UNIVERSITY OF GENOVA

POLYTECHNIC SCHOOL

DIME

Department of Mechanical, Energy, Management
and Transportation Engineering

Master of Science Thesis

in

Mechanical Engineering

Implementation of a two-stage drying model for liquid drops containing insoluble solids in OpenFOAM for spray drying applications

Supervisor:

Chiar.^{mo} Prof. Ing. Jan Oscar Pralits (Unige)

Co Supervisor:

Dott. Ing. Matteo Colli (Unige) & Marco Atzori (KTH)

Candidate:

Giovanni Beati

July 2017

Implementation of a two-stage drying model for liquid drops containing insoluble solids in OpenFOAM for spray drying applications

Giovanni Beati

supervised by
Prof. Ing. Jan Oscar Pralits (Unige)
&
Dott. Ing. Matteo Colli (Unige) & Marco Atzori (KTH)

Abstract

The work presented in this thesis is integrated in a well started project that Dott. Ing. Stefano Pastorino presented last year as his master thesis. His work concerned the development of a numerical model of a spray dryer with OpenFoam, an open-source CFD software freely available and without license costs. The usage of CFD technique allowed to understand the complexity of flow fields of drying agent as well as droplet/particle motion. In his work Stefano studied the evaporation of water droplets and the motion of solid particles (olive pomace extract mixed to maltodextrin) in a separated way. The aim of this work is to start the implementation of a library in OpenFoam that is able to consider the liquid phase as well as the solid phase in a two stage drying model. A number of models of the two stage drying process were first implemented and tested in a simplified environment (Python). Then

one of the models tested with both drying stages has been implemented in OpenFOAM®, and a comparison of the first drying stage has been made with a model already distributed with OpenFOAM®.

Sommario

Il lavoro presentato in questa tesi si integra in un progetto ben avviato e iniziato lo scorso anno dal Dott. Ing. Stefano Pastorino. L'obiettivo del suo lavoro è stato quello di svulippare un modello numerico che simulasse il funzionamento di uno Spray Dryer utilizzando OpenFOAM®, un software CFD completamente gratuito e scaricabile senza alcun costo di licenza. L'utillizzo della fluidodinamica computazionale ha permesso di capire la complessità del campo di moto dell' aria utilizzata come agente asciugante così come il moto delle particelle inserite nel dominio computazionale. Stefano, nel suo lavoro ha studiato separatamente l'evaporazione di gocce d'acqua e il moto di particelle solide. L'obiettivo di questo lavoro è stato quello di studiare un modello di evaporazione basato su due stadi, che fosse specifico per l'applicazione dello Spray Dryer, ovvero per gocce che contengono una frazione iniziale di solido. Il primo stadio riguarda l'evaporazione della componente liquida; quando viene raggiunto un valore critico di umidità il modello prevede la formazione di una crosta solida, di spessore via via crescente, che avvolge un nucleo contenente ancora una certa frazione di liquido.

Inizialmente il modello è stato implementato in un ambiente semplificato (Python) che ha permesso una più immediata visualizzazione dei risultati. Successivamente questo modello è stato inserito in OpenFOAM® e, i risultati forniti dalla nostra nuova applicazione, sono stati confrontati nel caso di pure gocce d'acqua con quelli forniti da un solutore già implementato nel software. La differenza fondamentale tra la nostra nuova applicazione e quella già esistente è che le particelle che hanno una massa al di sotto di un certo valore non vengono eliminate ma vengono continuamente tracciate all'interno del dominio di calcolo. A queste particelle è applicato un modello di secondo stadio molto semplificato: infatti esse vengono considerate come particelle completamente asciutte e quindi solide, le quali tendono a scaldarsi in quanto in contatto con l'aria calda.

Acknowledgements

I would really like to thank my supervisor Prof. Jan Pralits. From the very first time we decided to cooperate on this project, he has provided me immediately all the necessary tools for the development of the work. He also put me in touch with the "right" people and he has followed me until the end, helping me to organize the whole work. I also have to thank Marco Atzori; he always followed me, and gave me all the support necessary with weekly Skype meetings despite his endless commitments; indeed he is a Phd student at the KTH Royal Institute of Technology in Sweden, but it was like he was here always trying to resolve my doubts; of course without him it would be much harder to conclude this work. A big thank goes also to Matteo Colli for his great amount of knowledge about the topic of this project, and his precious advices on how to carry out the work;

There are no words to describe the kind of help that came from my family. My mother, my father and my brother always gave me support during this long period of studies where there were very difficult moments and without them I would not be able to overcome. I thank also my student mates Andrea e Alberto. With them the study has been much more interesting and the constant comparison between our opinions has been very positive.

Last but not least I have to thank Alice. I am sure that without her presence everything would have been much less tolerable. I also have to thank her for having endured me for all this time, alone it would have been much more difficult.

A big thanks goes also to **KTH Royal Institute of Technology**; indeed part of the results presented in the thesis have been obtained using resources of the Swedish National Infrastructure for Computing (SNIC), allocation SNIC 2016/34-10 (p.i. Prof. Dan Henningson), for courtesy of Prof. Philipp Schlatter, KTH Mechanics

Contents

Chapter 1

Introduction

The topic of this thesis concerns the area of spray drying which is a well-established industrial process for converting liquid feed materials into a dry powder. In particular we focused on the implementation of a two stage drying model for liquid drops containing insoluble solids. Once compared with experimental data, this model has been implemented in OpenFOAM®, a CFD open-source software.

In spray drying operations, CFD simulation tools are now often used, because measurements of air flow, temperature, particle size and humidity within the drying chamber are very difficult and expensive to obtain in large-scale dryers [2]. In most cases the spray dryer modelling is performed using a mixed Eulerian/Lagrangian approach, in which the single phase Reynolds Averaged Navier Stokes (RANS) or Unsteady Reynolds Averaged Navier Stokes (URANS) equations are solved to determine the flowfield and the droplets are modelled using the Lagrangian technique [9].

A large amount of literature exists about this topic, and very detailed studies have been carried out by many research groups and authors. For example in [15] and in [27] it is presented a study of the air flow inside a tall form spray dryer, including the effect of different turbulence models and their influence on particle trajectories and their residence time; in [19] there are studies on the interaction of droplets with the spray dryer walls, modelling the rebound as a function of droplets moisture content; also particles agglomeration can be important and it determines the character of the final product as well described in [9]; moreover in [24], [22], [20], [21] the description of accurate drying models are found both for insoluble or dissolved solids with particular attention paid on the second stage, that is the one related to the formation of a solid shell around a wet core. However, all these CFD models use commercial software, especially ANSYS Fluent is the most used.

In this context Gea Niro, a core technology centre in GEA Process Engineering, developed an ANSYS suit called DRYNETICS which is a very specific tool box for spray drying modelling that is able to provide the solution by incorporating real-world measurements into the CFD software. Their main idea is that every liquid feed has its own characteristics, and these can be determined accurately only with real tests;

The aim of this project is to continue the work of Stefano Pastorino that, in his master thesis, began to treat spray drying modelling in OpenFOAM®. Indeed he focused on the construction of the geometry of a tall-form spray dryer, and he modelled the evaporation of pure liquid drops, using a built-in OpenFOAM® solver. Our goal is to start the implementation of a new Lagrangian solver in OpenFOAM® suitable for modelling the drying kinetic of liquid drops with insoluble solids.

The thesis is organised as following:

- **Part I**: the aim of this part is to provide the fundamental concepts about spray drying;
- **Part II**: this parts deals with the fundamental concepts about Computational Fluid Dynamics, and with the introduction of the two stage drying model;
- **Part III**: in this part is given an overview of the open source CFD software (OpenFOAM®) is given with some details about the programming language;
- **Part IV**: here the numerical dicretization of the equations for the two stage drying model and their implementation in the Python environment is discussed;
- **Part V**: in this part the case study is presented, with details about the modified geometry and the one developed by Stefano Pastorino and the implementation in OpenFOAM® of the evaporation model;
- **Part VI**: finally the first section of this part presents the results about the two drying stage model from the Python code and then in the second one the results from the OpenFOAM® simulations.

Part I

Description of spray dryer

Chapter 2

Description of a Spray Dryer

Transformation of a liquid feed containing solid fraction into particles by supplying the feed as a spray into a chamber with a hot drying agent is named spray drying. Spray drying is a widely used process in many industries, among them are food manufactures, pharmaceutical, chemical and biochemical industries. The spray drying process involves multiphase flow with heat, mass and momentum transfer between the three-dimensional, complex, swirling, drying gas flow (continuous phase) and the discrete phase (droplets/particles). Fig.2.1 shows that spray drying usually involves three stages of operation:

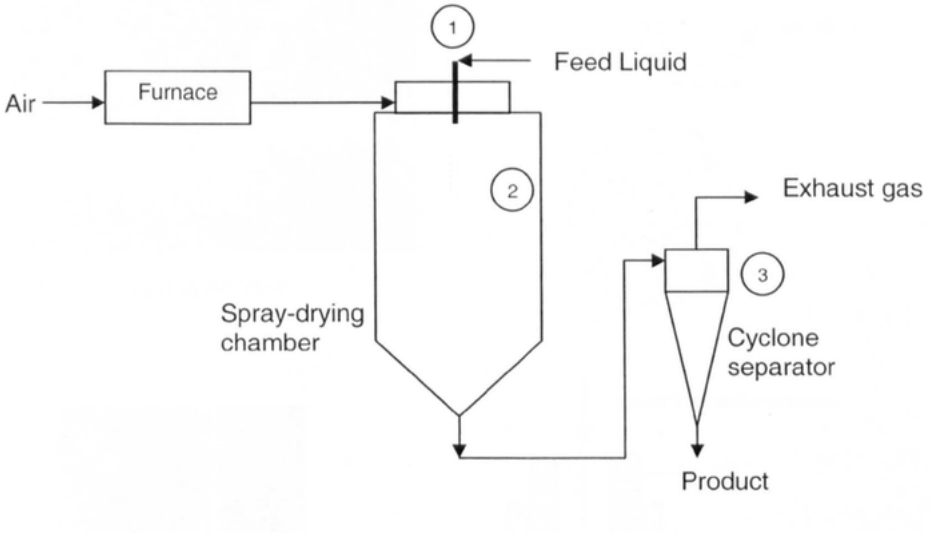


Figure 2.1: *The process stages of spray drying [2].*

- atomization of liquid feed into a spray chamber;
- contact between the spray and the drying agent;

- separation of dried products from air stream;

The atomization process is essentially one in which bulk liquid is converted into small drops. The quality of the final product is heavily dependent on the choice of the atomizer type. An atomizer nozzle can take many forms: there are centrifugal or rotary atomizer, pressure nozzle atomizer, two-fluid nozzle atomizer. For centrifugal or rotary atomizer the size of droplets produced from the nozzle varies directly with feed rate and feed viscosity and inversely with wheel speed and wheel diameter Pressure nozzle atomizer in which droplet size varies directly again with feed rate and feed viscosity, but inversely with pressure. Finally, atomization can be obtained also through the two-fluid nozzle atomizer, meaning that a shear field created by compressed air, atomizes the liquid and produces a wide range of droplet sizes.

During spray-air contact, the hot drying gas (air in most cases) can be blown in the same direction as the sprayed liquid (co-current flow) or it can be against the flow from the atomizer (counter-current flow). With the co-current configuration the hot air and droplets are in contact from the beginning of the injection and, because of high rate of evaporation, their temperature is kept low. Along the drier, droplets moisture content decreases as well as air temperature resulting in smaller heat transfer rate from the continuous phase to the dried particles. In the counter-current arrangement, on the other hand, the spray inlet corresponds to the drying medium outlet and this arouses a final product temperature higher than the exhaust drying agent temperature and that is why this configuration is used for non-heat-sensitive products only [2].

Regarding the last stage, the separation of dried particles from air flow can be done in different ways depending on the operating conditions such as particle size, shape, bulk density and powder outlet position. For example, dried particles can be picked up at the base of the dryer and absorbed by a cyclone separator or a screw conveyor. Other equipment useful to collect the dry powder are bag filters and electrostatic precipitators. The two main designs of spray dryer commonly used are the short-form and tall-form driers shown in Fig.2.2.

