

# Mathematical modeling of a spray-dryer

in Matlab and OpenFoam.

**Somma Andrea**

*Chemical plants and process operations management*

*Politecnico di Milano - 2022*



## Mathematical Formulation:

Balances on an evaporating particle and gas phase

- **MomB** on the particle;
- **MassB** on the particle
- **MassB** on the gas phase;
- **EnergyB** on the particle;
- **EnergyB** on the gas phase;
- **Velocity to axial coordinate conversion**

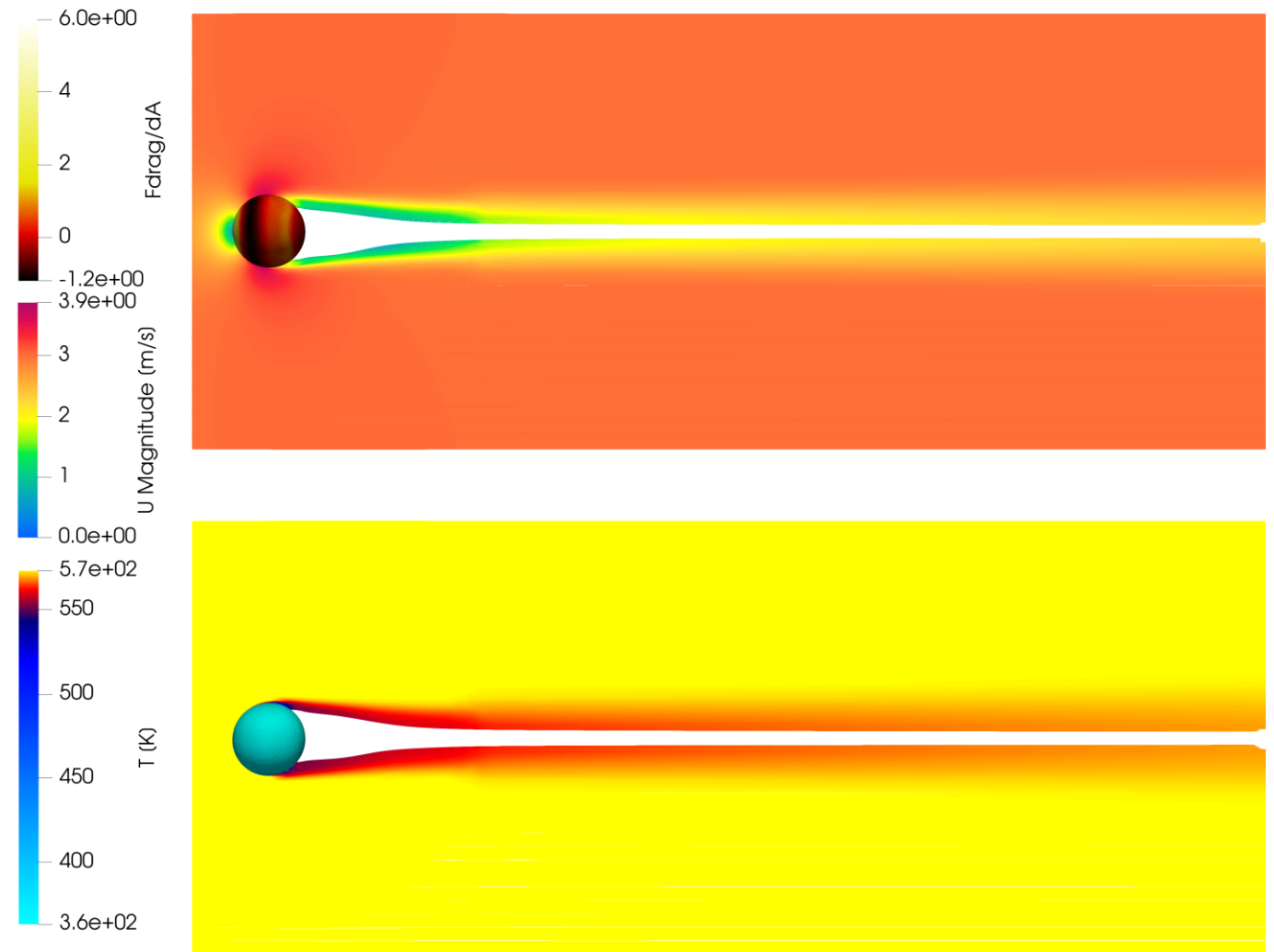
- ›  $m_p \frac{dv_s}{dt} = (\rho_p - \rho_g) V_p g - 3 f_D \mu \pi D_p v_s - v_s K_p S_p (P_w - P^0(T_g))$
- ›  $\frac{dm_p}{dt} = K_p S_p (P_w - P^0(T_g))$
- ›  $\frac{dG_i}{dt} = K_p S_p (P_w - P^0(T_g)) \eta A v_p$
- ›  $m_p c_{pL} \frac{dT_p}{dt} = h S (T_g - T_p) + K_p S_p (P_w - P^0(T_g)) \Delta H_{ev}$
- ›  $G c_p \frac{dT_g}{dt} = h S_p (T_p - T_g) \eta A v_p$
- ›  $\frac{dz}{dt} = v_p$

