



Lab Migration Report

On

**Modelling Passive Scalar As Reactive Chemical Species**  
**Following First Order Chemical Reaction In An Ideal Batch**  
**Reactor**

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. ....	5
1.4 Solver Setup .....	6
1.4.1 Fluid Properties .....	6
1.5 Case Setup.....	6
1.6 Initial and Boundary Conditions .....	8
1.6.1 Flow Field .....	<b>Error! Bookmark not defined.</b>
1.6.2 Scalar.....	8
2. Results and Discussions.....	9

## List of Figures

Fig 1: Schematic Diagram of Computational Domain .....	5
Fig 2: Mesh Generation using BlockMesh .....	5
Fig 3 Detailed Tree diagram of the directories .....	6
Fig 4 Scalar concentration vs time.....	9
Fig 5 Scalar concentration at various cross section at time=5 seconds .....	10

## List of Tables

Table 1-1 Boundary Conditions for U .....	<b>Error! Bookmark not defined.</b>
Table 1-2 Boundary Conditions for p .....	<b>Error! Bookmark not defined.</b>
Table 1-3 Boundary Conditions for scalar.....	8

# 1. Governing Equations and Models

## 1.1 Problem Statements

To model **passive scalar** as **reactive** chemical species following first order **chemical reaction** and simulate the reaction in an **ideal batch reactor**.

Objective – The purpose is to understand how a passive scalar can be used as reactive chemical species (with or without diffusion) without flow condition. To do this we need a model for chemical reactions involving desired chemical species. The chemical reaction term is incorporated as **source term** in the passive scalar transport equation. It is this source term which differentiates whether the chemical species is reactive or non-reactive. Besides, numerical value of the source term is negative for reactants, which means consumption and positive for the products which means generation. Simulating the domain as batch reactor and comparing the results with known results is important to verify the correctness of the model of the reaction rate implemented in the code. How to simulate the domain with moving flow as batch reactor (no flow) is equally important, that the user will learn here.

## 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.

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

The analytical solution for this equation is

$$C_A = C_{A0} \exp(-kt) \quad (5a)$$

where  $C_{A0}$  is initial concentration of A. Similarly, for B, the solution would be

$$C_B = C_{A0}[1 - \exp(-kt)] \quad (5b)$$

where the initial concentration of B would be zero. The above solution can be obtained by integrating  $-r_A = -\frac{dC_A}{dt} = \frac{dC_B}{dt} = r_B \rightarrow -dC_A = dC_B$  from 0 to t.

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

$$S_{C1} = -r_A = -kC_A \quad (6a)$$

and for S2, it would be

$$S_{C2} = r_B = kC_A \quad (6b)$$

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

$$\frac{\partial C_A}{\partial t} + \mathbf{u} \cdot \nabla C_A = \nabla(\Gamma \nabla C_A) + (-r_A) = \nabla(\Gamma \nabla C_A) - kC_A \quad (7)$$

For the case when there is no flow and no diffusion, the above equation would be reduced to:

$$\frac{\partial C_A}{\partial t} = (-r_A) = -kC_A \quad (8)$$

**Which is nothing but the equation of batch reactor eqn. (4).**

Similarly, we will obtain from eqn. 3:

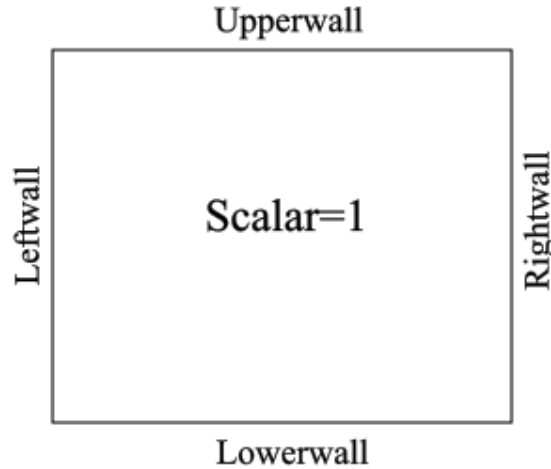
$$\frac{\partial S}{\partial t} = S_c = (\text{reaction equation}) \quad (9)$$

Eqn. (9) will be used for simulating batch reactor condition. The purpose of this exercise is to check whether we are getting correct results from the simulation for this case comparing the results against the analytical solution shown in eqn. (5). This process of comparing the results to check the correctness of the model is called verification. This does not require large number of meshes. Only one mesh is enough if allowed by the software. The same approach can be applied to the whole domain where reaction is to be simulated and verify that it is giving same and correct result everywhere thus behaving as fully mixed ideal batch reactor. Any deviation in space or time will indicate the potential issue either with model or simulation. This will enable the user to identify the problem and rectify it before fully implementing the reaction scheme along with flow.