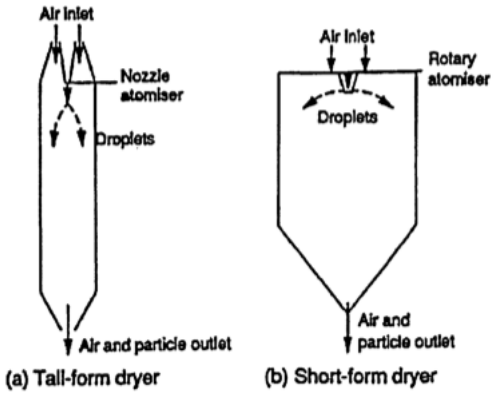


Figure 2.2: *Schematic description of main types of spray driers [7].*

Short-form dryers are characterized by a restrained aspect ratio meaning that the height-diameter ratio is of around 2:1 while tall-form dryers have a height-diameter ratio greater than 5:1. In the latter case dryers have less complex flow patterns than short-form dryers, but they are afflicted by an higher percentage of particles impacting on the cylindrical wall which is a negative effect on the final product quality.

Part II

Governing equations

Chapter 3

Fundamental concepts in fluid mechanics

3.1 Conservation laws

The principle of conservation states that for an isolated system certain physical measurable quantities are conserved over a local region. This conservation principle or conservation law is an axiom that cannot be proven mathematically but can be expressed by a mathematical relation. Laws of this type govern several physical quantities such as mass, momentum, and energy (the Navier-Stokes equations)[12]. The conservation laws involving fluid flow and related transfer phenomena can be mathematically formulated following either a Lagrangian (material volume, MV) or an Eulerian (control volume) approach. Assuming a continuous phase the most common method of describing fluid flow is the fixed reference system Eulerian approach that is synthetically presented below. A short description of the Lagrangian method will be introduced in the next section.

3.1.1 Continuity equation

The principle of conservation of mass indicates that in the absence of mass sources and sinks, a region will conserve its mass on a local level [12]. Being ρ the density, through the application of the Reynolds transport theorem, the general expression for conservation of mass as applied to a control volume will be:

$$\int_V \frac{\partial \rho}{\partial t} dV + \int_S \rho \mathbf{V} \cdot \mathbf{n} dS = 0, \tag{3.1}$$

where S is the surface of the control volume V. Thanks to the divergence theorem – and noticing that the conservation of mass should be respected for every control

volume – this equation can be written in a differential form, called the continuity equation:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{V}) = 0 \tag{3.2}$$

3.1.2 Conservation of Linear Momentum

Through application of the Reynolds transport theorem and divergence theorem, the general expression for conservation of linear momentum as applied to a control volume is:

$$\frac{\partial (\rho \mathbf{v})}{\partial t} + \nabla \cdot (\rho \mathbf{v}\mathbf{v}) = \mathbf{f}. \tag{3.3}$$

Where $\mathbf{f} = \mathbf{f_s} + \mathbf{f_b}$ is the sum of the external surface forces $\mathbf{f_s}$ and body forces $\mathbf{f_b}$ acting on the control volume.

Surface forces

The forces acting on the control volume surface are due to pressure and viscous stresses which can be expressed in terms of the total stress tensor $\boldsymbol{\sigma}$ that in Cartesian coordinates is given by:

$$\boldsymbol{\sigma} = \begin{pmatrix} \sigma_{xx} & \tau_{xy} & \tau_{xz} \\ \tau_{yx} & \sigma_{yy} & \tau_{yz} \\ \tau_{zx} & \tau_{zy} & \sigma_{zz} \end{pmatrix} = \begin{pmatrix} -P & 0 & 0 \\ 0 & -P & 0 \\ 0 & 0 & -P \end{pmatrix} + \begin{pmatrix} \tau_{xx} & \tau_{xy} & \tau_{xz} \\ \tau_{yx} & \tau_{yy} & \tau_{yz} \\ \tau_{zx} & \tau_{zy} & \tau_{zz} \end{pmatrix} = -p\mathbf{I} + \boldsymbol{\tau}, \tag{3.4}$$

where $\mathbf{I}$ is the identity tensor, p the pressure and $\boldsymbol{\tau}$ is the deviatoric of viscous stress tensor. The pressure is the negative part of the mean of the normal stresses and is given by:

$$p = -\frac{1}{3}(\sigma_{xx} + \sigma_{yy} + \sigma_{zz}). \tag{3.5}$$

Hence the surface force acting on a differential surface element dS is:

$$\int_S \mathbf{f_s} dS = \int_A \boldsymbol{\sigma} \cdot \mathbf{n} dA = \int_V \nabla \cdot \boldsymbol{\sigma} dV \Rightarrow \mathbf{f_s} = \nabla \cdot \boldsymbol{\sigma} = -\nabla p + (\nabla \cdot \boldsymbol{\tau}). \tag{3.6}$$

Body forces

Body forces are forces per unit volume and the predominant ones are given below:

- Gravitational forces $\mathbf{f_b} = \rho \mathbf{g}$, due to the presence of a gravitational field
- Coriolis and centrifugal forces, respectively $\mathbf{f_b} = -2\rho(\boldsymbol{\omega} \times \mathbf{v}) - \rho(\boldsymbol{\omega} \times (\boldsymbol{\omega} \times \mathbf{r}))$, due to a rotating frame of reference

Hence introducing the expressions of surface and body forces in Eq.(3.3) the general conservative form of the momentum equation is obtained as:

$$\frac{\partial(\rho\mathbf{v})}{\partial t} + \nabla \cdot (\rho\mathbf{v}\mathbf{v}) = -\nabla p + (\nabla \cdot \boldsymbol{\tau}) + \mathbf{f_b}. \tag{3.7}$$

To procede further the type of fluid should be specified in order to relate $\boldsymbol{\tau}$ with the other flow variables. For a Newtonian fluid the stress tensor is a linear function of the strain rate and is given by:

$$\boldsymbol{\tau} = \mu(\nabla \boldsymbol{v} + (\nabla \boldsymbol{v})^T) + \lambda(\nabla \cdot \mathbf{v})\mathbf{I}, \tag{3.8}$$

where μ is the molecular viscosity, λ the bulk viscosity coefficient usually set equal to $\lambda = \frac{2}{3}\mu$. Taking the divergence of Eq.(3.8) and substituting in Eq.(3.7) the final conservative form of the momentum equation for Newtonian fluids becomes:

$$\frac{\partial(\rho\mathbf{v})}{\partial t} + \nabla \cdot (\rho\mathbf{v}\mathbf{v}) = \nabla \cdot (\mu\nabla \boldsymbol{v}) - \nabla p + \nabla \cdot (\mu(\nabla \mathbf{v}^T)) + \nabla(\lambda\nabla \cdot \mathbf{v}) + \mathbf{f_b}. \tag{3.9}$$

For incompressible flows the divergence of velocity vector is zero, $\nabla \cdot \mathbf{v} = 0$, and for constant molecular viscosity the momentum equation can be further simplified:

$$\frac{\partial(\rho\mathbf{v})}{\partial t} + \nabla \cdot (\rho\mathbf{v}\mathbf{v}) = -\nabla p + \mu\nabla^2\mathbf{v} + \mathbf{f_b}. \tag{3.10}$$

3.1.3 Conservation of Energy

The conservation of energy (the first law of thermodynamics) simply states that energy can be neither created nor destroyed during a process; it can only change from one form (mechanical, kinetic, chemical, etc.) into another. Consequently, the sum of all forms of energy in an isolated system remains constant. Considering a material volume MV of mass m, density ρ, and moving with a velocity $\mathbf{v}$ the total energy E can be written as:

$$E = m(\hat{u} + \frac{1}{2}\mathbf{v} \cdot \mathbf{v}), \tag{3.11}$$

where $\hat{u}$ is the internal energy per unit mass. The first law of thermodynamic states that the rate of change of the total energy of the material volume is equal to the rate of heat addition and work extraction through its boundaries:

$$(\frac{dE}{dt})_{MV} = \dot{Q} - \dot{W}. \tag{3.12}$$

Defining $e = \hat{u} + \frac{1}{2}\mathbf{v}\cdot\mathbf{v}$ as the total energy per unit mass and considering that:

$$\dot{W} = \dot{W}_s + \dot{W}_b, \tag{3.13}$$

$$\dot{W}_s = -\int_S (\mathbf{f_s}\cdot\mathbf{v})dS = -\int_S (\boldsymbol{\sigma}\cdot\mathbf{v})\cdot\mathbf{n}dS = -\int_V -\nabla\cdot(p\mathbf{v}) + \nabla\cdot(\boldsymbol{\tau}\cdot\mathbf{v})dV, \tag{3.14}$$

$$\dot{W}_b = -\int_V (\mathbf{f_b}\cdot\mathbf{v}), \tag{3.15}$$

$$\dot{Q} = \dot{Q}_s + \dot{Q}_v, \tag{3.16}$$

$$\dot{Q}_s = -\int_S \dot{q}_s\cdot\mathbf{n}dS = -\int_V \nabla\cdot\dot{q}_s dV, \tag{3.17}$$

$$\dot{Q}_v = \int_V \dot{q}_v dV. \tag{3.18}$$

Using the Reynolds transport theorem Eq.(3.12) becomes:

$$\begin{aligned}\Big(\frac{dE}{dt}\Big)_{MV} = \dot{Q} - \dot{W} = \int_V \Big(\frac{\partial}{\partial t}(\rho e) + \nabla\cdot(\rho\mathbf{v}e)\Big)dV = \\ = -\int_V \nabla\cdot\dot{q}_s dV + \int_V -\nabla\cdot(p\mathbf{v}) + \nabla\cdot(\boldsymbol{\tau}\cdot\mathbf{v})dV + \int_V (\mathbf{f_b}\cdot\mathbf{v})dV + \int_V \dot{q}_v dV.\end{aligned} \tag{3.19}$$

Collecting terms within the volume integral and setting the integrand equal to zero gives:

$$\frac{\partial}{\partial t}(\rho e) + \nabla\cdot(\rho\mathbf{v}e) = -\nabla\cdot\dot{q}_s - \nabla\cdot(p\mathbf{v}) + \nabla\cdot(\boldsymbol{\tau}\cdot\mathbf{v}) + \mathbf{f_b}\cdot\mathbf{v} + \dot{q}_v. \tag{3.20}$$

In order to write the energy equation with temperature as the main variable some constraints have to be imposed [12]. Denoting with h the specific enthalpy and assuming a Newtonian fluid it's possible to express $h = f(p,T)$ the variation of enthaply dh can be written as:

$$dh = \Big(\frac{\partial h}{\partial T}\Big)_p dT + \Big(\frac{\partial h}{\partial p}\Big)_T dp. \tag{3.21}$$

Using the thermodynamic relation:

$$\Big(\frac{\partial h}{\partial p}\Big)_T = v - T\Big(\frac{\partial v}{\partial T}\Big)_p, \tag{3.22}$$

where v is the specific volume, the expression for dh can be modified to

$$dh = c_p dT + \Big[v - T\Big(\frac{\partial v}{\partial T}\Big)_p\Big]. \tag{3.23}$$

After some manipulation, in order to express Eq.(3.20) in terms of specific enthalpy h and introducing Eq.(3.23), the energy equation with T as the main variable can be written as:

$$c_p\Big[\frac{\partial}{\partial t}(\rho T) + \nabla\cdot(\rho\mathbf{v})\Big] = -\nabla\cdot\dot{q}_s - \Big(\frac{\partial(Ln\rho)}{\partial(LnT)}\Big)_p\frac{Dp}{Dt} + (\boldsymbol{\tau}:\nabla\mathbf{v}) + \dot{q}_v. \tag{3.24}$$