## Closure Terms: Evaluation of Nu,Sh,Cd

### Chosen relations:

- $Nu = 2 + 0.4Re^{0.5}Pr^{0.33}$
- $Sh = 2 + 0.4Re^{0.5}Sc^{0.33}$
- $Cd = 24/Re(1 + 0.14Re^{0.7})$

Simulation of a still, **cold** particle in a **hot** stream;  
drag coefficient, velocity field and temperature  
field are presented.



## Code Overview:

Everything is evaluated inside the integration loops

```
1 function spraydryer = solver(t,x)
2     ...
3     x_H2O = (mp - mp0*fat) / mp;
4     rho_avg = x_H2O * rhoL + (1 - x_H2O) * rho_milk;
5     V = mp / rho_avg;
6     dp = (6 * V / pi)^(1/3);
7     Dmin = 83.87e-6;
8     dp = max(Dmin, dp);
9     ...
10 end
```

***Adimensional numbers** and **coefficients** are calculated inside the integration function, furthermore to ensure **physical results** some considerations have been done.*

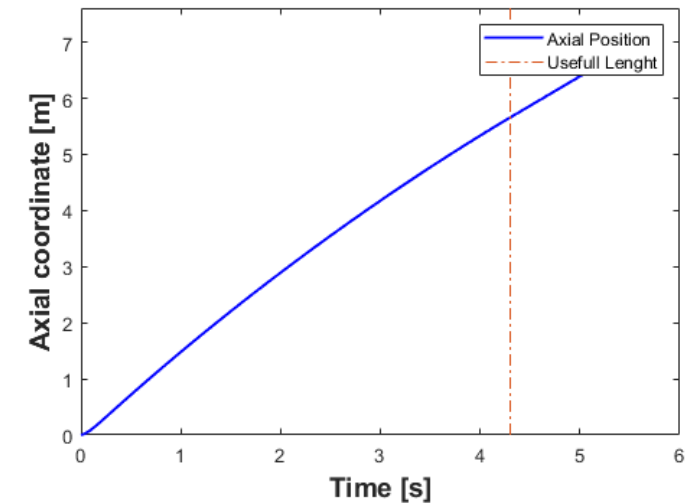
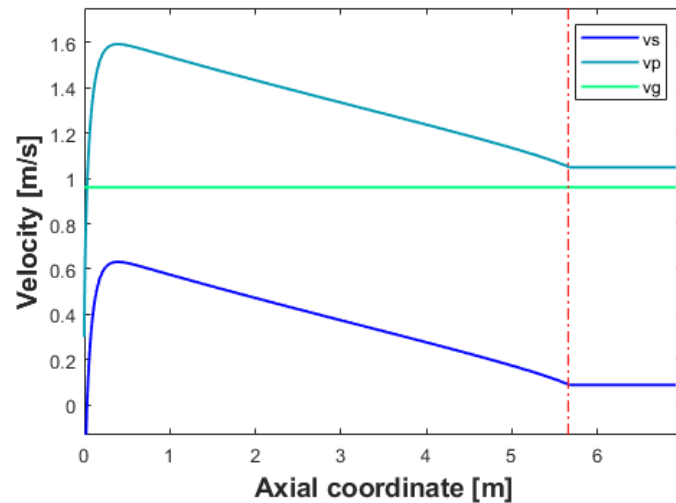
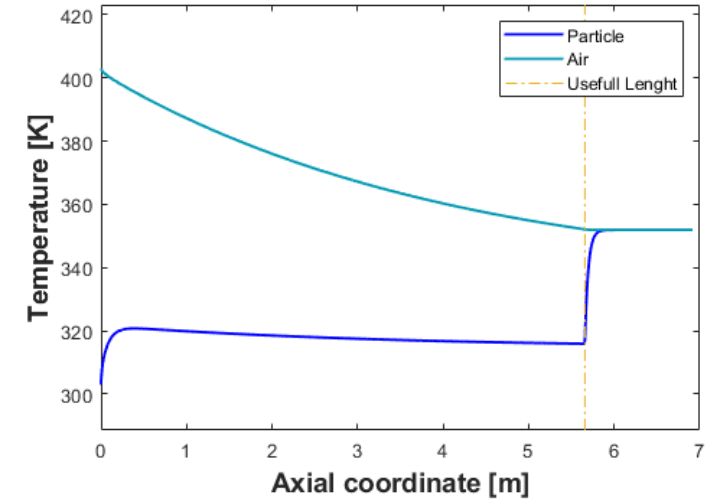
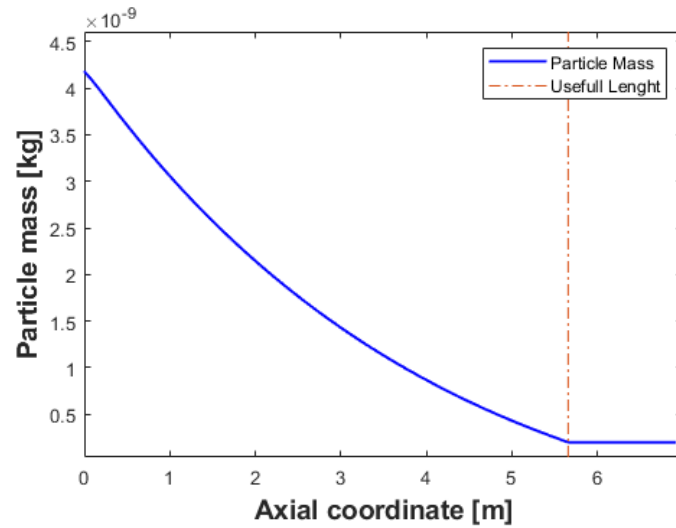
```
1 function spraydryer = solver(t,x)
2     ...
3     Re = abs(rhoG * vs * dp / muG);
4     Pr = muG * CpG / kG;
5     Sc = muG / rhoG / Diff;
6     Nu = 2 + 0.4 * Re^0.5 * Pr^(1/3);
7     Sh = 2 + 0.4 * Re^0.5 * Sc^(1/3);
8     f = (1 + 0.14*Re^0.7);
9     h = Nu * kG / dp;
10    Kc = Sh * Diff / dp;
11    Kpw = Kc / R / Tg * MWwater;
12    ...
13 end
```

```
1 function spraydryer = solver(t,x)
2     ...
3     mmin = mp0 * fat;
4     mp = max(mmin, mp);
5     ...
6     if mp > mmin + mmin/10000
7         spraydryer(1) = Kpw * Sp * (Pw - P0);
8     else
9         spraydryer(1) = 0;
10    end
11    ...
12 end
```

