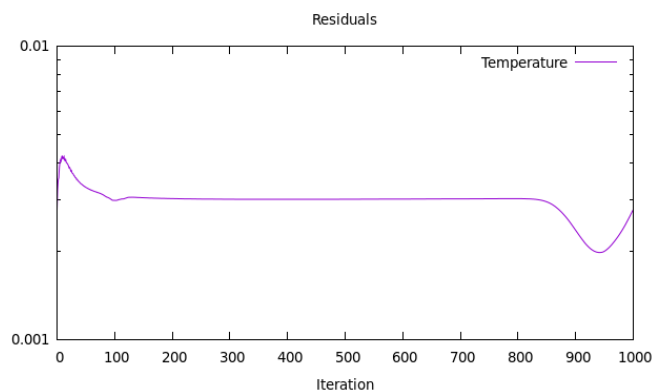
**Graph 1: Cubic**



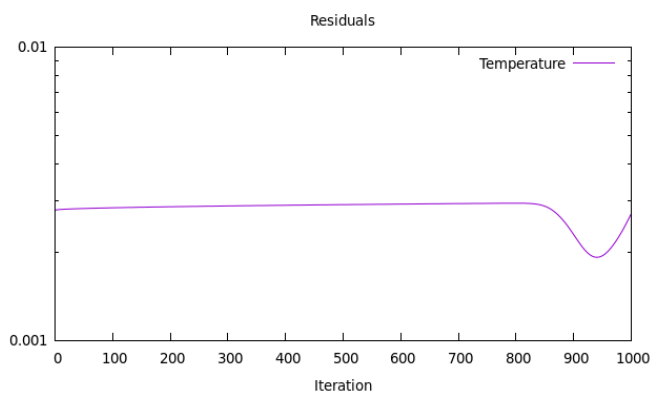
**Graph 5: LimitedVanLeer**



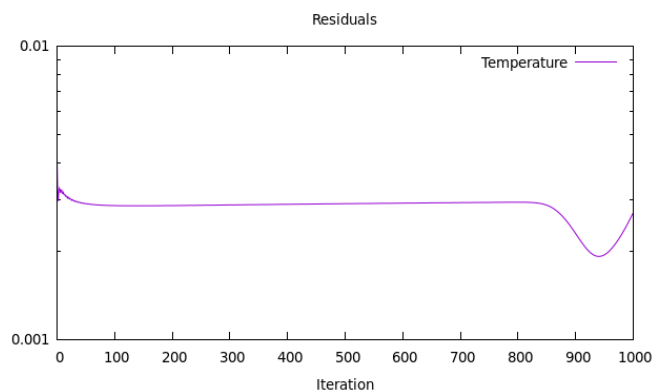
**Graph 2: Limitedcubic**



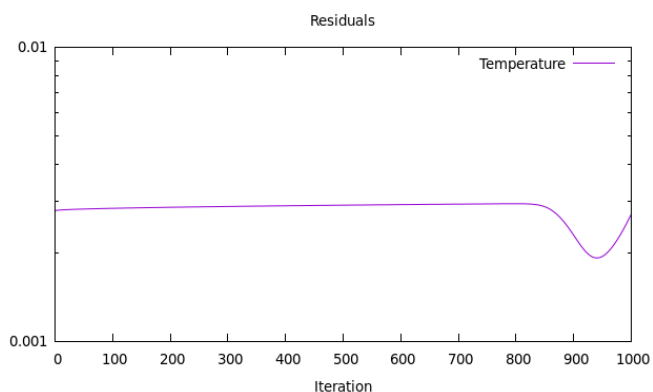
**Graph 6: Linear**



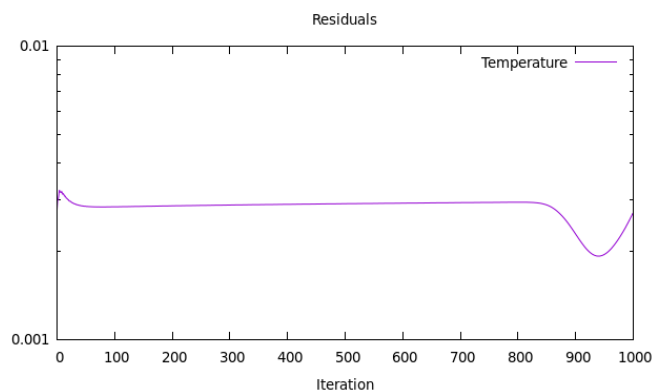
**Graph 3: LimitedlimitedLinear**



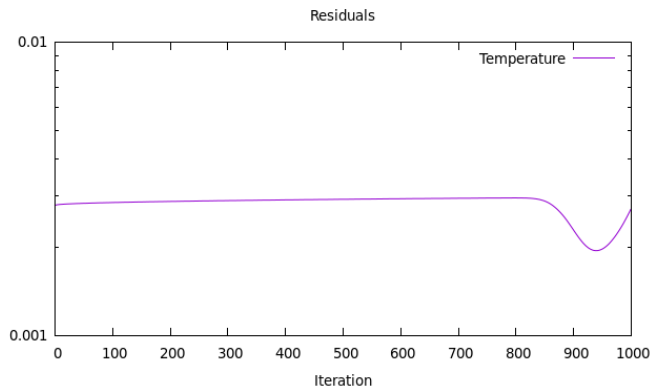
**Graph 7: linearUpwind**



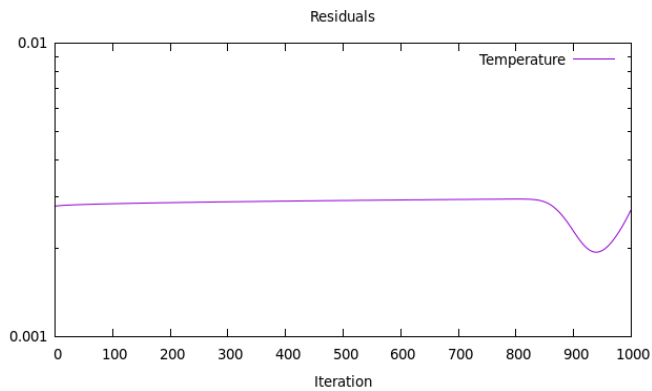
**Graph 4: LimitedLinear**



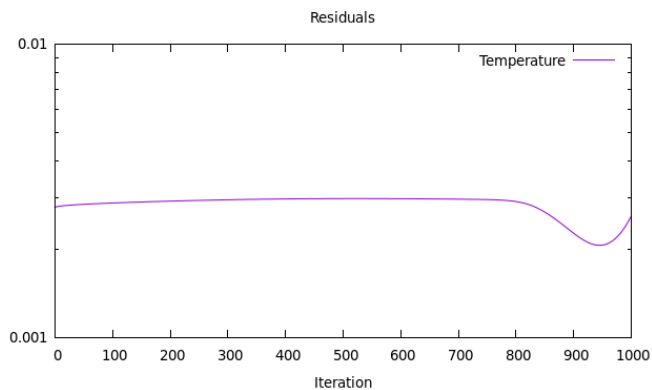
**Graph 8: QUICK**



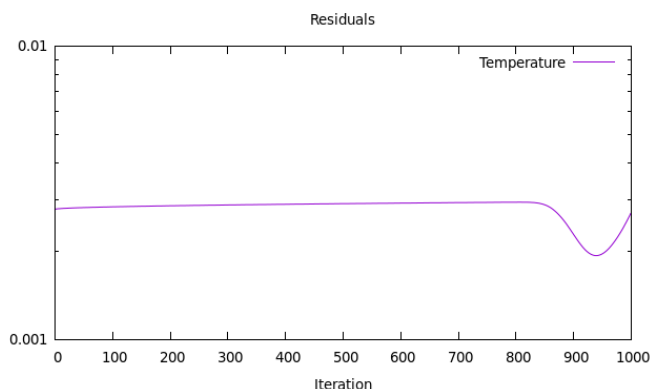
**Graph 9: SFCD**



**Graph 10: UMIST**



**Graph 11: Upwind**



**Graph 12: VanLeer**

## ACKNOWLEDGMENT

It is always a pleasure to remind the fine people in the Indian Institute of Technology (IIT), Bombay and Government Engineering college, Valsad for their the sincere guidance I received to uphold my theoretical as well as CAE skills in Computational fluid dynamics.

First of all, thanks to my parent for giving encouragement,enthusiasm and invaluable assistance to me. Without all this,I might not be able to complete this subject properly.

Second, I would like to thanks to Professor Shivasubramanian Gopalakrishnan (Department of Mechanical Engineering) forgive me the opportunity to do the marvelous project study. He also gives me their guidance and support.

Thirdly, I also want to express my deepest thanks to Mr Rohit Panday as an industry professional advisory for CAE Department that has helped me a lot in dealing with the industrial project. He had supported me by showing a different method of information collection about the CAE. He helped all the time when I needed and He gave the right direction toward completion of the project.

Besides, I would like to thank Mr Sathish kanniappan, Miss. Deepa Vedartham for extending their friendship towards me and making a pleasure-training environment in the IIT, Bombay during the internship. A paper is not enough for me to express the support and guidance I received from them almost for all the work I did there.

Finally, I apologize for all other unnamed who helped mein various ways to have a good training.



**Divyesh Variya** (M'97) received the B.E. degree in mechanical engineering from the gujarat technological univer sity, in 2018 and also work as an in tern under Prof. Shivasubramanian Gopalakrishnan for FOSSEE (Free and Opensource Software for Education) Project on OpenFOAM in Indian In stitute of Technology, Bombay. His re search interests include all CAE

projects with linear/non-linear structural analysis, compua- tional fluid dynamics, dynamic robotics analysis, failure anal- ysis and prevention, and design development.

**Contact:**

+91 7777908833

divyeshvariya7@gmail.com