The heat flux $\nabla \cdot \dot{q}_s$ represents heat transfer by diffusion, which is a phenomenon occurring at the molecular level and is governed by Fourier's law according to:

$$\dot{q}_s = -(k\nabla T), \tag{3.25}$$

where Ln is the natural logarithm and k is the thermal conductivity of the substance. The above equation states that heat flows in the direction of temperature gradient and assumes that the material has no preferred direction for heat transfer with the same thermal conductivity in all directions (the medium is isotropic). Introducing Eq.(3.25) in Eq.(3.24), defining Ψ and Φ as:

$$\Psi = \Big(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z}\Big)^2, \tag{3.26}$$

$$\Phi = 2\Big[\Big(\frac{\partial u}{\partial x}\Big)^2 + \Big(\frac{\partial v}{\partial y}\Big)^2 + \Big(\frac{\partial w}{\partial z}\Big)^2\Big] + \Big(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\Big)^2 + \Big(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}\Big)^2 + \Big(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}\Big)^2, \tag{3.27}$$

and expanding the double dot product the energy equation in terms of T becomes:

$$\begin{aligned}\frac{\partial}{\partial t}(\rho c_p T) + \nabla \cdot (\rho c_p \mathbf{v} T) =& \nabla \cdot (k\nabla T) \\ &+ \rho T \frac{Dc_p}{Dt} - \Big(\frac{\partial(\ln \rho)}{\partial(\ln T)}\Big)_p \frac{Dp}{Dt} + \lambda\Psi + \mu\Phi + \dot{q}_v.\end{aligned} \tag{3.28}$$

This equation is rarely solved in its full form and depending on the physical situation several simplified versions can be developed. For example the dissipation term Φ is negligible except for supersonic speed with large velocity gradient [12]. For incompressible fluids both Ψ and $\left(\frac{\partial(\ln \rho)}{\partial(\ln T)}\right)_p$ are equal to zero and Eq.(3.28) is reduced to

$$\frac{\partial}{\partial t}(\rho c_p T) + \nabla \cdot (\rho c_p \boldsymbol{v} T) = \nabla \cdot (k\nabla T) + \dot{q}_v + Q^T. \tag{3.29}$$

For the case of a solid, the density is constant, the velocity is zero, and if changes in temperature are not large then the thermal conductivity may be considered constant and the equation of energy becomes:

$$\rho c_p \frac{\partial T}{\partial t} = k\nabla^2 T + \dot{q}_v. \tag{3.30}$$

3.2 Turbulence modelling

Most of industrial applications involve turbulent flows. However a precise definition is somewhat difficult and all that can be done is a brief outline of some of its characteristics [28]. One characteristic is the irregularity, or randomness, of all turbulent flows. This makes a deterministic approach to problems including turbulence

impossible; instead, and statistical methods have to be relied on. Another important turbulence feature is its diffusivity that leads to rapid mixing, thereby increasing transfer rates of momentum, heat and mass throught the flow domain. Turbulent flows always occur at a large Reynolds number, and often originate as the instability of laminar flows with increasing Reynolds numbers. Instabilities are related to the interaction of viscous terms and nonlinear inertia terms in the equations of motion. Turbulence is a 3-D phenomenon and there are no satisfactory 2-D approximation for determining fine details of turbulent flows; all turbulent flows are inherently dissipative and turbulence observe a cascade process whereby its kinetic transfer from larger eddies to smaller eddies and the latter dissipate into heat due to molecular viscosity. Turbulence is a continuum phenomenon governed by the equations of fluid mechanics. Even the smallest scales in any turbulent flow are much larger than any molecular length scale. Finally, turbulence is a flow feature, and not a fluid feature. It is possible to estimate the magnitude of the smallest scale through dimensional analysis. As stated above the cascade process involves a transfer of turbulent kinetic energy k (associated to fluctuating turbulent velocity) from larger eddies to smaller ones. The smaller eddies should be in a state where the rate of receiving energy from larger eddies is very nearly equal to the rate at which the smallest eddies dissipate the energy to heat [30]. Hence the motion at the smallest scales should depend only upon the rate at which the larger eddies supply energy, $\epsilon = -\frac{dk}{dt}$ and the kineamtic viscosity ν. Having established appropriate dimensional quantities for ϵ and ν one can derive the *Kolmogorov* scales of length, time and velocity

$$\eta = \left(\frac{\nu^3}{\epsilon}\right)^{\frac{1}{4}}, \qquad \tau = \left(\frac{\nu}{\epsilon}\right)^{\frac{1}{2}}, \qquad \upsilon = (\nu\epsilon)^{\frac{1}{4}}. \tag{3.31}$$

With dimensional analysis the dissipation rate ϵ could be related with k through:

$$\epsilon \sim \frac{k^{\frac{3}{2}}}{l}, \tag{3.32}$$

where l is the integral length scale of the largest eddies. Hence the ratio

$$\frac{l}{\eta} \sim Re_t^{\frac{3}{4}}, \tag{3.33}$$

with Re_t being the turbulence Reynolds number based on l and k. Thus, the energy cascade involves a number of scales proportional to N:

$$N = Re_t^{\frac{9}{4}}. \tag{3.34}$$

It is now clear that in order to ensure that all the features of turbulence are predicted correctly, a large computational domain and a very dense grid are requested.

This is a DNS (direct numerical simulation) approach and it is not affordable for industrial applications because of the need to obtain results within a reasonable time and because of the great request of computational resources. Hence a mathematical model is required to predict turbulent flow properties but modelling turbulence involves statistical studies of the equations of fluid flow and always leads to the closure problem: more unknowns than equations. In order to make the number of equations equal to the number of unknowns, assumptions are imperative. Usually there are two approaches: filtering in space or averaging in time. The first approach called LES (large eddy simulation) consists on applying a spatial filter to Navier Stokes equations with only the length scales smaller than the size of the filter modelled. Nevertheless, nowadays time averaging is still the most common turbulence model approach in industrial applications and all turbulent fluctuations need to be modelled. The key approach is to decompose the flow variables into a time-mean value component and a fluctuating one, substituting in the original equations, and time-averaging the obtained equations. Expressing the instantaneous velocity as the sum of a mean and a fluctuating part so that:

$$\boldsymbol{v}(\mathbf{x},t) = \overline{\boldsymbol{v}}(\boldsymbol{x}) + \boldsymbol{v'}(\boldsymbol{x},t). \tag{3.35}$$

The time-averaging properties lead to the following expression for the incompressible $RANS$ continuity, momentum, energy equations:

$$\boldsymbol{\nabla} \cdot (\rho\overline{\boldsymbol{v}}) = 0, \tag{3.36}$$

$$\frac{\partial \rho\overline{\boldsymbol{v}}}{\partial t} + \boldsymbol{\nabla} \cdot \{\rho\overline{\boldsymbol{v}}\overline{\boldsymbol{v}}\} = -\nabla\overline{p} + \nabla \cdot \left(\overline{\boldsymbol{\tau}} - \rho\overline{\boldsymbol{v'}\boldsymbol{v'}}\right) + \rho\boldsymbol{g}, \tag{3.37}$$

$$\frac{\partial}{\partial t}(\rho c_p\overline{T}) + \nabla \cdot (\rho c_p\overline{\boldsymbol{v}}\overline{T}) = \nabla \cdot (k\nabla\overline{T} - \rho c_p\overline{\boldsymbol{v'}T'}) + \overline{S^T}. \tag{3.38}$$

Keeping the unsteady term $\dfrac{\partial \rho\overline{\boldsymbol{v}}}{\partial t}$ in the momentum equation usually brings to the definitions of $URANS$ (unsteady Reynolds averaged Navier-Stokes), but attention should be maintained for those turbulent flows where there is no clear distinction between timescale characteristic of slow variations of the mean flow and that related to turbulent fluctuations. Indeed, the approximation

$$\overline{\frac{\partial \boldsymbol{v}(\mathbf{x},\boldsymbol{t})}{\partial t}} \approx \frac{\partial \overline{\boldsymbol{v}}(\boldsymbol{x})}{\partial t}, \tag{3.39}$$

is true if $|\boldsymbol{v'}| << |\overline{\boldsymbol{v}}|$. This is always questionable, however using time averaging in this manner is usefull for analysis especially for time marching numerical methods implemented for solving fluid dynamics problems but a degree of caution must be

exercised when fluctuations are not too small. Comparing Eq.(3.37) with Eq.(3.7) and Eq.(3.29) with Eq.(3.38), one can note the appearance of new terms on the right-hand-side. These terms are called *Reynolds Stresses Tensor* and *turbulent heat fluxes*. So what Reynolds-averaging does is to introduce 9 new variables and to solve the *RANS* (Reynolds averaged Navier-Stokes) equations, but additional equations are required.

Here comes into play the *Boussinesq Hypothesis* which makes an analogy with Newtonian fluids by assuming that the *Reynolds stresses* are a linear function of the *mean velocity gradients*

$$-\rho\overline{\boldsymbol{v}'\boldsymbol{v}'} = \mu_T\left[\nabla\boldsymbol{v} + (\nabla\boldsymbol{v})^T\right] - \frac{2}{3}\rho k\boldsymbol{I}. \tag{3.40}$$

This assumption reduces the number of unknown from 6 to 2: the *turbulent eddy viscosity* μ_T and the *turbulent kinetic energy* k.

For incompressible flows, the equations can be rearranged by defining a *turbulent pressure* p [12]:

$$p \leftarrow p + \frac{2}{3}\rho k. \tag{3.41}$$

In this manner, the only unknown that remains to compute is the *turbulent eddy viscosity* μ_T. The great variety of turbulence models derive from different ways of evaluating μ_t. In a similar way, the turbulent thermal fluxes are calculated in analogy with Fourier's law such that

$$-\rho c_p\overline{\boldsymbol{v}'T'} = \alpha_t\nabla T, \tag{3.42}$$

where α_t is the turbulent thermal diffusivity.

In the following section are presented the principle and the capabilities of the *Shear Stress Transport (SST) k-ω model* that is the one employed in this work. However for a better understanding also the $k-\epsilon$ and the $k-\omega$ models are briefly explained.

3.3 Shear Stress Transport $k-\omega$ Model

The k-ω family of linear eddy viscosity models seems to be by far the most widely used ones. Before describing the *SST* $k-\omega$ model it is necessary to briefly introduce *standard* $k-\epsilon$ and $k-\omega$ models, since *SST* is a combination of these two approach. Both methods belong to the *two-equations* family of turbulence models. This class involves the resolution of two additional partial differential equations in order to locally compute the *turbulent eddy viscosity* μ_T and the *turbulent thermal diffusivity* α_t.

$k - \epsilon$

Like other models based on the *Boussinesq Hypothesis*, the $k - \epsilon$ model is based on the following expressions for *turbulent eddy viscosity* μ_t and for turbulent thermal diffusivity α_t:

$$\mu_t = \rho C_\mu \frac{k^2}{\epsilon}, \qquad \alpha_t = c_p \frac{\mu_t}{Pr}, \tag{3.43}$$

where C_μ is a calibration constant, k is the *turbulent kinetic energy* and ϵ is the *rate of dissipation of turbulent kinetic energy per unit mass due to viscous stresses.* Solving the following transport equations for k and ϵ a local value of μ_t can be computed:

$$\frac{\partial}{\partial t}(\rho k) + \nabla \cdot (\rho \boldsymbol{v} k) = \nabla \cdot ((\mu + \frac{\mu_T}{\sigma_k})\nabla k) + S_k, \tag{3.44}$$

$$\frac{\partial}{\partial t}(\rho \epsilon) + \nabla \cdot (\rho \boldsymbol{v} \epsilon) = \nabla \cdot ((\mu + \frac{\mu_T}{\sigma_\epsilon})\nabla \epsilon) + S_\epsilon, \tag{3.45}$$

It must be kept in mind that the construction of this model is based on two important assumptions:

- *fully turbulent flow*
- *negligible molecular viscosity effects*

that establish the limits of this approach:

- *validity only for high Reynolds*
- *inability to reach the wall*

To account for this lack the so-called *low Reynolds* $k - \epsilon$ model have been developed. These models use damping functions to damp the turbulent viscosity while getting close to the wall.