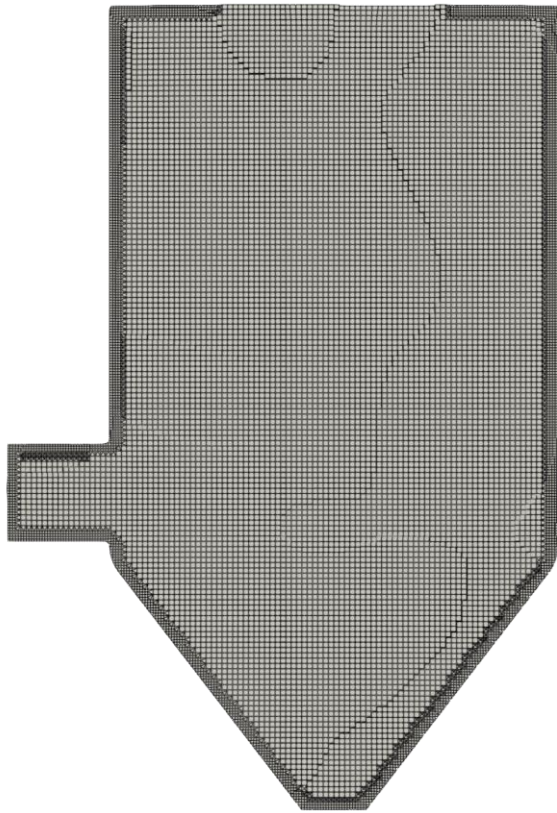
# Results: Realistic trends obtained

Proposed results present **realistic trends** for the calculated fields:

