



Lab Migration Report

On

**Modelling Passive Scalar As Reactive Chemical Species  
Without Diffusion In A Laminar Flow Reactor Following  
First Order Chemical Reaction**

Proposed By:

Prof. Dhiraj Garg

Solution By:

John Pinto

Manjil Sitoula

## Table of Contents

1. Governing Equations and Models .....	3
1.1 Problem Statements.....	3
1.2 Governing Equations.....	3
1.3 Geometry and Mesh. ....	4
1.4 Solver Setup .....	5
1.4.1 Fluid Properties .....	5
1.5 Case Setup.....	6
1.6 Initial and Boundary Conditions .....	9
1.6.1 Flow Field .....	9
1.6.2 Scalar.....	10
2. Results and Discussions.....	13

## List of Figures

Fig 1: Schematic Diagram of Computational Domain .....	4
Fig 2: Mesh Generation using BlockMesh .....	5
Fig 3 Tree diagram of the main folder .....	6
Fig 4 Detailed Tree diagram of the directories .....	6
Fig 5 Scalar Concentration along the tube.....	13
Fig 6 Scalar concentration at 0.375 m at 20 seconds.....	<b>Error! Bookmark not defined.</b>
Fig 7 Scalar concentration at 0.375 m after steady State .....	<b>Error! Bookmark not defined.</b>
Fig 8 Scalar concentration along a line at the cross section of Outlet after steady state .....	<b>Error! Bookmark not defined.</b>
Fig 9 Scalar Concentration along various sections of the tube.....	14
Fig 10 Scalar concentration vs time at the Outlet .....	<b>Error! Bookmark not defined.</b>

## List of Tables

Table 1-1 Boundary Conditions for U .....	9
Table 1-2 Boundary Conditions for p .....	9
Table 1-3 Boundary Conditions for scalar.....	10

# 1. Governing Equations and Models

## 1.1 Problem Statements

To model **passive scalar** as **reactive** chemical species **without** diffusion following first order **chemical reaction** and simulate the reaction in a laminar **flow reactor**.

Objective – The purpose is to understand how a passive scalar can be used as reactive chemical species **without** diffusion under **flow** conditions. To do this we need a model for chemical reactions involving desired chemical species which is taken to be first order here. The chemical reaction term is incorporated as **source term** in the passive scalar transport equation. The effect of flow process (convection only) and chemical reaction on concentration of chemical species in both upper and lower half along the flow is to be observed.

## 1.2 Governing Equations

The continuity and momentum conservation equations are solved for the calculation of flow parameters like velocity and pressure. The conservation equations are expressed below:

Continuity Equation:

$$\nabla \cdot \mathbf{u} = 0 \quad (1)$$

Momentum Equation:

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\frac{1}{\rho} (\nabla p) + \nu \nabla^2 \mathbf{u} \quad (2)$$

where  $\nu$  is the kinematic viscosity,  $\rho$  is the density,  $\mathbf{u}$  is the velocity vector and  $p$  is the pressure.

A concentration transport equation is incorporated for the calculation of passive scalar concentration using modified solver.

Equation to model the transport of Passive Scalars (S1 & S2):

$$\frac{\partial S}{\partial t} + \mathbf{u} \cdot \nabla S = \nabla (\Gamma \nabla S) + S_c \quad (3)$$

Where  $S$  (in our case S1 & S2) is the passive scalar,  $\Gamma$  is the diffusion coefficient and  $S_c$  is the source term of respective passive scalar. The source term would be required to model chemical reaction related to S1 and S2.

Let the chemical reaction be  $A \rightarrow B$  and assume the rate of reaction is of first order in a batch reactor, i.e.

$$-r_A = -\frac{dC_A}{dt} = kC_A = r_B = \frac{dC_B}{dt} \quad (4)$$

$C_{A0}$  is the initial concentration of A, while the initial concentration of B would be zero.