$k - \omega$

In this model the equation for ϵ is substituted by an equation for ω, where ω is called *specific turbulent dissipation* and represents the rate at which the turbulent kinetic energy is converted into thermal energy per unit time and unit volume.

$$\omega = \frac{\epsilon}{C_\mu k}. \tag{3.46}$$

The *turbulent eddy viscosity* and *turbulent thermal diffusivity* are then given by:

$$\mu_T = \rho \frac{k}{\omega}, \qquad k_T = \frac{\mu_T}{Pr_T}. \tag{3.47}$$

The two additional equations are:

$$\frac{\partial}{\partial t}(\rho k) + \nabla \cdot (\rho \boldsymbol{v} k) = \nabla \cdot ((\mu + \frac{\mu_T}{\sigma_k})\nabla k) + S_k, \tag{3.48}$$

$$\frac{\partial}{\partial t}(\rho \omega) + \nabla \cdot (\rho \boldsymbol{v} \omega) = \nabla \cdot ((\mu + \frac{\mu_T}{\sigma_\omega})\nabla \omega) + S_\omega. \tag{3.49}$$

This new equation has three advantages [12]:

- *it is easier to integrate*;
- *it is integrable also in the sub-layer without using damping functions*;
- *it is capable to deal with weak adverse pressure gradients.*

As pointed out by its inventor, the $k-\omega$ model is accurate for both free shear flows and wall-bounded (attached boundary layer and mildly separation) [30]. But, unfortunately, this model has a strong dependence on the free stream values.

Looking at capabilities and flaws of the two models, they seems to be "complementary". The $k-\epsilon$ model, thanks to its insensitivity to the free stream, predicts with more accuracy away from the wall, while the $k-\omega$ behaves better in the boundary layer and with weak adverse pressure gradients.

These considerations have led to the development of the *Baseline (BSL)* $k-\omega$ model, that uses a *blending function* to switch from the $k-\omega$ and a rearrangement of $k-\epsilon$ in terms of ω.

The *Shear Stress Transport* $k-\omega$ model represents a further improvement to the *Baseline*, by limiting the shear stress in adverse pressure gradient flows. Menter, the developer of these two methods, writes [23]: "*It (BSL) has a performance similar to the Wilcox model, but avoids that model's strong freestream sensitivity. The second model (SST) results from a modification to the definition of the eddy-viscosity in the BSL model, which accounts for the effect of the transport of the principal turbulent shear stress. The new model is called shear-stress transport-model and leads to major improvements in the prediction of adverse pressure gradient flows.*"

3.4 Near the wall treatment

On every solid surface, due to the fluid viscosity, a boundary layer develops. This layer of fluid can be divided in three regions:

- *viscous sub-layer* ($0 < y^+ < 5$), where the effect of viscosity dominates;
- *buffer sub-layer* ($5 < y^+ < 30$), where viscous and inertial effects are equal;
- *inertial (log-law) sub-layer* ($30 < y^+ < 500$), where the effect of inertia dominates.

These three sub-layers can be identified by the value of y^+ that is the adimensionalized normal distance ($d_\perp$) from the wall:

$$y^+ = \frac{d_\perp u_\tau}{\nu}, \tag{3.50}$$

where $u_\tau = \sqrt{\tau_w/\rho}$ is the velocity scale.

This subdivision of the boundary layer is schematized in *Figure 3.1*.

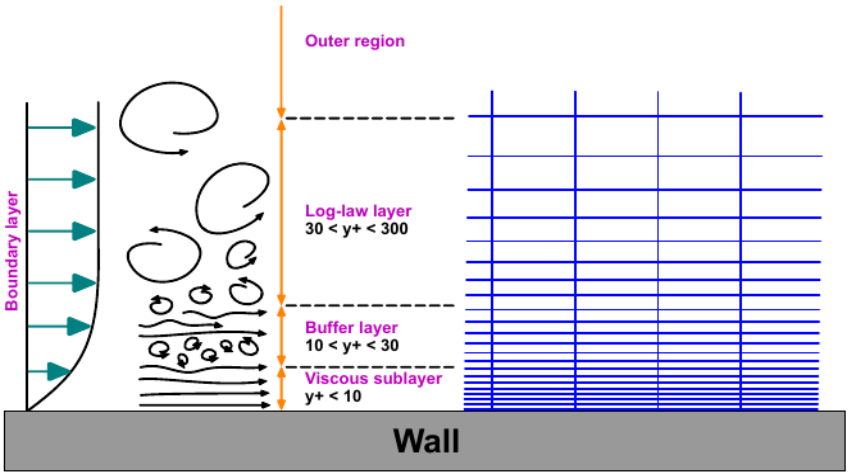


Figure 3.1: *Boundary layer subdivision and correspondent y^+ ranges. (courtesy of Wolf Dynamics srl)*

Turbulence models avoid the *buffer sub-layer*, because the high turbulent production, by placing the first cell center in the *viscous sub-layer* or in the *inertial sub-layer*.

The first option leads to accurate prediction of the boundary layer, but requires a very fine discretization near the wall, usually leading to unaffordable costs.

The second, combined by the definition an appropriate wall-value to each new variable introduced, significantly reduces computational costs while giving a good accuracy. This velocity profile is called *wall function* and its action is schematized in *Figure 3.2*.

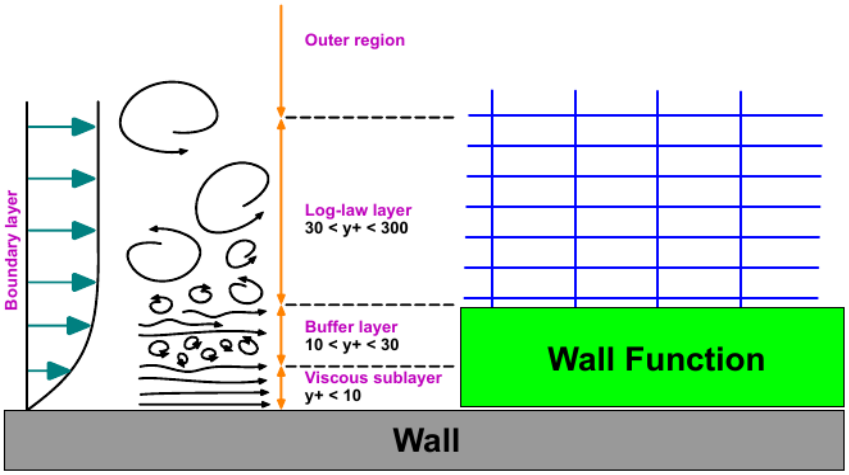


Figure 3.2: *Representation of the wall function approach. (courtesy of Wolf Dynamics srl)*

3.5 Flow with particles

To numerically model a multiphase flow, it is often important to use separate formulations for the different phases. The *particle phase* consists of bubble, particle, or drops and the *continuous phase* is the fluid in which these particles are generally immersed. The particle can be composed of solid, liquid, or gas, and the continuous fluid can be a liquid or a gas. The coupling between the particle motion and its surroundings can be used to classify the character of the multiphase flow, and thus help determine appropriate numerical techniques. The broadest division is between dispersed and dense flows, and refers to which coupling mechanism primarily determines the particle motion. A multiphase flow can be considered dispersed if the effect of particle–fluid interactions dominates the overall transport of the particles, while it is said to be dense if particle-particle interaction dominates particles motion. Dispersed flows includes one-way coupling (where the dispersed-phase motion is affected by the continuous phase, but not vice versa) and two-way coupling (where the dispersed phase also affects the continuous phase through the interphase coupling). Dense flows usually have four-way coupling where mutual interactions between particles become significant and the effects of the particles on the continuous fluid are weak and often neglected. As well described in [11] volume fraction of particles is the main parameter to make the division between one, two or four way coupling and therefore between dispersed and dense. The volume fraction is defined as $\Phi_p = \frac{MV_p}{V}$ where M is the number of particles, V_p is the volume of a single particle and V is the volume occupied by particles and fluid:

- for $\Phi_p < 10^{-6}$ there will be one-way coupling;
- for $10^{-6} < \Phi_p < 10^{-3}$ there will be two-way coupling and the particles can also increase or dump turbulence;
- for $\Phi_p > 10^{-3}$ there will be four-coupling and the flow will be considered as dense.

Another important parameter that may contribute to the selection of the appropriate model is the particle momentum *Stokes number* defined as the ratio between the particle response time τ_p and that of the system τ_s:

$$St = \frac{\tau_p}{\tau_s}, \tag{3.51}$$

$$\tau_p = \frac{\rho_p d^2}{18\rho_f \nu} \qquad (only\ for\ Stokes\ flows), \tag{3.52}$$

$$\tau_s = \frac{L_s}{v_s}. \tag{3.53}$$

If $St \to 0$, the particle behaves as a fluid tracer (momentum one-way coupling) and if $St \to \infty$ is unresponsive to the flow variations. One would define a Stokes number not only for momentum but also for mass and temperature in order to evaluate with more precision the mass coupling and energy coupling of particles with the continuous phase. If two-way coupling is considered, it simply involves some source terms in the continuous phase equations (momentum, energy, turbulence models...) that are generally described in an Eulerian reference frame. For the sake of simplicity, we have limited our interest to the one-way coupling.

3.5.1 Particle equations

Various treatments of the particle field can be employed. Particles could be described in an Eulerian or Lagrangian reference frame and, as suggested by E. Loth in his paper [18], distinctions could be done about the treatment of particle surface forces. In this work only the Lagrangian approach will be discussed. With this reference frame the particles are treated as individual and properties are updated along the path of each particle. For the treatment of the surface forces, the point-force treatment represents the flow over the particle with empirical and theoretical treatments (specifying a drag or lift coefficient) to obtain the force on the particle. For the resolved surface treatment, the fluid dynamics (e.g., pressure and shear stress distributions) are fully resolved over the entire particle surface and then integrated to obtain the overall hydrodynamic forces. Following the point-surface approach

and defining $\boldsymbol{x}_p$ as the particle centroid and m_p the particle mass, the Lagrangian particle equation of motion is:

$$m_p \frac{d\boldsymbol{v}}{dt} = \boldsymbol{F}_{body} + \boldsymbol{F}_{surf}. \tag{3.54}$$

The left hand side represents the particle mass inertia and the right hand side represents the sum of body forces and surface forces on the particle. Body forces are those related to gravitational effects:

$$\boldsymbol{F}_{body} = \frac{(\rho_p - \rho)\pi d_p^3}{6}\boldsymbol{g}. \tag{3.55}$$

Where d_p are ρ_p are respectively the droplet diameter and droplet density. Surface forces can be seen as the sum of different terms: drag, virtual mass, a term related to pressure gradient and one to the "history" of particle (Basset term). The expressions for all these terms are listed below without a rigorous derivation:

- The drag force is:

$$F_{D,i} = \frac{1}{2}\frac{\pi d_p^2}{4}\rho_f C_d |\boldsymbol{u} - \boldsymbol{u}_p|(u_i - u_{p,i}), \qquad C_d = \frac{24}{Re_p}(1 + \frac{3}{16}Re_p). \tag{3.56}$$

- The pressure gradient force is:

$$F_{P,i} = \frac{1}{6}\pi d_p^3 \rho_f \frac{Du_i}{Dt}. \tag{3.57}$$

- The added mass force (virtual force) is:

$$F_{A,i} = \frac{\pi d_p^3}{12}\rho_f \Big(\frac{Du_i}{Dt} - \frac{du_{p,i}}{dt}\Big). \tag{3.58}$$

- The Basset force is:

$$F_{B,i} = \frac{3}{2}d_p^2 \rho_f \sqrt{\pi\nu} \int_{-\infty}^{t} \frac{d}{d\tau}(u_i - u_{p,i})\frac{d\tau}{\sqrt{t-\tau}}, \tag{3.59}$$

where Re_p is the particle Reynolds number based on relative velocity:

$$Re_p = \frac{\rho d_p(\boldsymbol{u} - \boldsymbol{u}_p)}{\mu}, \tag{3.60}$$

where ρ is the density of the continuous phase. A better description of all these terms can be found in [18] and in [6].

3.5.2 Kinetic of drying process

The dispersed phase can exchange not only momentum but also mass and heat with the continuous phase. Such phenomena are discussed below in a complete description of the entire drying kinetics for a single droplet. Before introducing the model, a brief introduction to mechanism of mass and heat transfer involved in that process is provided.