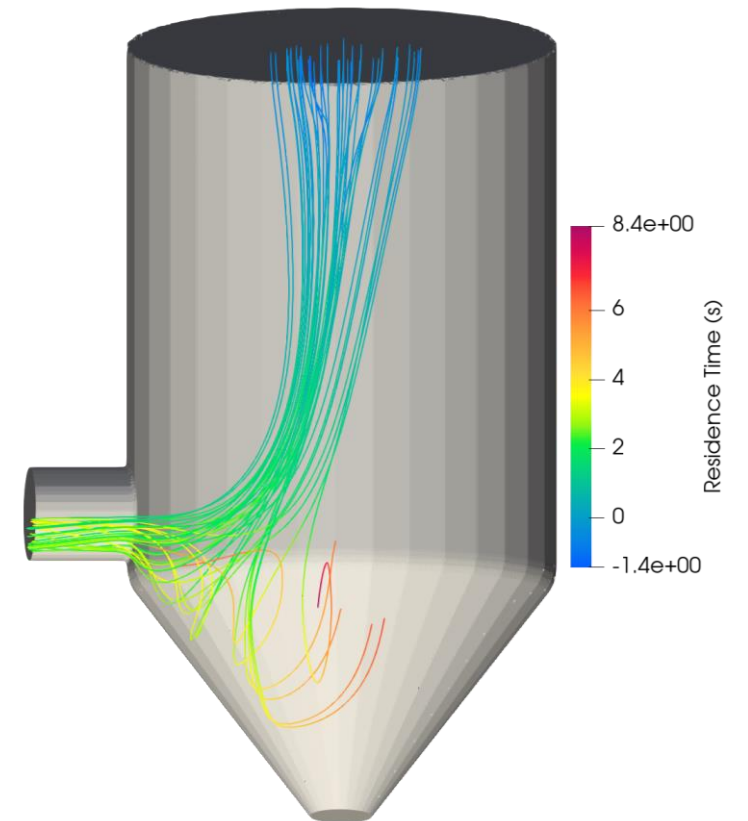
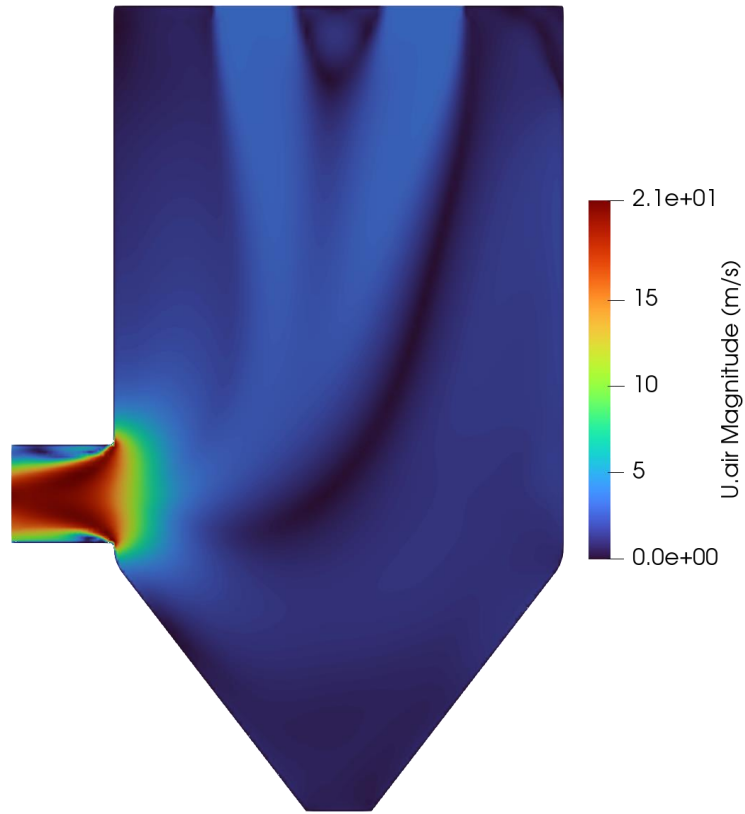
- Particle mass reaching a **minimum**;
- $p$ -velocity almost reaching **G-velocity**;
- **Temperature** sharply increasing after the consupcion of water.