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

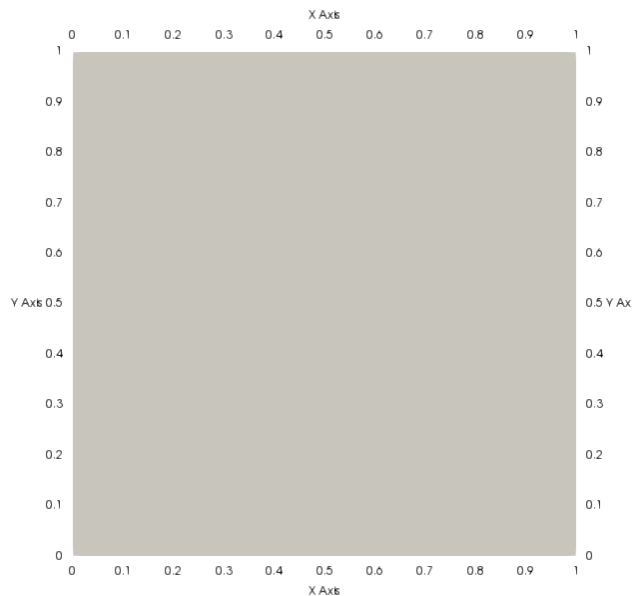
### 1.3 Geometry and Mesh.

The domain is a rectangular block with a single grid with 1m in x and y direction. The geometry is 3 dimensional and the simulation is carried out in 2D keeping the front and back patch as empty as shown in Fig 1. There will be no flow and no diffusion.



*Fig 1: Schematic Diagram of Computational Domain*

The meshing is done using blockMesh utility, openFOAM's built-in tool. The geometry is divided into 1 block and 8 vertices. The number of cell is 1. The geometry in itself is a grid. 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 and Scalar Properties

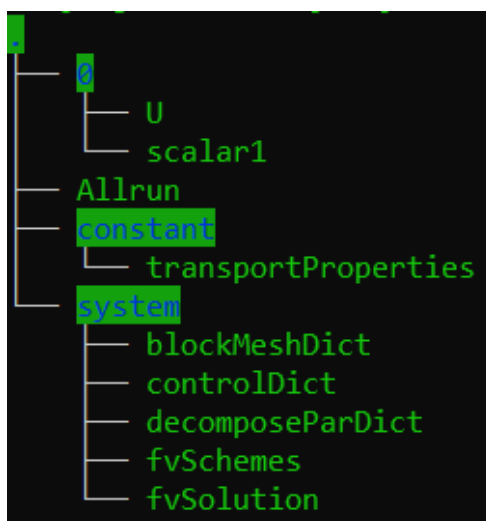
We model this without any flow field. So, there is no requirement for fluid properties. But we need to assign the properties for scalar to model its diffusivity and reaction rates.

Scalar Diffusivity coefficient (DS1)	$0 \text{ m}^2/\text{s}$
Kinetic rate coefficient (kS1)	$1 \text{ s}^{-1}$

### 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:

Since we do not need velocity field, U file is kept just for the case to run and is not used during simulation.



*Fig 3 Detailed Tree diagram of the directories*

The 0 directory of scalar folder consists of initial and boundary conditions for scalar1. Scalar1 is used to model the chemical species which after reaction is converted and decreases with time.

- 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 batchReactor\_New**

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 4 newScalarTransportFoam -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 newScalarTransportFoam. 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 newScalarTransportFoam**
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 batchReactor\_New /scalar1**

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 4 newScalarTransportFoam -parallel
reconstructPar
```

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;
kS1            [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 scalar and since, it models chemical reaction the kinetic rate coefficient is set to 1. This value could be changed to any value by the user to observe the effect on the results.

### 1.4.3 Initial and Boundary Conditions

The initial and boundary conditions for both the scalars are required.

#### 1.4.3.1 Scalar1 and Sacalar2

##### 1.4.3.1.1 Initial Conditions

The scalar1 is initialized with internal field value of 1. The scalar2 is initialized with internal field value of 0.

##### 1.4.3.1.2 Boundary conditions

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

*Table 1-1 Boundary Conditions for scalar1*

Patch	Condition
leftWall	zeroGradient
rightWall	zeroGradient
upperWall	zeroGradient
lowerWall	zeroGradient
frontAndBack	empty

##### 1.4.3.1.3 Source term

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