Let S1 represents the concentration of chemical species A and S2 represents concentration of chemical species B. As we can see, A is reactant and will be consumed in the reaction whereas B is product and thus will be produced during the reaction. So, the source term for S1 would be

$$S_{C1} = -r_A = -kC_A \quad (5a)$$

and for S2, it would be

$$S_{C2} = r_B = kC_A \quad (5b)$$

So the transport equation for  $C_A$  can be written as

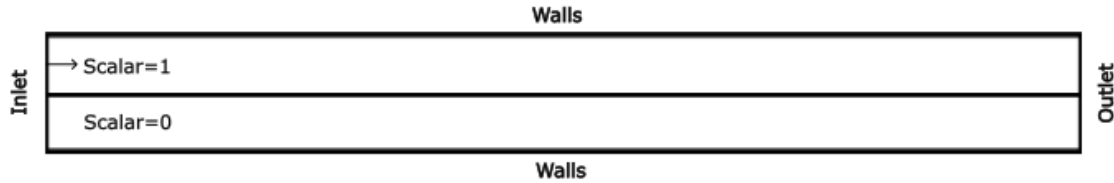
$$\frac{\partial C_A}{\partial t} + \mathbf{u} \cdot \nabla C_A = \nabla(\Gamma \nabla C_A) + (-r_A) = \nabla(\Gamma \nabla C_A) - kC_A \quad (6)$$

The ideal laminar flow reactor in context of straight cylindrical tube means to have parabolic radial velocity profile from inlet to outlet with **no** radial mixing. The only way a particle can move radially would be through diffusion only. Since the diffusion is zero in current case, theoretically, the chemical species should not diffuse to lower half of the cylinder while moving along the flow in its upper half. The purpose of this exercise is to check whether we are getting physically correct results from the simulation for no diffusion along with flow conditions. If any diffusion in the lower half of the cylinder is observed, it must be coming from numerical diffusion. So, solver settings, higher order discretization schemes etc. have to be modified to minimize or eliminate it. Else we won't be able to observe the correct effect of diffusion on the reaction and flow.

For more information regarding the source term, you can refer to this document [Theory](#).

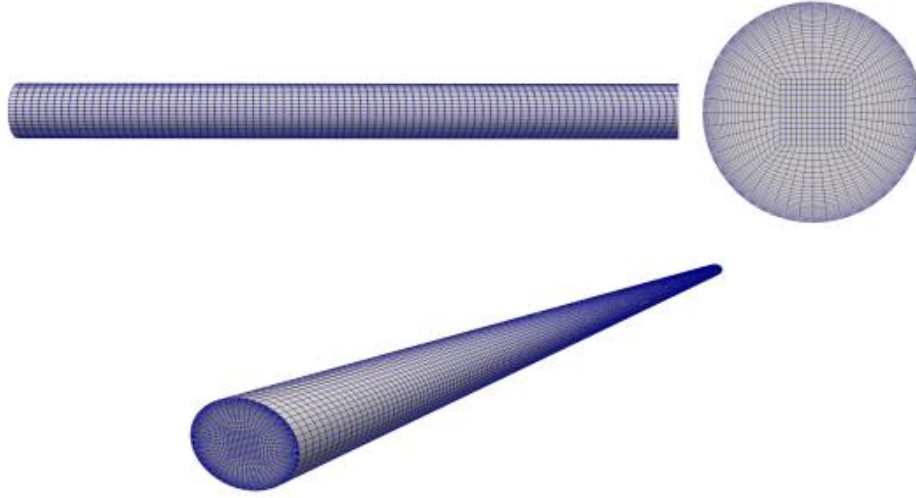
### 1.3 Geometry and Mesh.

The domain is a straight cylindrical tube with length of 0.5 m and diameter of the cross section as 0.01 m as shown in Fig. 1. The geometry is 3D. The geometry is long enough for the flow to fully develop. The chemical species will be injected at upper half of the inlet.