## OpenFoam setup: Ras-model, K-omega-SST

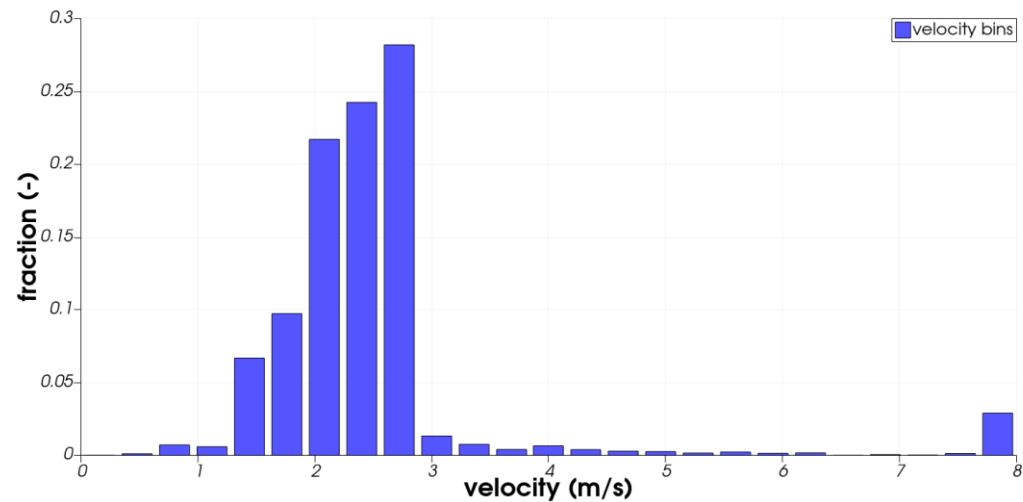


**3D-case** hex-dominant mesh 2Mln cells

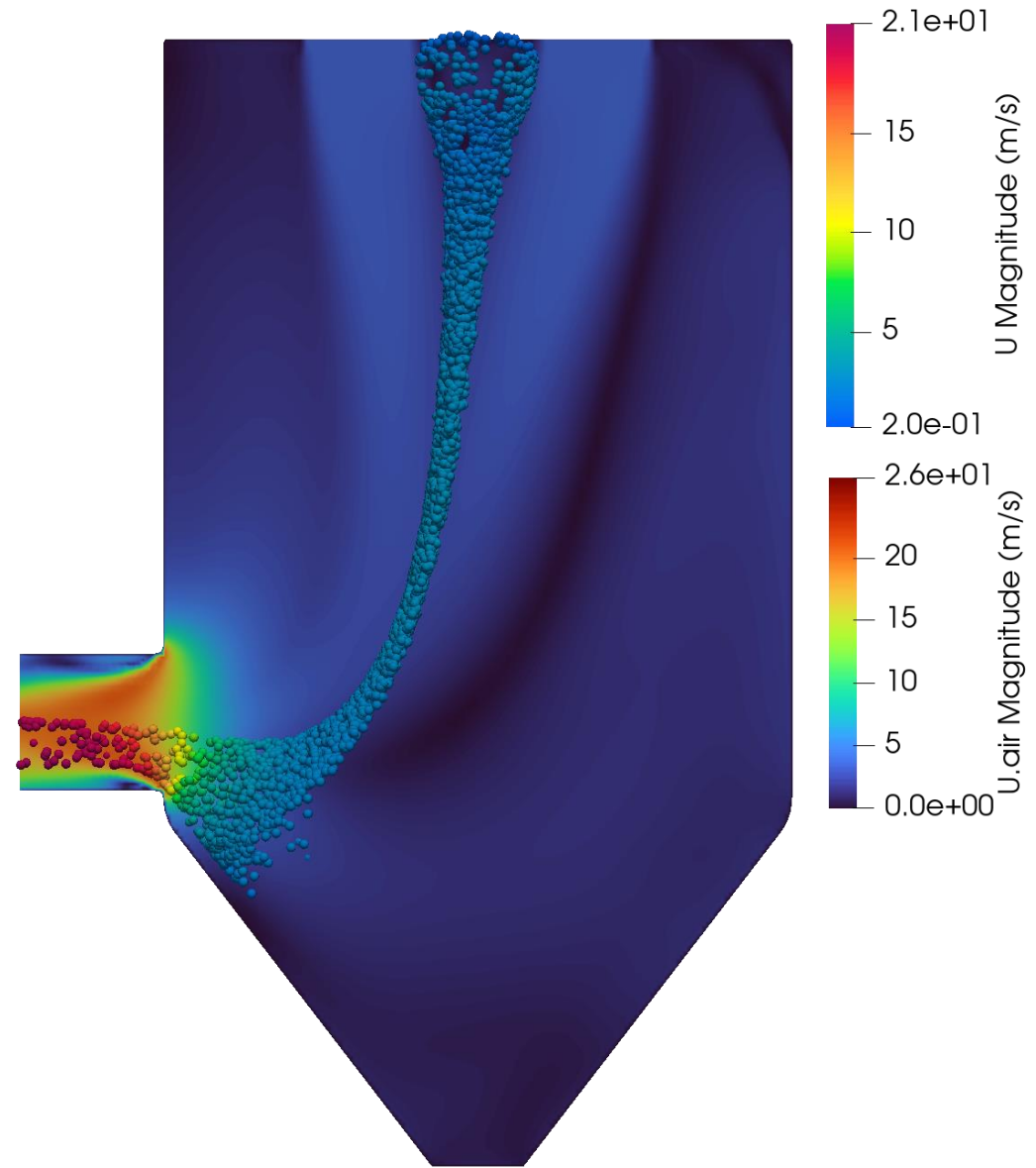


**Velocity field** and streamline **residence time**, in SI units. Recirculations and velocity gradients are **not** negligible.