3.5.3 Mass transfer: diffusion and convection

Fick's law of diffusion

Diffusion is the process by which molecules, ions, or other small particles spontaneously mix, moving from regions of relatively high concentration into regions of lower concentration. This process can be analyzed through the Fick's law of diffusion that states that the rate of diffusion of a chemical species at a location in a gas mixture (or liquid or solution) is proportional to the concentration gradient of that species at that location. The following notation is coherent with [4] where the concentration of a species can be expressed in a mass basis or mole basis way. On a mass basis, concentration is expressed in terms of density ρ or in dimesionless form in terms of mass fraction w:

$$\rho_i = \frac{m_i}{V}, \tag{3.61}$$

$$\rho = \sum \rho_i, \tag{3.62}$$

$$w_i = \frac{\rho_i}{\rho}. \tag{3.63}$$

On a mole basis, concentration is expressed in terms of molar concentration C or in dimesionless form in terms of mole fraction y:

$$C_i = \frac{N_i}{V}, \tag{3.64}$$

$$C = \sum C_i, \tag{3.65}$$

$$y_i = \frac{C_i}{C}. \tag{3.66}$$

The mass m and the mole number N are related by $m = NM$ where M is the molar mass. Therefore for the ith species i:

$$C_i = \frac{\rho_i}{M_i}, \tag{3.67}$$

$$w_i = y_i \frac{M_i}{M}. \tag{3.68}$$

Following the Dalton's law of pressure according to which the total pressure of a gas mixture P is equal to the sum of the partial pressures P_i and reminding that for ideal gas $PV = NR_uT$, the mole fraction y_i may be written as:

$$\frac{P_i}{P} = \frac{N_i R_u T/V}{N R_u T/V} = y_i. \tag{3.69}$$

The linear relationship between the rate of diffusion and the concentration gradient was proposed by Fick in 1855. Considering a *stationary* binary mixture composed by specie A and B the diffusive mass flux of species A, j_A, in the specified direction x is given by:

$$j_A = -\rho D_{AB} \frac{d(\rho_A/\rho)}{dx} = -\rho D_{AB} \frac{dw_A}{dx}, \tag{3.70}$$

$$\bar{j}_A = -C D_{AB} \frac{d(C_A/C)}{dx} = -C D_{AB} \frac{dy_A}{dx}. \tag{3.71}$$

If $\rho = \rho_A + \rho_B$ and $C = C_A + C_B$ is constant through the mixture:

$$j_A = -D_{AB} \frac{d\rho_A}{dx}, \tag{3.72}$$

$$\bar{j}_A = -D_{AB} \frac{dC_A}{dx}. \tag{3.73}$$

$$\tag{3.74}$$

For two three-dimensional cases, Fick's law can conveniently be expressed in vector form:

$$\boldsymbol{j}_A = -\rho D_{AB} \nabla w_A, \tag{3.75}$$

$$\bar{\boldsymbol{j}}_A = -C D_{AB} \nabla y_A, \tag{3.76}$$

where D_{AB} is the diffusion coefficient usually determined experimentally.

Convection

Mass transfer problems usually involve diffusion in a moving medium, therefore species are transported both by molecular diffusion and by the bulk motion of the medium; that means by convection. Usually when dealing with this kind of problems it is common to refer to some experimental correlations in such a way very similar to the well known heat transfer convective correlations. Also, mass convection is usually analyzed on a mass basis approach and for the sake of simplicity the attention will be focused on fluids that are or can be treated as binary mixtures. The *Schimdt* number is a dimesionless number that expresses the ratio between the momentum

diffusivity and the mass diffusivity, hence it is a usefull parameter to compare the velocity boundary layer and concentration boundary layer. The Shmidt number is written

$$Sc = \frac{\nu}{D_{AB}}. \tag{3.77}$$

It is important to stress that the real physical mechanism of mass transfer is controlled by Fick's law of diffusion because of the no slip boundary condition for the bulk flow. However it is common to define a convective mass transfer coefficient h_{mass} in order to express the rate of mass convection as follow:

$$\dot{m}_{conv} = h_{mass} A(\rho_{As} - \rho_{A\infty}). \tag{3.78}$$

Where ρ_{As}, $\rho_{A\infty}$ are respectively the density of species A on the surface and out of the concentration boundary layer. The h_{mass} coefficient is expressed in terms of correlations through the *Sherwood* number, a dimensionless parameter that represents the effectiveness of mass convection at the surface. Defining a characteristic length L_c the Sherwood number is:

$$Sh = \frac{h_{mass} L_c}{D_{AB}}. \tag{3.79}$$

For a given geometry and for a flow type, Sh is a function of the Reynolds number and Schimdt number Eq.(3.77). A well known example is given by the Ranz-Marshall correlation for mass transfer:

$$Sh = 2 + 0.6 Re^{\frac{1}{2}} Sc^{\frac{1}{3}}, \qquad 0 \leqslant Re < 200. \tag{3.80}$$

The use of Eq.(3.78) is valid only for low mass flux because of the no slip boundary condition. This condition is still verified only if the rate of mass transfer of a species is small relative to the flow rate of that species. However it is possible to use with good approximation Eq.(3.78) for evaporation of water into air unless the water temperature reaches the saturation temperature for the external pressure condition. This for examples implies that Eq.3.78 can't be used for evaporation of droplets in combustion chambers or generally speaking to mass transfer in boilers and condensers.

3.5.4 Heat transfer: diffusion and convection

Conduction

Heat transfer due to conduction takes place in solids and quiescent fluids. The heat is transferred by diffusion and collisions between particles, without any mass

flow [25]. Heat transfer conduction is controlled by Fourier's law (see Eq.(3.25)) that states the relationship between the heat flow and the temperature gradient through the constant of proportionality k_c that is the thermal conductivity that in general varies with temperature. The 1D version of Eq.(3.30) without heat generation is:

$$\rho c_p \frac{\partial T}{\partial t} = \frac{\partial}{\partial x}\Big(k_c \frac{\partial T}{\partial x}\Big). \tag{3.81}$$

If k_c is constant with temperature then Eq.(3.81) reduces to:

$$\frac{1}{\alpha}\frac{\partial T}{\partial t} = \frac{\partial^2 T}{\partial x^2}, \tag{3.82}$$

where $\alpha = \frac{k_c}{\rho c_p}$ is the thermal diffusivity and it's a measure of how much heat is conducted with respect to the heat stored within the body. For transient heat conduction problems, α, together with a reference length L_c, is used to make the distinction between "early" regime and "late" regime. Denoting with t the time associated with the process involved, and with Fo the respective dimensionless time it is possible to say:

$$Fo = \frac{\alpha t}{L_c^2} << 1 \qquad "Early\ regime", \tag{3.83}$$

$$Fo = \frac{\alpha t}{L_c^2} \sim 1 \qquad "Transition", \tag{3.84}$$

$$Fo = \frac{\alpha t}{L_c^2} >> 1 \qquad "Late\ regime". \tag{3.85}$$

For the "late regime" the lumped approximation is suitable if the dimensionless Biot number $Bi = \frac{hL_c}{k_c} \leqslant 0.1$, therefore it is possible to assume a uniform temperature distribution throughout the body. Hence Eq.(3.82) can be simplified:

$$mc_p \frac{dT}{dt} = hA(T_\infty - T), \tag{3.86}$$

where h is the convective heat transfer coefficient described in the next section. The Biot number could be seen as the ratio between the heat convected to the body and the heat conducted within the body. The smaller the Biot number the more accurate the lumped approximation. For the "early regime" and "transition" regime Eq.(3.82) should be solved with appropriate boundary conditions; analytical solutions exist also in cylindrical and spherical problems, but they involve infinite series which are difficult to deal with. For $Fo > 0.2$ the one term approximation leads to small errors and solutions are also available in graphical form (Heisler's chart [4]).

Convection

Heat transfer with the presence of bulk fluid motion is usually called convection, and the analogy with convection mass transfer will be clear at the end of this section. The bulk fluid motion increases heat transfer since it brings hotter and cooler fluid layers into contact and the higher the fluid velocity, the higher the rate of heat transfer. Usually the rate of convection heat transfer is expressed through the Newton's law:

$$\dot{q}_{conv} = hA(T_s - T_\infty), \tag{3.87}$$

where h is the convection heat transfer coefficient. Because of the no slip boundary condition for fluid flow, heat is always transferred by conduction in the fluid layer near the surface, and then convected away because of the fluid motion. Therefore a rigorous definition of h is:

$$h = \frac{-k_{fluid}\frac{\partial T}{\partial y}|_{y=0}}{T_s - T_\infty}. \tag{3.88}$$

The complete energy equation was presented in section 3.1.3, the resolution of which could bring the information about temperature distribution. If the temperature distribution is unknown, h is determined through correlations very similar to those described in mass convection section depending on the nature of fluid flow motion (laminar, turbulent, external, internal, forced or natural convection). Correlations usually involve adimensional numbers, such as the Prandtl's number $Pr = \frac{\nu}{\alpha}$ and Nusselt's number $Nu = \frac{hL_c}{k_{fluid}}$. Prandtl's number is the ratio between momentum diffusivity and thermal diffusivity and provides information about the thickness of velocity and thermal boundary layers. The Nusselt number represents the enhancement of heat transfer through a fluid layer as a result of convection relative to conduction across the same fluid layer. The larger the Nusselt number, the more effective the convection. Nu is a function of Re and Pr, for example the Ranz-Marshall correlation for convective heat transfer is:

$$Nu = 2 + 0.6Re^{\frac{1}{2}}Pr^{\frac{1}{3}}. \tag{3.89}$$

3.5.5 Two-stage evaporation model

The overall drying process of droplets inside the spray dryer can be divided in two stages. During the first stage droplets containing solids and great amount of liquid enters the drying volume, gets sensible heat and evaporation occurs on the surface resulting in droplet diameter shrinking. During this first stage the liquid excess envelops the entire droplet volume and evaporation is very similar to the

evaporation of pure water droplets [22]. When the droplet moisture content reaches a critical value, according to the model adopted, a solid crust surrounding a wet core is formed. At this point the droplet diameter is considered to be constant and only the wet core shrinks until the droplet reaches the final moisture content. During the second drying stage water vapour diffuses through the solid crust, and the rate at which it reaches the droplet surface depends on the crust porosity. The entire drying process is summarized in Fig.(3.3), where all features previously introduced are stressed.

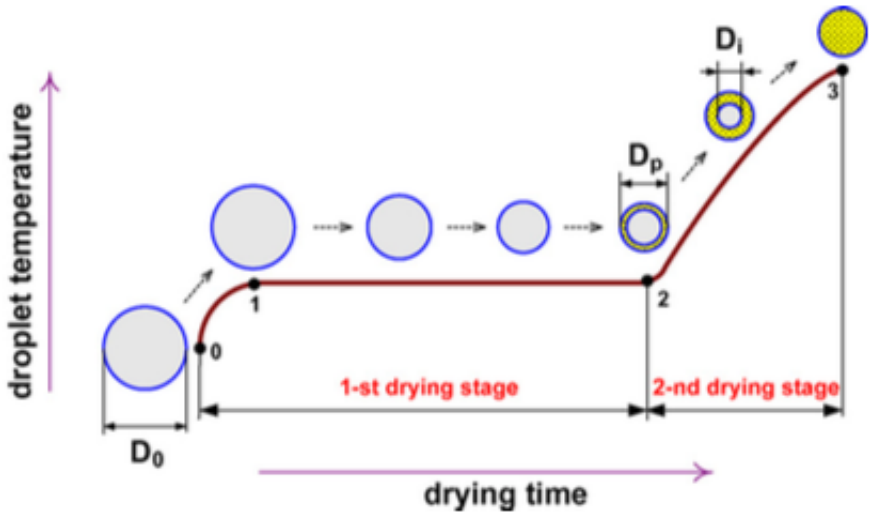


Figure 3.3: *Two stage model of droplet drying [21].*

3.5.6 First stage

Figure (3.3) shows that the first stage of evaporation can be considered as the sum of two additional steps:

- Droplet initial heating, where the droplet temperature rises, the rate of evaporation is very low, and the droplet radius is approximately constant;
- Constant temperature evaporation period, where the rate of evaporation (and therefore the rate at which heat is lost due to vaporization enthalpy) is enough to balance the heat transfer rate from the surrounding hot air;