*Fig 1: Schematic Diagram of Computational Domain*

The meshing is done using blockMesh utility, openFOAM's built-in tool. The geometry is divided into 5 blocks and 18 vertices. The number of cells is 672000. The user is free to choose different types of meshing and numbers to get similar results. For the detailed meshing process, one can go through the openFOAM spoken tutorial number 6. [spoken tutorial](#)



*Fig 2: Mesh Generation using BlockMesh*

## 1.4 Solver Setup

### 1.4.1 Fluid Properties

Water at room temperature is used as the fluid and the thermo physical properties of water are taken for calculations. The kinematic viscosity of water at room temperature (300 K) is  $8.58 \times 10^{-7} \text{ m}^2/\text{s}$  and the average velocity of the flow is  $0.1 \text{ m/s}$ , which is half the maximum velocity. The Reynolds number is expressed as:

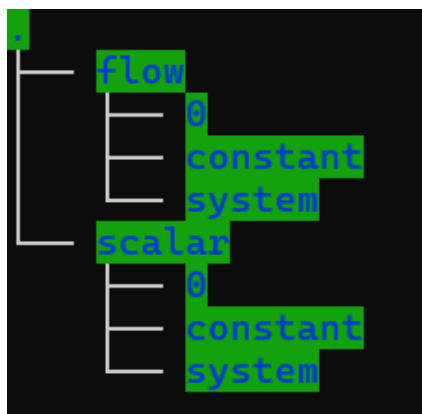
$$Re = \frac{U_{avg} * D}{\nu}$$

Where  $U_{avg}$  is the average velocity,  $D$  is the diameter of the tube and  $\nu$  is the kinematic viscosity. For the above input flow parameters, the Reynolds number of the flow is 1165 which is in the laminar regime ( $<2100$ ). So, a laminar model is used for the simulation.

Flow Parameters	Value
Max. Velocity ( $U_{max}$ )	0.2 m/s
Average Velocity ( $U_{avg}$ )	0.1 m/s
Density ( $\rho$ )	1000 kg/m <sup>3</sup>
kinematic viscosity ( $\nu$ )	$8.58 \times 10^{-7} \text{ m}^2/\text{s}$
Reynolds No.	1165
Scalar Diffusivity constant (DS1)	0 m <sup>2</sup> /s
Scalar Kinetic rate coefficient(kS1)	1 s <sup>-1</sup>

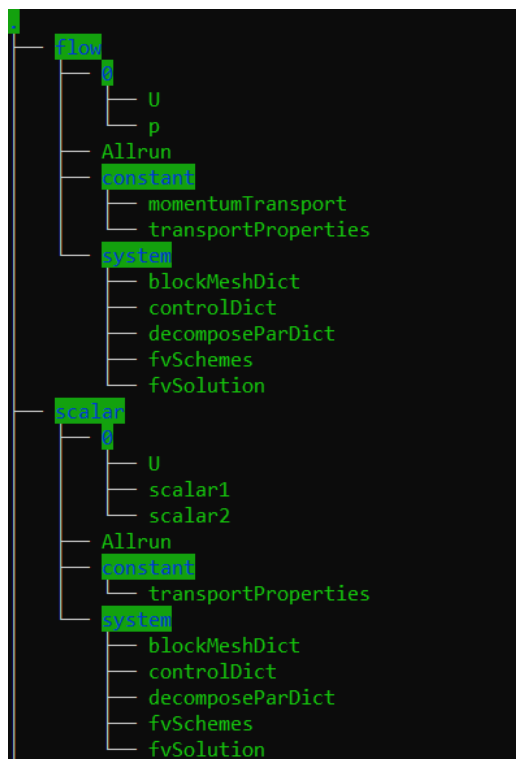
### 1.4.2 Case Setup

The case files for the current session are available in this [link](#). Download and extract these files into your run directory. A general overview of the setup is explained below:



*Fig 3 Tree diagram of the main folder*

The main folder consists of flow and scalar folder as in Fig 3. The flow folder consists of files to solve the flow field. The scalar folder has the necessary files to simulate the scalar. The velocity field solved using flow folder is used by the scalar to move along the tube and trace the path of the flow. Since, newScalarTransportFoam1S, does not solve for velocity, we need to simulate for the flow field and provide as a path for the scalar. The scalar folder is used in simulating the scalar change depicting the chemical reaction. The folders can be further expanded along this tree:



*Fig 4 Detailed Tree diagram of the directories*

The 0 directory of scalar folder consists of initial and boundary conditions for scalar1 and scalar2. Scalar 1 is used to model one of the chemical species which after reaction is converted to another species, modeled by scalar2.