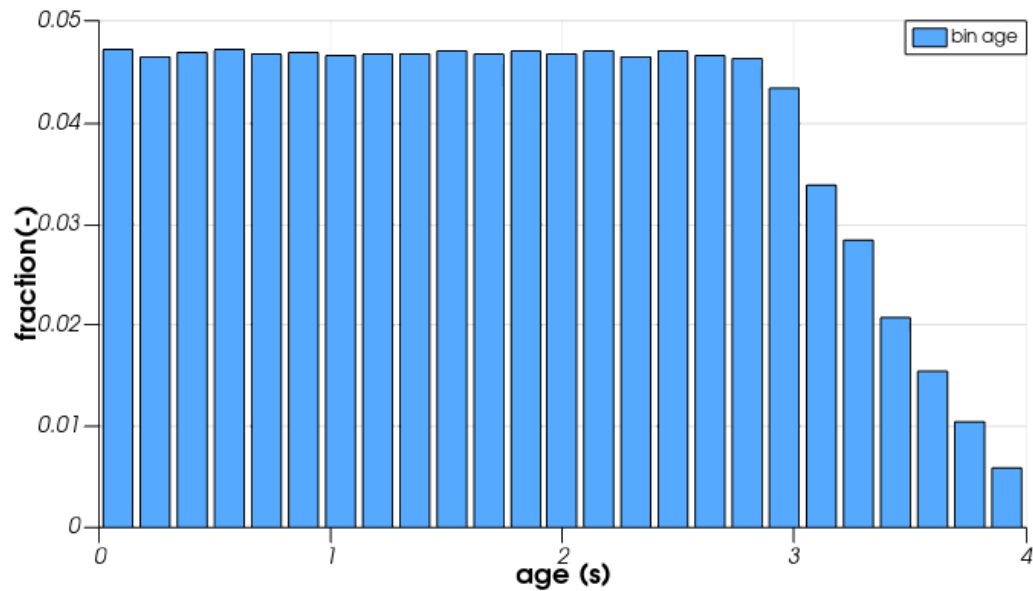
# OpenFoam setup: Ras-model, denseParticleFoam solver



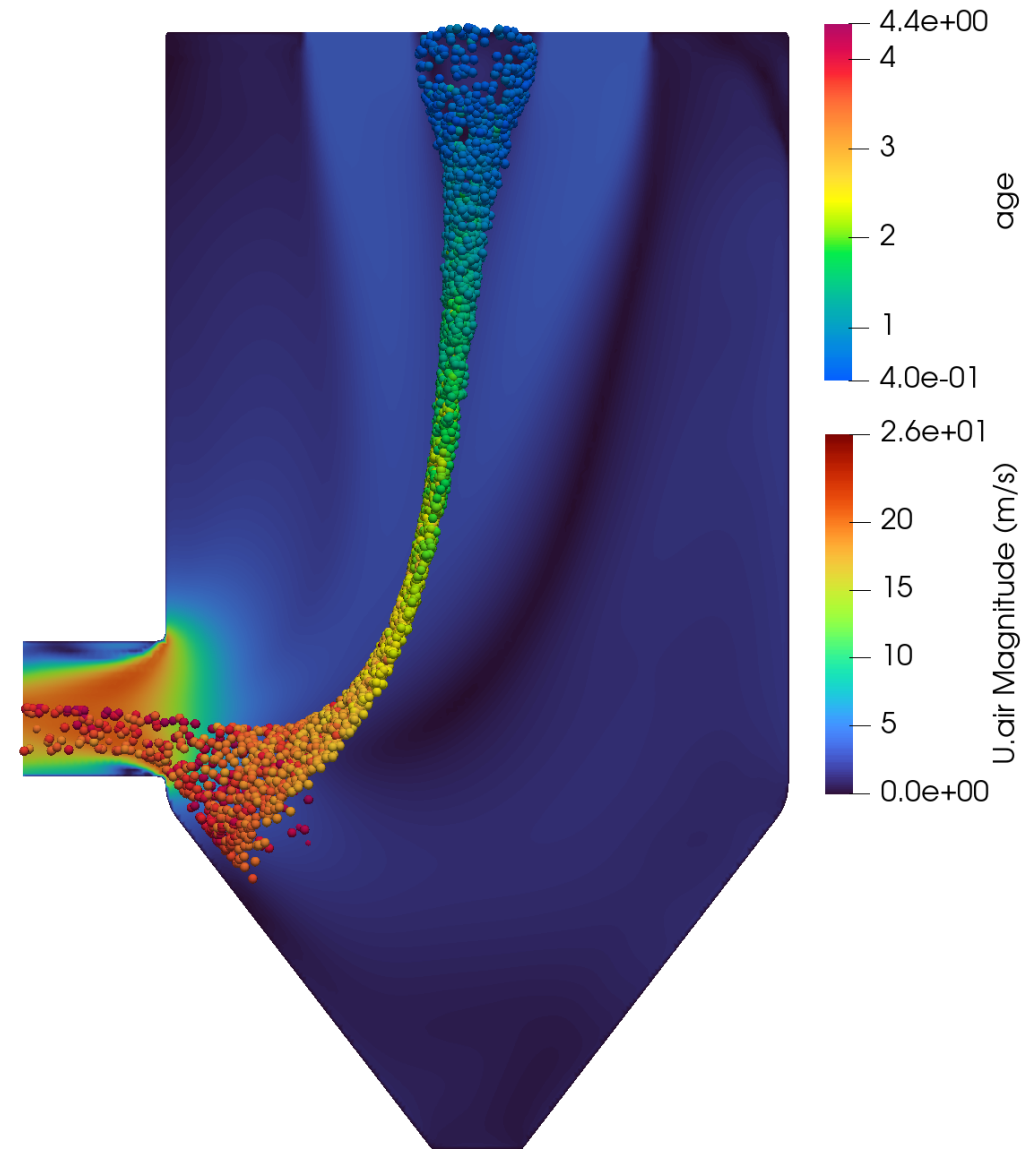
Particles tend to **strictly follow** the air stream, given the almost negligible mass (very mass-diluted system). The displayed graph is at 4s after the first injection.