```

while (simple.correctNonOrthogonal())
{
    fvScalarMatrix scalar1Eqn
    (
        fvm::ddt(scalar1)
        + fvm::div(phi, scalar1)
        - fvm::laplacian(DS1, scalar1)
        ==
        -kS1*scalar1 ← Source
    );

    scalar1Eqn.relax();
    fvConstraints.constrain(scalar1Eqn);
    scalar1Eqn.solve();
    fvConstraints.constrain(scalar1);
}

```

## 2. Results and Discussions

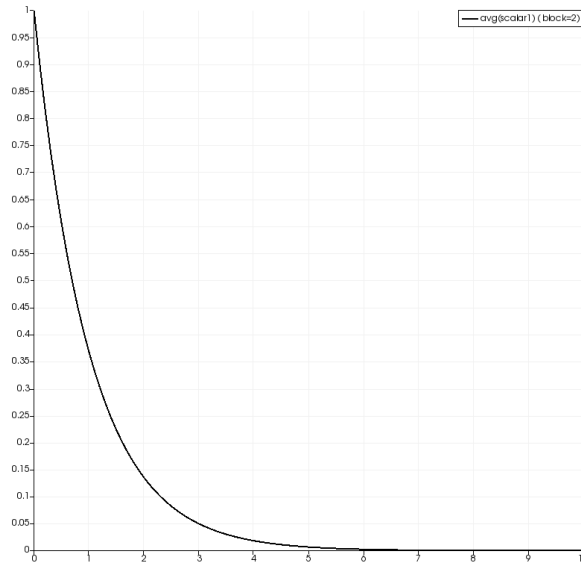
Since there is no convection and diffusion of the scalar, the scalar transport equation can be simplified to obtain

$$\frac{\partial S}{\partial t} = S_c$$

Where  $S_{C1} = -r_A = -kC_A = -kS1.S1$ , and  $S_{C2} = r_B = kC_A = kS1.S1$  where  $kS1$ ,  $S1$  and  $S2$  are the kinetic rate coefficient, scalar1 and scalar 2 concentration respectively.

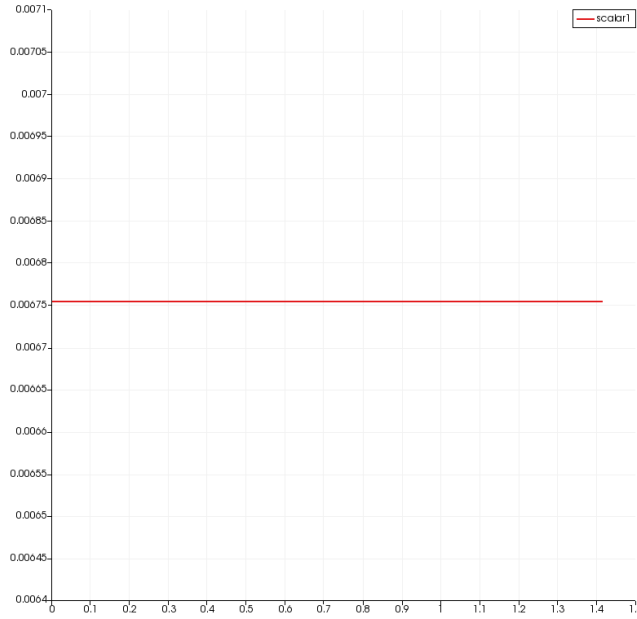
This differential equation provides the solution of scalar  $S1$  as  $S1 = 1. e^{-kS1.t}$ .

The transient simulation is run for 10 seconds. At time,  $t=0$ , the cell is filled with scalar  $S1$  concentration of 1 and scalar concentration in the cell is plotted in Fig. 4 over time. This shows exponential decay similar to the analytical solution mentioned above. This verifies that the first order reaction is correctly modelled and simulated.



*Fig 4 Scalar concentration vs time*

Since there is no convection and diffusion, we can expect the scalar concentration to be equal at all places of the cell. If we plot scalar concentration at various cross sections of the cell at a particular time, the plot obtained should be same. Fig. 5 shows the same. This suggests there is uniform decay of scalar S1 at all points in the cell.



*Fig 5 Scalar concentration at various cross section at time=5 seconds*

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

$$-r_A = -\frac{dC_A}{dt} = kC_A^n \quad (10)$$

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. 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. 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. (10),

$$C'_A = \frac{C_A}{C_{A0}} \quad (11)$$

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

we have

$$-r'_A = -\frac{dC'_A}{dt} = k \cdot C_{A0}^{n-1} C_A'^n = k' C_A'^n \quad (12)$$

$$\text{Where } k' = k \cdot C_{A0}^{n-1} \quad (13)$$

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.