### Flow case setup:

The initial and boundary conditions for velocity and pressure are provided in the  $U$  and  $p$  files of 0 directory. You can see the boundary conditions by accessing these files. The boundary conditions are further explained in the next section.

- The kinematic viscosity of fluid is provided in **constant/transportProperties**.

```
transportModel  Newtonian;
nu              [0 2 -1 0 0 0 0] 8.58E-07;
```

- Similarly, in momentumTransport dictionary, type of model for the simulation is provided. In our case, we have used the laminar model.

```
FoamFile
{
    format      ascii;
    class       dictionary;
    location    "constant";
    object      momentumTransport;
}

simulationType laminar;
```

- The blockMeshDict consists of mesh information and the controlDict dictionary consists of case controls like timing, write information etc.
- System/decomposeParDict dictionary is used for parallel computing.

The steps for the simulation are provided below:

1. First, you need to navigate to the flow folder in your run directory.

```
cd $FOAM_RUN
```

```
cd HalfBore_0D_New/flow
```

2. The Allrun file consists of necessary commands to run the simulation. Type **./Allrun** and press enter.

The Allrun file consists of following commands:

```
blockMesh
decomposePar
mpirun -np 6 simpleFoam -parallel
reconstructPar
```

The blockMesh command is used to generate the mesh. Command decomposePar decomposes the domain into subdomains and assigns the number of processors to these subdomains based on the method like simple, scotch etc. In this case, 6 processors are used in parallel and simpleFoam solver is used. At last, reconstructPar command is used to reconstruct a single domain from the processor sub-domains.

### **Scalar Case setup:**

#### **Solver modification and compilation**

To simulate the scalar, we have modified the scalarTransportFoam solver and named it newScalarTransportFoam1S. To make executables for this solver we will need to compile it first. To do that follow the steps given below:

1. Open your terminal and navigate to the run directory.  
**cd \$FOAM\_RUN**
2. Navigate to the solver folder by typing the following command.  
**cd newScalarTransportFoam1S**
3. Compile the solver by typing the following command and press enter.  
**wclean**  
**wmake**

After this the following steps are required.

1. First, you need to navigate to the flow folder.

**cd HalfBore\_0D\_Source\_New /scalar**

2. The Allrun file consist of necessary commands to run the simulation. Type **./Allrun** and press enter.

The Allrun file consists of following commands:

```
blockMesh
decomposePar
mpirun -np 6 newScalarTransportFoam1S -parallel
```

These commands are explained in the previous section.

### **Transport Properties:**



```

FoamFile
{
    format      ascii;
    class       dictionary;
    location    "constant";
    object      transportProperties;
}

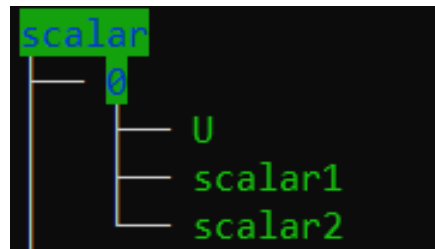
DS1            [0 2 -1 0 0 0 0] 0;
DS2            [0 2 -1 0 0 0 0] 0;
kS             [0 0 -1 0 0 0 0] 1;

```

Here, DS1 is the diffusivity and kS1 is the kinetic rate coefficient. For this case, there is no diffusion of both the scalars. Since, it models chemical reaction the kinetic rate coefficient is set to 1. This can be changed to different values.

### 1.4.3 Initial and Boundary Conditions

The initial and boundary conditions for the flow and scalar simulation are used separately. The scalar term represents the concentration of the passive scalar in the simulation. At first, the flow is simulated as steady state to obtain a velocity field through which scalar moves. The flow field conditions for the scalar simulation are provided by the flow simulation and a new solver is implemented to simulate the concentration of passive scalar. For the flow simulation, simple Foam, a steady state solver is used whereas newScalarTransportFoam1S, an incompressible transient solver is used for the scalar. The initial and boundary conditions are provided in the 0 directory which contains U, Scalar1 and Scalar2 dictionary.