# OpenFoam setup: Ras-model, denseParticleFoam solver



Particles **have not** the required age, due to the non uniform velocity field. The displayed graph is at 4s after the first particle injection.



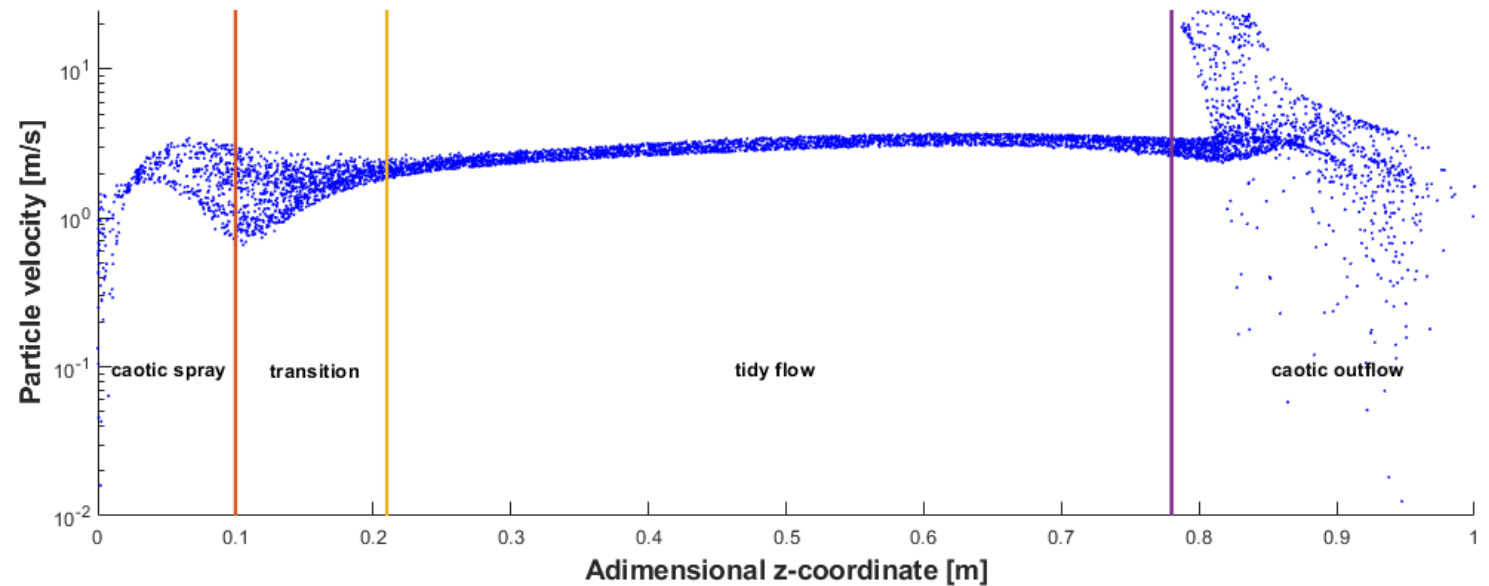


# Jet correlation:

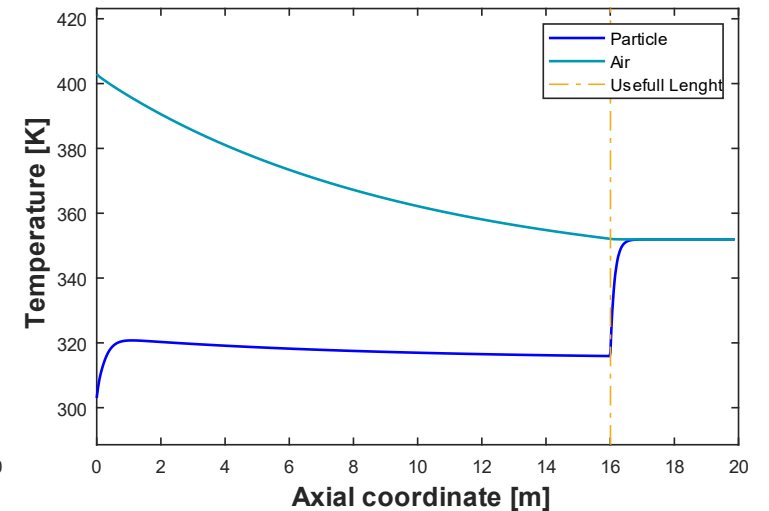
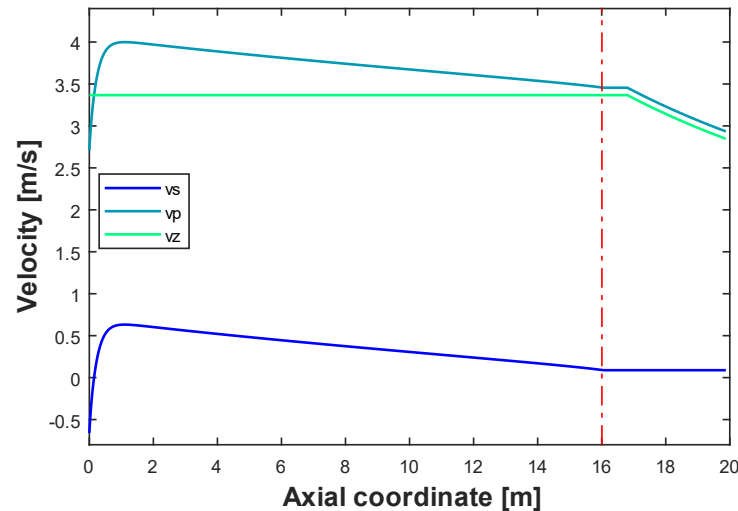
Riyaz and  
Ahmad 2014

$$v_z = \begin{cases} U_{exit}, & \text{if } 0 \leq z \leq x \\ 6.11 D_{outlet} U_{exit} \frac{1}{z}, & \text{if } (z > x \ \& \ v_z > v_{dev}) \\ v_{dev}, & \text{otherwise} \end{cases}$$