If both conditions for a lumped approximation are satisfied, (see section 3.5.4) that means both $Fo >> 1$ and $Biot < 0.1$, the equation of energy conservation for the droplet is:

$$h_{fg}\dot{m}_v + c_{p,d}m_d\frac{dT_d}{dt} = h(T_{air} - T_d)4\pi R_d^2, \tag{3.90}$$

where h_{fg} is the specific heat of evaporation, $\dot{m}_v$ the vapour mass transfer rate, m_d,$c_{p,d}$, T_d and R_d corresponding to mass, specific heat, temperature and radius of the dried droplet and h is the heat transfer coefficient [20]. The rate of moisture evaporation is controlled by Eq.(3.78) here specified for the specific case of single droplet:

$$\dot{m}_v = -\frac{dm_d}{dt} = 4\pi R_d^2 h_{mass}(\rho_{v,s} - \rho_{v_\infty}), \tag{3.91}$$

where $\rho_{v,s}$ is the saturation density of water at droplet surface and it is a function of $T_{d,s}$. ρ_{v_∞} is the vapour density in the surrounding air depending on its relative humidity and h_{mass} is the convective mass transfer coefficient. According to [8], the water vapour densities of the droplet and the gas are given by the following expressions:

$$\rho_{v,s} = \frac{M_w p_{sat}}{R_u T_{d,s}}, \tag{3.92}$$

$$\rho_{v_\infty} = \frac{M_w p_{v,air}}{R_u T_{air}}, \tag{3.93}$$

where M_w is the water molecular weight, P_{sat} is the saturation pressure of water at droplet surface temperature, $P_{v,air}$ is the vapour partial pressure in the surrounding air and R_u is the universal gas constant. The convective mass transfer coefficient is evaluated through the modified Ranz-Marshall correlation for spherical evaporating droplets:

$$Sh = \frac{d_d h_{mass}}{D_v} = (2 + 0.6 Re^{\frac{1}{2}} Sc^{\frac{1}{3}})(1 + B)^{-0.7}. \tag{3.94}$$

The factor $(1 + B)^{-0.7}$ takes into consideration Stefan flow in the droplet boundary layer and $B = c_{p,v}\frac{(T_g - T_d)}{h_{fg}}$ is the Spalding number [21]. The diffusion coefficient of vapour in air in atmospheric conditions is evaluated as follows:

$$D_v = 3.564\ 10^{-10}(T_{d,s} + T_g)^{1.75}. \tag{3.95}$$

For the convective heat transfer coefficient in Eq.(3.90) a correlation very similar to Eq.(3.94) is used:

$$Nu = \frac{d_d h}{k_{air}} = (2 + 0.6 Re^{\frac{1}{2}} Pr^{\frac{1}{3}})(1 + B)^{-0.7}, \tag{3.96}$$

where k_{air} is the air thermal conductivity. The expression for droplet specific heat $c_{p,d}$ in Eq.(3.90) takes into account the properties of the water and the solid fraction:

$$c_{p,d} = (1 - c)c_{p,w} + cc_{p,s}, \tag{3.97}$$

Where c is the mass concentration of solid that is connected to the droplet moisture content, by:

$$x = \frac{m_w}{m_s} = \frac{m_d}{m_{d,0}}(1 + x_0) - 1, \tag{3.98}$$

$$c = \frac{1}{1 + x}, \tag{3.99}$$

where x_0 and $m_{d,0}$ are the droplet initial moisture content and initial mass. The droplet diameter is shrinking due to evaporation and, for the mass conservation, it can be computed from:

$$\frac{dR_d}{dt} = -\frac{\dot{m}_v}{\rho_w 4\pi R_d^2}. \tag{3.100}$$

Integrating Eq.(3.100) it is possible to follow also the evolution of the droplet mass:

$$m_d = m_{d,0} - \pi\rho_w \frac{8}{6}(R_{d,0}^3 - R_d^3). \tag{3.101}$$

Usually a lumped approximation is acceptable for small droplets since $Fo \sim \frac{1}{R_d^2}$ and $Biot \sim R_d$. However, if the droplets radius isn't small enough, the Fo number at the end of the initial heating period does not satisfy the lumped condition, therefore the transient is too fast and the effect of temperature rise do not have interested all the characteristic length of the droplet (radius), and the temperature profile within the droplet may be considered. In this case the equation of energy conservation for the initial heating is:

$$\rho c_{p,d} \frac{\partial T_d(r,t)}{\partial t} = \frac{1}{r^2}\frac{\partial}{\partial r}\Big(k_d r^2 \frac{\partial T_d}{\partial r}\Big), \tag{3.102}$$

and the corresponding boundary conditions are:

$$\begin{cases} \frac{\partial T_d}{\partial r} = 0 \quad for \quad r = 0, \\ h(T_g - T_d) = k_d \frac{\partial T_d}{\partial r} \quad for \quad r = R_d. \end{cases}$$

At the end of the initial heating stage, when evaporation is considerable and the droplet radius starts to shrink significantly, the droplet Fourier and Biot numbers could rapidly exceed the lumped approximation limits and, for the subsequent evaporation stage at constant temperature, both radial and temporal variations of the droplet can be neglected. From this point on this approach will be called "Uniform temperture approach". This results in the following equation of energy conservation:

$$h_{fg}\dot{m}_v = h(T_g - T_d)A_d. \tag{3.103}$$

Combining Eq.(3.103) with Eq.(3.78) and introducing Eq.(3.96), Eq.(3.94) and Eq.(3.92) the following equation is obtained:

$$\frac{T_g - T_d}{h_{fg}} = \frac{2 + 0.6 Re_d^{\frac{1}{2}} Sc^{\frac{1}{3}}}{2 + 0.6 Re_d^{\frac{1}{2}} Pr^{\frac{1}{3}}} \frac{D_v M_w}{k_{air} R_u} \left(\frac{p_{v,s}}{T_d} - \frac{p_{v,\infty}}{T_{air}} \right). \tag{3.104}$$

Solving for T_d the equilibrium evaporation temperature is obtained and droplet properties could be tracked with Eq.(3.98), Eq.(3.100) and with Eq.(3.101). In the case when lumped conditions are not satisfied, the fully transient approach is needed and the energy conservation equation is applied to a time dependent spatial domain:

$$\rho c_{p,d} \frac{\partial T_d(r,t)}{\partial t} = \frac{1}{r^2} \frac{\partial}{\partial r} \left(k_d r^2 \frac{\partial T_d(r,t)}{\partial r} \right), \tag{3.105}$$

and the corresponding boundary conditions are:

$$\begin{cases} \frac{\partial T_d}{\partial r} = 0 \quad for \quad r = 0, \\ h(T_g - T_d) = k_d \frac{\partial T_d}{\partial r} + h_{fg} \frac{\dot{m}_v}{A_d} \quad for \quad r = R_d. \end{cases}$$

3.5.7 Second stage

When the droplet moisture content falls below a certain critical value, a solid crust starts to develop on the entire droplet surface. A wet core still exists but, from now on, vapour diffuses through the crust that is considered to be porous [24]. For this stage the fully transient approach is considered and both temperature variation with time and radius are evaluated.

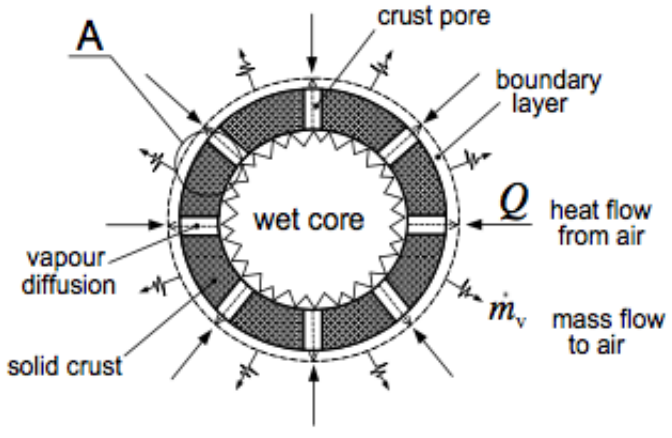


Figure 3.4: *Details of the second stage.*

During the second stage of evaporation the external diameter is constant while the wet core shrinks because of evaporation and, as a result, the crust thickness

increases. This problem is classified as a problem with internal moving evaporating interface [24]. Therefore two energy equation are required: one for the crust and one for the wet core. With respect to the crust region, the assumption of temperature independent crust thermal conductivity leads to:

$$\frac{\partial T_{cr}(r,t)}{\partial t} = \frac{\alpha_{cr}}{r^2}\frac{\partial}{\partial r}\Big(r^2\frac{\partial T_{cr}(r,t)}{\partial r}\Big) \quad R_i(t) \leq r \leq R_p, \tag{3.106}$$

and the corresponding boundary conditions are:

$$\begin{cases} k_{cr}\frac{\partial T_{cr}}{\partial r} = k_{wc}\frac{\partial T_{wc}}{\partial r} + h_{fg}\frac{\dot{m}_v}{A_i} \quad for \quad r = R_i(t), \\ T_{wc} = T_{cr} \quad for \quad r = R_i(t), \\ h(T_g - T_{cr}) = k_{cr}\frac{\partial T_{cr}}{\partial r} \quad for \quad r = R_P, \end{cases}$$

where R_P is the fixed external radius, and h is the convective heat transfer coefficient evaluated with Eq.(3.96). The wet core the energy conservation equation is:

$$\rho_{wc}c_{p,wc}\frac{\partial T_{wc}(r,t)}{\partial t} = \frac{1}{r^2}\frac{\partial}{\partial r}\Big(k_{wc}r^2\frac{\partial T_{wc}(r,t)}{\partial r}\Big) \quad 0 \leq r \leq R_i(t). \tag{3.107}$$

The corresponding boundary conditions are:

$$\begin{cases} \frac{\partial T_{wc}}{\partial r} = 0 \quad for \quad r = 0, \\ k_{cr}\frac{\partial T_{cr}}{\partial r} = k_{wc}\frac{\partial T_{wc}}{\partial r} + h_{fg}\frac{\dot{m}_v}{A_i} \quad for \quad r = R_i(t), \\ T_{cr} = T_{wc} \quad for \quad r = R_i(t). \end{cases}$$

The rate of interface receding is given by [8]:

$$\frac{dR_i}{dt} = -\frac{\dot{m}_v}{\varepsilon\rho_w 4\pi R_i^2}, \tag{3.108}$$

where ε is the crust porosity. The mass transfer rate from the spherical wet core can be evaluated through the Stefan's flow approximation [8] which leads to:

$$\dot{m}_v = -\frac{8\pi\varepsilon D_v M_w p_g}{R(T_{cr,s}+T_{wc,s})}\frac{R_P R_i}{R_P - R_i}\ln\left(\frac{p_g - p_{v,i}}{p_g - T_{cr,s}\big(\frac{R_u\dot{m}_v}{4\pi M_w h_{mass}R_P^2} + \frac{p_{v,\infty}}{T_g}\big)}\right). \tag{3.109}$$

The diffusion coefficient D_v is evaluated with Eq.(3.94). The particle moisture content and mass are given by:

$$x = m_p\frac{1+x_0}{m_{d,0}} - 1, \tag{3.110}$$

$$m_p = \frac{m_{d,0}}{1+x_0}\big(1 - \frac{\rho_w}{\rho_{solid}}\big) + \frac{4}{3}\pi\rho_w(\varepsilon R_i^3 + (1-\varepsilon)R_P^3). \tag{3.111}$$

According to the lumped approximation described for the first drying stage, also here it is possible to track the droplet temperature with a simpler approach. When the critical moisture content is reached, the wet particle turns into a non evaporating dry particle. This nonevaporating particle and the drying gas continue their interaction by convective heat transfer until thermal equilibrium. The particle temperature is determined from the following heat balance equation:

$$m_p c_p \frac{\partial T_p}{\partial t} = 4\pi R_p^2 h (T_g - T_p). \tag{3.112}$$

Because there is no more evaporation, the particle mass during this period remains invariable as well as its radius and its moisture content. This is a great simplification, the wet core is neglected and the crust does not grow, but this can provide some preliminary informations about the wet particle temperature rise.