#### 1.4.3.1 Flow Field

The initial conditions are set to zero and the necessary boundary conditions for the flow are tabulated below:

Table 1-1 Boundary Conditions for U

Patch	Condition
Inlet	codedFixedValue
Outlet	zeroGradient
Walls	noslip

Table 1-2 Boundary Conditions for p

Patch	Condition
Inlet	zeroGradient
Outlet	fixedValue(uniform 0)
Walls	zeroGradient

Since the velocity at the inlet is parabolic and the cylinder is 3D, velocity at various locations (x, y) is calculated using a code which can be accessed in U file of 0 folder.

```

inlet
{
    type          codedFixedValue;
    value          uniform (0 0 0);

    name parabolicVelocity;
    code
    #{
        const vectorField& Cf = patch().Cf();

        vectorField& field = *this;
        const scalar R = 0.005;
        const scalar c = 0;
        const scalar Umax = 0.2;

        forAll(Cf, faceI)
        {
            const scalar x = Cf[faceI][0];
            const scalar y = Cf[faceI][1];

            field[faceI] = vector(0,0,Umax*(1-((pow((y-c)/R,2))+((pow((x-c)/R,2))))));
        }
    #};
}

```

The velocity at the inlet is computed using the coded boundary condition.

A steady state simulation is done to calculate the steady velocity profile in the tube which is used as an input to the scalar which is carried out as transient simulation.

#### 1.4.3.2 Scalar (Scalar1 & Scalar2)

##### 1.4.3.2.1 Initial Conditions

The steady state velocity profile obtained from the flow simulation is used as a local flow field for the scalar to trace the path of flow. The U file of last time step from the flow simulation is kept in the 0 directory of the scalar. At first, the scalar2 is set to 0 and scalar1 is injected at the upper half of the inlet cross section with concentration of 1. With time scalar1 is slowly converted to scalar2 as in the chemical reaction involving change of one species to another.

##### 1.4.3.2.2 Boundary conditions

The flow field is used from the flow simulation and the boundary conditions for the scalar1 and scalar2 is tabulated below:

*Table 1-3 Boundary Conditions for scalar1*

Patch	Condition
Inlet	codedfixedValue
Outlet	zeroGradient
Walls	zeroGradient

### **codedFixedValue for Scalar1:**

```
inlet
{
    type          codedFixedValue;
    value         uniform 1;

    name halfBore;
    code
    #{
        const vectorField& Cf = patch().Cf();

        scalarField& field = *this;

        forAll(Cf, faceI)
        {
            const scalar r = Cf[faceI][1];
            if (r>=0)
            {
                field[faceI] = 1;
            }

            else
            {
                field[faceI] = 0;
            }
        }
    #};
}
```

Here, r takes the y coordinates of the inlet face and if r is greater than zero i.e. the upper half, it assigns the value of scalar field at inlet to 1, otherwise 0.

### **Boundary condition for Scalar2:**

The boundary condition at the inlet is set to zero. Also, initial condition for the scalar is zero all over domain. The scalar1 is converted to scalar2 after some time and its concentration increases with decrease in scalar1.

```

FoamFile
{
    format      ascii;
    class       volScalarField;
    object      scalar2;
}

dimensions     [0 0 0 0 0 0 0];

internalField   uniform 0;

boundaryField
{
    inlet
    {
        type      fixedValue;
        value      uniform 0;
    }

    outlet
    {
        type      zeroGradient;
    }

    wall
    {
        type      zeroGradient;
    }
}

```

#### 1.4.3.2.3 Source term

The scalar S1 is consumed during reaction which is modeled by newScalarTransportFoam1S solver through source term.

```

fvScalarMatrix scalar1Eqn
(
    fvm::ddt(scalar1)
  + fvm::div(phi, scalar1)
  - fvm::laplacian(DS1, scalar1)
  ==
    -kS*scalar1
);

fvScalarMatrix scalar2Eqn
(
    fvm::ddt(scalar2)
  + fvm::div(phi, scalar2)
  - fvm::laplacian(DS2, scalar2)
  ==
    kS*scalar1
);