A new correlation is needed to ensure that the particle is affected by **a realistic velocity**, with this a new corrected 1D-model can be developed.

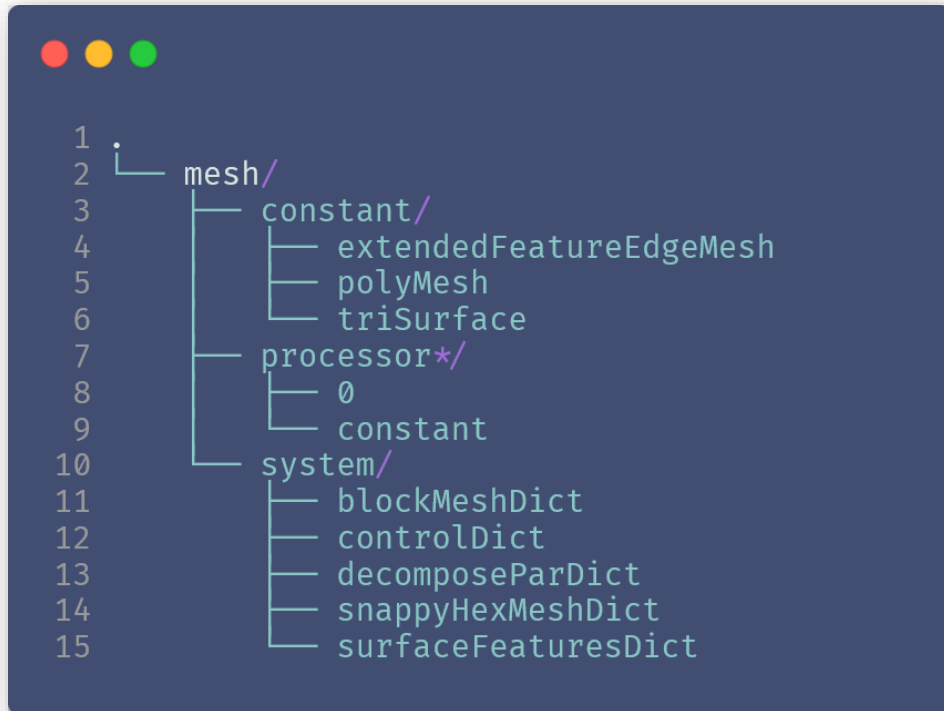


D ratio = 2, length = 16.0072m



# Workflow Overview:

Mesh, Boundary conditions,  
Solve!



Typical folder tree of a **mesh** in OpenFoam, with SnappyHexMesh.



Typical folder tree of a **denseParticleFoam** case in OpenFoam, cloudProperties is the file where all particle properties/models are defined.

# Thank you for the attention.

 [github.com/sommaa](https://github.com/sommaa)