Part III

Open Source Tools

Chapter 4

OpenFOAM

4.1 About OpenFOAM

OpenFOAM® (*Open source Field Operation And Manipulation*) is an open source finite volume software for computational fluid dynamics (CFD), owned by the OpenFOAM®Foundation and distributed exclusively under the GNU General Public Licence (GPL)[1]. That means it is freely available and according to the GNU general public license principles the users can modify and share the source code that is freely distributed. Generally speaking OpenFOAM® is a $C++$ library, used to solve partial differential equations (PDEs), and ordinary differential equations (ODEs). Its primary usage is to create executables, known as applications that fall into two categories: solvers, that are each designed to solve a specific problem in continuum mechanics, and utilities, that are designed to perform tasks that involve data manipulation[1]. The OpenFOAM® distribution has an extensive range of features to solve anything from complex fluid flows involving combustion and chemical reactions, multiphase flows and mass transfer, turbulence and heat transfer, particle methods (DEM,DSMC,MD) and lagrangian particles tracking to acoustics, solid mechanics and electromagnetics. It includes tools for meshing in and around complex geometries, and for data processing and visualisation, and more. Almost all computations can be executed in parallel as standard to take full advantage of today's multi-core processors and multi-processor computers. OpenFOAM® is supplied with pre- and post-processing environments. The interface to the pre- and post-processing are themselves OpenFOAM utilities, thereby ensuring consistent data handling across all environments. [1]. The overall structure of OpenFOAM® is shown in *Figure 4.1.*

Hence it is clear that OpenFOAM® capabilities mirror those of commercial

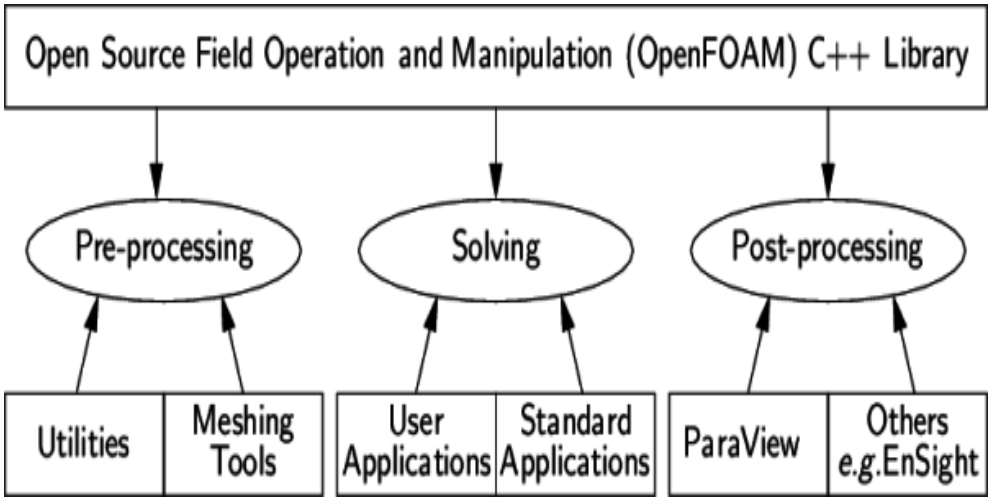


Figure 4.1: *Overview of OpenFOAM structure [1].*

CFD applications, but there are still some disadvantages compared to them, such as the lack of a native GUI, not much available documentation and, in wider terms, it is less user friendly; however, as the users have complete access to the source code, they have total freedom to modify existing solvers or use them as the starting point for new ones with some pre-requisite knowledge of the underlying method, physics and programming techniques involved. Summing up some of the features of OpenFOAM® are listed below taken from the official web-site [1]:

- Fluid Dynamics & Physical Modelling
  - Turbulence modelling (Reynolds-Averaged (RANS), Large-Eddy Simulation (LES), Detached-Eddy Simulation (DES,DDES,etc)
  - Thermophysical modelling
  - Transport/rheology
  - Multiphase flows
  - Rotating flows with multiple reference frames (MRF)
  - Rotating flows with arbitrary mesh interface (AMI)
  - Dynamic meshes
  - Compressible/thermal flows
  - Conjugate heat transfer
  - Porous media
  - Lagrangian particle tracking

  - Reaction kinetics/chemistry

- Geometry & Meshing
  - Mesh generation for complex geometries with *snappyHexMesh*
  - Mesh generation for simple geometries with *blockMesh*
  - Mesh conversion tools
  - Mesh manipulation tools

- Numerical Solution
  - Numerical method
  - Linear system solvers
  - Ordinary Differential Equation system solvers

- Computing & Programming
  - Equation syntax
  - Libraries of functionality
  - Parallel computing

- Data Analysis
  - *ParaView* post-processing
  - Post-processing command line interface (CLI)
  - Graphs and data monitoring

4.2 OpenFOAM case structure

The basic directory structure for a OpenFOAM® case, with the minimum set of files to run an application, is presented in *Figure 4.2*.

The roles of the main directories, contained in the case folder, are listed below:

- `system`, it contains the dictionaries to set up the entire solution procedure (from meshing to solving); at least it must contain three files:
  - `fvSchemes` to specify (run-time) the numerical schemes to discretize the equations;

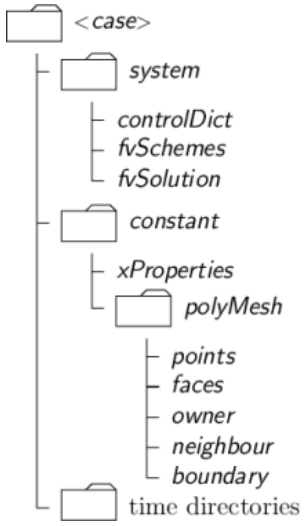


Figure 4.2: *Structure of an* OpenFOAM® *case [1].*

  - `fvSolution` to set equation solvers, tolerances and other algorithm controls;
  - `controlDict` to control (run-time) the simulation run (start/end time, time-step, function objects etc.)

- `constant`, it contains a folder (`polyMesh`) with the full description of the case mesh and files that specify the physical properties involved (transport and turbulence properties, gravity, dynamic properties etc.)

- `time directories`, it contains files that represent the specific fields at initial condition (e.g. `0` folder) or computed by OpenFOAM® (e.g. `0.01`, `0.02`, ... folders[1]) at consecutive times; it must be underlined that OpenFOAM® always require fields to be initialized, even in steady-state problems

A lot of pages should be written to exhaustively explain OpenFOAM®, but that is beyond the scope of this thesis. For further details the *CFD direct* website [1] is suggested.

[1]The name of the folder corresponds to the simulated time at which data are written.

4.3 The programming language of OpenFoam

4.3.1 Why *C++*

C++ is an *Object-Oriented Programming* language that attemps to provide techniques for managing enormous complexity, achieving the aim of the reuse of software components. As an *Object-Oriented Programming* languages is based on three pillars of the object-oriented development [17]:

- encapsulation
- inheritance
- polymorphism.

Encapsulation

C++ supports the properties of encapsulation through the creation of user-defined types, called classes. Once created a well defined class acts as a fully encapsulated entity and it is used as a whole unit. The actual inner workings of the class should be hidden. Users of a well defined class do not need to know how the class works; they just need to know how to use it.

Inheritance

C++ supports inheritance; a new type (class), which is an extension of an existing type, can be declared. This new subclass is said to derive from the existing type (sometimes is called a derived type) and inherits all its qualities, but the user can add new ones as needed.

Polymorphism

C++ supports the idea that different objects (belonging to the same class) do "the right thing" when the user chooses one of them. Being more exhaustive, in n programming languages, polymorphism means that some code or operations or objects behave differently in different contexts.

A clarifier example inherent to a CFD code is relative to a velocity field. The expression encapsulates the idea of movement with direction and magnitude and relates to other physical properties. In mathematics, we can represent a velocity field by a single symbol, e.g. U, and express certain concepts using symbols, e.g.

"the field of velocity magnitude" by $|U|$. The advantage of mathematics over verbal language is its greater efficiency, making it possible to express complex concepts with extreme clarity. The problems that we wish to solve in continuum mechanics are not presented in terms of intrinsic entities, or types, known to a computer, e.g. bits, bytes, integers. They are usually presented first in verbal language, then as partial differential equations in 3 dimensions of space and time. The equations contain the following concepts: scalars, vectors, tensors, and fields thereof; tensor algebra; tensor calculus; dimensional units. The solution to these equations involves discretisation procedures, matrices, solvers, and solution algorithms. Programming languages that are *Object-Oriented*, as stated in the introduction to this chapter, provide the mechanism to declare types -*classes*- and associated operations that are part of the verbal and mathematical languages used in science and engineering. The velocity field introduced earlier can be represented in programming code by the symbol U and "the field of velocity magnitude" can be $mag(U)$. The velocity is a vector field for which there should exist, in an *Object-Oriented*, a *vectorField* class. The velocity field U would then be an instance, or object, of the *vectorField class*; The clarity of having objects in programming that represent physical objects and abstract entities should not be underestimated. The class structure concentrates code development to contained regions of the code, the classes themselves, thereby making the code easier to manage. New classes can be derived or inherit properties from other classes, e.g. the *vectorField* can be derived from a *vector class* and a *Field class*. *C++* provides the mechanism of template classes such that the template class *Field*<*Type*> can represent a field of any <*Type*>, e.g. scalar, vector, tensor. The general features of the template class are passed on to any class created from the template. Templating and inheritance reduce duplication of code and create class hierarchies that impose an overall structure on the code [5].

Part IV

Numerical discretization

Chapter 5

Numerical discretization

5.1 First drying stage

In section 3.5.6 three different models for the first drying stage of water droplets with insoluble solids have been presented, which differ from one another depending on if the lumped conditions are satisfied. In this section the discretization of all equations for the three different approaches is described. Table 5.1 outlines what models will be treated within this section (IH denotes the initial heating period):

Table 5.1: *First stage models.*

Model	**T=f(r)**	**T=f(t)**	**Simplification**
Lumped	No	Yes	High
IH + Uniform T	Only for *IH*	Only for *IH*	Mid
Complete	Yes	Yes	Low

5.1.1 Lumped approximation

The lumped approximation is usually adopted for micron sized droplets since both *Fourier* and *Biot* numbers respect the lumped conditions even from the beginning of the process. The energy equation, the mass transfer governing equation and the radius tracking equation are solved using a Euler implicit integration scheme with an iterative method derived from the one described in [24] and in [10]. Time has been discretized with a timestep Δt that leads to:

$$t = n\Delta t \qquad n = 0, \ldots, N. \tag{5.1}$$

The set of equation being discretized is given below:

$$h_{fg}^{n+1}\dot{m}_v^{n+1} + c_{p,d}^{n+1}m_d^{n+1}\frac{T_d^{n+1}-T_d^n}{\Delta t} = h^{n+1}A_d^{n+1}(T_{air}-T_d^{n+1}) \tag{5.2}$$

$$\dot{m}_v^{n+1} = h_{mass}^{n+1}(\rho_{v,s}^{n+1}-\rho_{v,\infty})A_d^{n+1} \tag{5.3}$$

$$\frac{R_d^{n+1}-R_d^n}{\Delta t} = -\frac{\dot{m}_v^{n+1}}{\rho_w 4\pi[R_d^2]^{n+1}} \tag{5.4}$$

Where $h_{fg}^{n+1} = f(T_d^{n+1})$ is the specific heat of evaporation, $c_{p,d} = c_{p,w}(1-c)+c_{p,solid}c$ is a function of the solid concentration c, and the convective heat transfer and mass transfer coefficients come from Eq.(3.96) and Eq.(3.94).

The algorithm for the numerical solution is given in Fig.5.1.