```

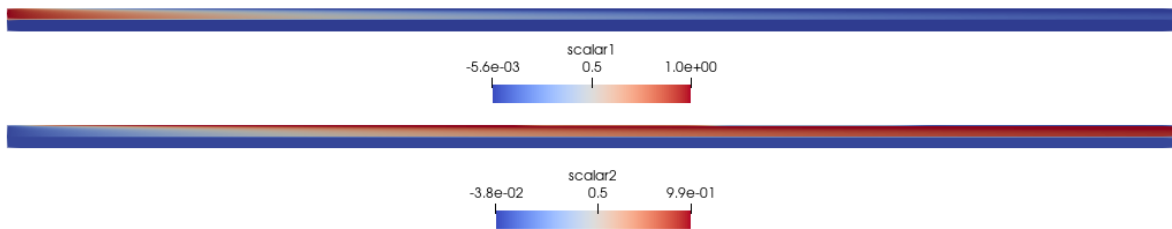
### 1.4.3.3 Steady State Study of Scalar

The scalar steady case is to see the steady state of the simulation. If one is interested in the final results or when the solution doesnot change with time and don't want to trace the scalar with time, once can navigate to this folder **cd HalfBore\_0D\_New /scalar\_steady** and type **./Allrun** and press enter. This only provides the steady state solution and observe the final state of the scalar.

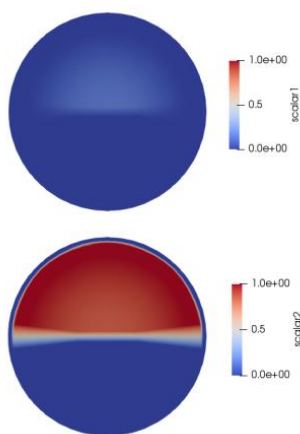
## 2. Results and Discussions

The scalar is injected from the upper cross section of the inlet with the concentration 1 and is gradually moves along the tube with time. Since there is no diffusivity of the scalar, the concentration of the scalar is only at the upper half and does not move to the lower region. The diffusion at the centre line is due to numerical diffusion alone or maybe due to poor post processing. Both possibilities should be checked to correct it.

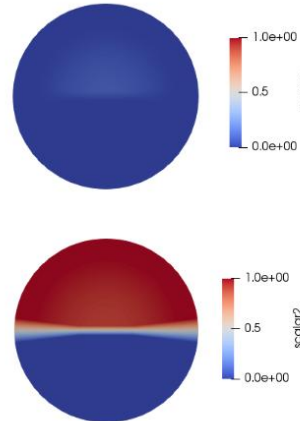
The scalar concentration along the tube is shown below at  $t = 20s$  in Fig 5. The scalar1 gets converted to scalar2 as the flow proceeds. At the initial sections of the tube near the inlet, scalar1 concentration is higher but as the time progresses and flow marches forward, it starts converting to scalar2. Since the velocity is parabolic, the velocity near the walls is less than at the center so it can be observed that the change of conversion is also higher near the walls.



*Fig 5 Scalar Concentration along the tube*



*Fig 6 Scalar concentration at 0.375 m at 20s.*



*Fig 7 Scalar concentration at 0.375 m after steady State*

The scalar concentration at the cross section of the tube at 0.375 m is shown in Fig 6. Since there is no diffusion, the flow does not cross the center to the lower half. The scalar2 at the walls being zero is due to the simulation time being insufficient for the flow to reach this section near the walls.

After steady state simulation, we can see that the scalar at the boundary layer reaches the concentration of 1 and can be seen in Fig 6 Scalar concentration at 0.375 m at 20s. .

After the flow reaches steady state, the area averaged concentration is calculated at various sections of the tube (0.125 m,0.25 m,0.375m) and cup average concentration is calculated at the outlet.

The area average concentration can be calculated as:

$$S_{avg} = \frac{\int S. dA}{A_{cross-section}}$$

The cup average concentration can be calculated as:

$$S_{avg} = \frac{\int u. S. dA}{U_{avg} * A_{outlet}}$$

Where u is the velocity and S is the scalar concentration at area dA.

The concentration of scalar1 decreases and scalar2 increases as the fluid flows along the tube which can be seen in Fig 8.

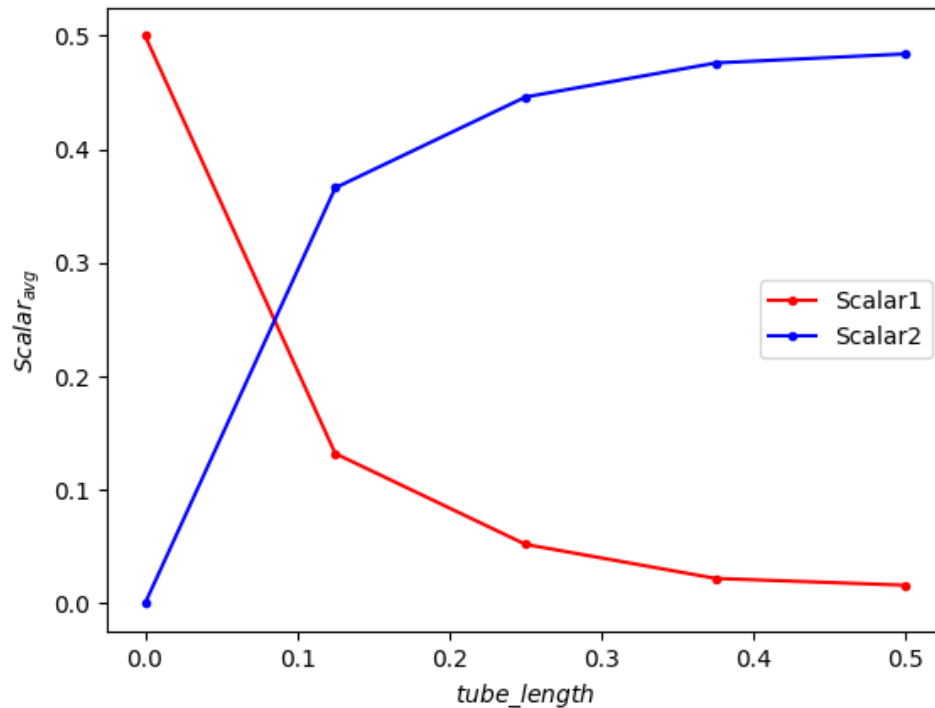


Fig 8 Scalar Concentration along various sections of the tube

The user can change the value of kinetic rate coefficient and observe the effect on the decay as to how fast and how slow it is happening. The user can also implement higher order reaction rate given by

$$-r_A = -\frac{dC_A}{dt} = kC_A^n \quad (7)$$

where  $n$  is the order of the reaction and can have any value from 0 onwards. The user can obtain an analytical solution corresponding to this and compare the results obtained by the simulation under plug flow or batch reactor condition. Similarly, the user can obtain the concentration profile for S2.

Another important aspect of CFD is that it is based on the law of conservation of mass and not moles. Moles are conserved only when there is no reaction. On the other hand, all rate equations are based on moles instead of mass as the units of concentration terms are in mol/vol and reaction rate is defined as mol/vol/time. The units of kinetic rate coefficient  $k$  can be obtained accordingly. So, the rate equation, thus kinetic rate coefficient and concentration terms have to be made dimensionless in terms of moles. This is achieved by following method:

Applying following transformation in eqn. (7),

$$C'_A = \frac{C_A}{C_{A0}} \quad (8)$$

This will make initial concentration as 1 at inlet which is the case here also.

$$\text{we have } -r'_A = -\frac{dC'_A}{dt} = k \cdot C_{A0}^{n-1} C_A'^n = k' C_A'^n \quad (9)$$

$$\text{Where } k' = k \cdot C_{A0}^{n-1} \quad (10)$$

This will work for all values of  $n$  making units of  $k$  as  $s^{-1}$ . This transformation will also achieve the purpose of normalization thus reducing the numerical errors also.