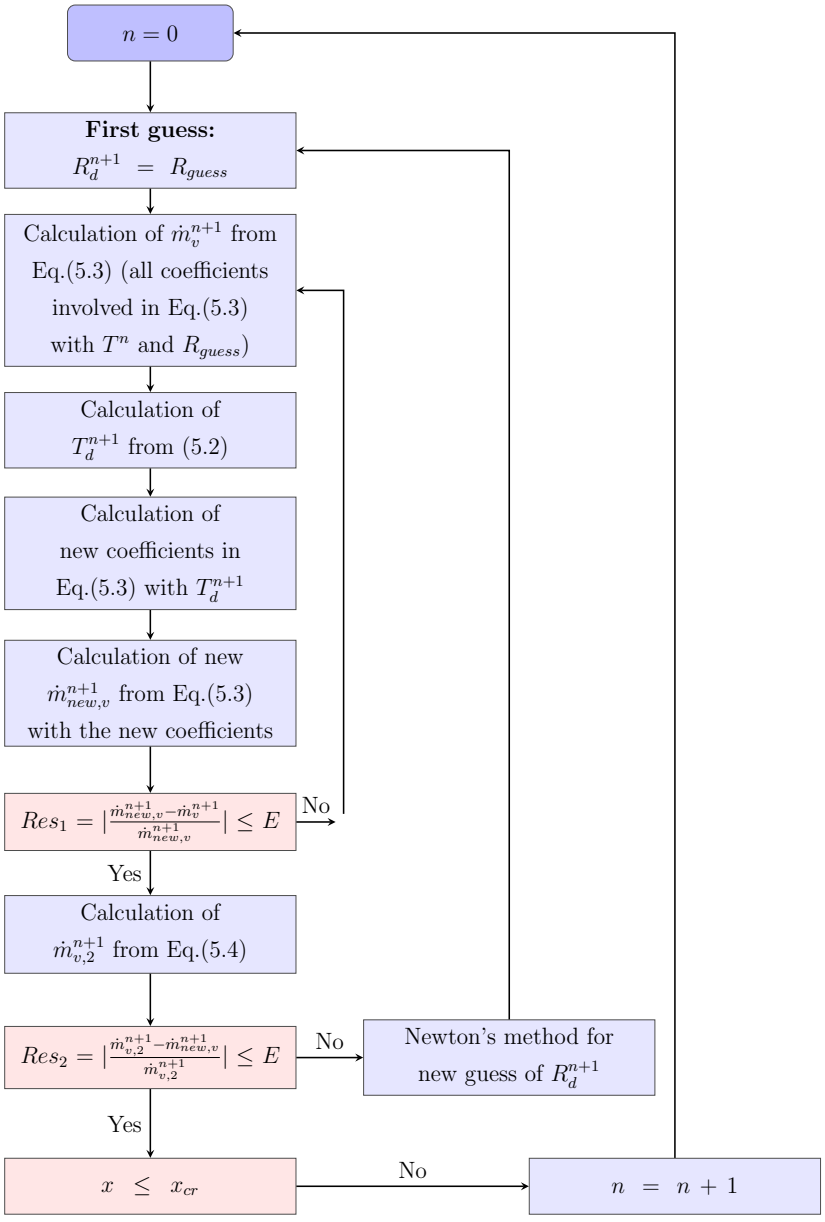


Figure 5.1: *Flow chart for the numerical solution.*

The Newton's method in the above flowchart must give the right "direction" along which to choose a new guess for the droplet radius. Imagine to have an objective function to reduce under a certain tolerance, as an example, for the algorithm, Res_2. The simplest way to proceed is just to take two guesses for the radius, $R_{guess,A}$ and $R_{guess,B}$, and evaluate two values for Res_2 that, for example, are $Res_{2,A}$ and $Res_{2,B}$. The numerical derivative is computed as:

$$\frac{\Delta Res_2}{\Delta R_{guess}} = \frac{Res_{2,A} - Res_{2,B}}{R_{guess,A} - R_{guess,B}}. \tag{5.5}$$

The new value for $R_{guess,C}$ is computed as:

$$R_{guess,C} = R_{guess,B} - \frac{Res_{2,B}}{\frac{\Delta Res_2}{\Delta R_{guess}}}. \tag{5.6}$$

With the value obtained from Eq.(5.6) we could update the value of Eq.(5.5)

However this is not the exact way we proceeded on. The above numerical derivative is analytically computed; since $\dot{m}_{v,2}^{n+1}$ is expected to converge to some finite value, our objective function to reduce is $\Delta\dot{m}_v = \dot{m}_{v,2}^{n+1} - \dot{m}_{new,v}^{n+1}$ and it is evaluated only by replacing the numerical value coming from the first loop $m_{new,v}^{n+1}$. If we denote by $x = R_{guess}$ then we can write

$$\Delta\dot{m}_v = x^3 - R_d^n x^2 + \frac{m_{new,v}^{n+1}}{4\pi\rho_w}. \tag{5.7}$$

The derivative is then computed analytically as:

$$\frac{\Delta\dot{m}}{dx} = 3x^2 - 2xR_d^n. \tag{5.8}$$

Finally $R_{guess,new}$ is computed as:

$$R_{guess,new} = R_{guess} - \frac{\Delta\dot{m}(R_{guess})}{\frac{\Delta\dot{m}}{dx}|_{R_{guess}}}. \tag{5.9}$$

Another temporal dicretization has been also implemented. This algorithm consists in a second order accurate time integration scheme using the explicit multistep Adams-Bashforth method.

$$\begin{aligned} T_d^{n+2} = T_d^{n+1} + \frac{3}{2}\Delta t \frac{h^{n+1}4\pi(R_d^{n+1})^2(T_{air} - T_d^{n+1}) - h_{fg}^{n+1}\dot{m}_v^{n+1}}{(c_{p,d}^{n+1}m_d^{n+1})} \\ - \frac{1}{2}\Delta t \frac{h^n 4\pi(R_d^n)^2(T_{air} - T_d^n) - h_{fg}^n\dot{m}_v^n}{(c_{p,d}^n m_d^n)}, \end{aligned} \tag{5.10}$$

$$R_d^{n+2} = R_d^{n+1} + \frac{3}{2}\Delta t \frac{\dot{m}_v^{n+1}}{\rho_w 4\pi(R_d^{n+1})^2}. - \frac{1}{2}\Delta t \frac{\dot{m}_v^n}{\rho_w 4\pi(R_d^n)^2} \tag{5.11}$$

For $\dot{m}_v^n$ the following equation is used:

$$\dot{m}_v^n = h_{mass}^n(\rho_{v,s}^n - \rho_{v,\infty})A_d^n. \tag{5.12}$$

5.1.2 Initial heating and uniform evaporation temperature

The Fo number associated with the initial heating period may not satisfy the lumped approximation and it could be grater than 0.1. This means that the droplet characteristic time associated with the initial heating is greater than the physical time and therefore, if also the $Biot$ number is greater than 0.1, the temperature profile within the droplet should be considered. The discretization of Eq.(3.102) consists of two parts: spatial and temporal discretization. After some algebra Eq.(3.102) is written as:

$$\rho_d c_{d,p} \frac{\partial T_d(r,t)}{\partial t} = \frac{2k_d}{r} \frac{\partial T_d(r,t)}{\partial r} + k_d \frac{\partial^2 T_d(r,t)}{\partial r^2}. \tag{5.13}$$

Since the above equation is not defined in $r = 0$, using the Hospital's rule the mid term in the above equation may be approximated as follows:

$$\frac{1}{r} \frac{\partial T_d(r,t)}{\partial r}\Big|_{r=0} = \frac{\frac{\partial}{\partial r}\left(\frac{\partial T_d(r,t)}{\partial r}\right)}{\frac{\partial r}{\partial r}}\Bigg|_{r=0} = \frac{\partial^2 T_d(r,t)}{\partial r^2}\Big|_{r=0}. \tag{5.14}$$

Hence in $r = 0$:

$$\rho_d c_{d,p} \frac{\partial T_d(r,t)}{\partial t} = 3k \frac{\partial^2 T_d(r,t)}{\partial r^2}. \tag{5.15}$$

Spatial discretization

The spatial discretization has been made using a second order central finite differences scheme has been adopted. The radius discretization leads to:

$$r = i\Delta r \qquad i = 0, \ldots, M. \tag{5.16}$$

In order to use the central finite differences approximation also in $r = 0$, an artificial node is added before the first node:

$$\rho_d c_{d,p} \frac{\partial T_0(t)}{\partial t} = 3k\Big(\frac{T_{-1}(t) - 2T_0(t) + T_1(t)}{\Delta r^2}\Big). \tag{5.17}$$

The symmetry boundary condition for Eq.(3.102), in $r = 0$ and therefore for $i = 0$ states:

$$\frac{\partial T(t)}{\partial r}\Big|_{r=0} = \frac{T_{-1}(t) - T_{,1}(t)}{2\Delta r} = 0 \quad \Longrightarrow T_{-1}(t) = T_1(t). \tag{5.18}$$

Substituting Eq.(5.18) in Eq.(5.17) yields:

$$\rho_d c_{d,p} \frac{\partial T(t)}{\partial t} = \frac{6k}{\Delta r^2}(T_1(t) - T_0(t)) \quad for \qquad r = 0 \tag{5.19}$$

For $r \neq 0$ Eq.(3.102) becomes:

$$\rho_d c_{d,p} \frac{\partial T_i(t)}{\partial t} = \frac{2k}{i\Delta r} \frac{T_{i+1}(t) - T_{i-1}(t)}{2\Delta r} + k \frac{T_{i+1}(t) - 2T_i(t) + T_{i-1}(t)}{\Delta r^2} \qquad i = 1, \ldots, M. \tag{5.20}$$

The convective boundary condition is discretized using again an artificial node beyond the last node:

$$k\frac{T_{M+1}(t)-T_{M-1}(t)}{2\Delta r}+hT_M(t)=hT_{air}. \tag{5.21}$$

Evaluating Eq.(5.20) in $i=M$, and substituting Eq.(5.21) in Eq.(5.20), it is possible to have the condition for the last node:

$$\begin{aligned}\rho_d c_{d,p}\frac{\partial T_M(t)}{\partial t} =&\frac{k}{M\Delta r}\frac{T_{M+1}(t)-T_{M-1}(t)}{\Delta r}+\\ &+k\frac{T_{M+1}(t)-2T_M(t)+T_{M-1}(t)}{\Delta r^2}\end{aligned}$$

$$\begin{aligned}\Longrightarrow \rho_d c_{d,p}\frac{\partial T_M(t)}{\partial t} =&\frac{k}{M\Delta r}\frac{2\Delta r}{k}h(T_{air}-T_M(t))+\frac{k}{\Delta r^2}(2T_{M-1}(t)+\\ &+\frac{2\Delta r}{k}h(T_{air}-T_M(t))-2T_M(t))\end{aligned}$$

$$\begin{aligned}\Longrightarrow \rho_d c_{d,p}\frac{\partial T_M(t)}{\partial t} =&\frac{2h}{M\Delta r}(T_{air}-T_M(t))+\frac{k}{\Delta r^2}(2T_{M-1}(t)+\\ &+\frac{2\Delta r}{k}hT_{air}-\frac{2\Delta r}{k}hT_M(t)-2T_M(t))\end{aligned}$$

$$\begin{aligned}\Longrightarrow \rho_d c_{d,p}\frac{\partial T_M(t)}{\partial t} =&\frac{2h}{M\Delta r}T_{air}-\frac{2h}{M\Delta r}T_M(t)+\frac{k}{\Delta r^2}2T_{M-1}(t)+\\ &+\frac{2}{\Delta r}hT_{air}-\frac{2}{\Delta r}hT_M(t)-\frac{2k}{\Delta r^2}T_M(t)\end{aligned}$$

$$\begin{aligned}\Longrightarrow \rho_d c_{d,p}\frac{\partial T_M(t)}{\partial t} =&\frac{2k}{\Delta r^2}T_{M-1}(t)+(-\frac{2h}{\Delta r}-\frac{2k}{\Delta r^2}-\frac{2h}{M\Delta r})T_M(t)+\\ &+(\frac{2h}{M\Delta r}+\frac{2h}{\Delta r})T_{air}.\end{aligned} \tag{5.22}$$

Temporal discretization

Two temporal discretizations have been implemented:

- Second order explicit Adams-Bashforth;
- First order implicit Euler;

As regards the explicit method we have:

$$\begin{aligned}\rho_d c_{d,p}\frac{\Delta T_i^{n+2}-T_i^{n+1}}{\Delta t} =&\frac{3}{2}\frac{2k}{i\Delta r}\frac{T_{i+1}^{n+1}-T_{i-1}^{n+1}}{2\Delta r}+\\ &+k\frac{T_{i+1}^{n+1}-2T_i^{n+1}+T_{i-1}^{n+1}}{\Delta r^2}-\frac{1}{2}\frac{2k}{i\Delta r}\frac{T_{i+1}^{n}-T_{i-1}^{n}}{2\Delta r}+\\ &+k\frac{T_{i+1}^{n}-2T_i^{n}+T_{i-1}^{n}}{\Delta r^2}\qquad i=1,\dots,M.\end{aligned} \tag{5.23}$$

As well known, the explicit Adams-Bashforth method requires small timestep to guarantee numerical stability. Therefore the Euler implicit method, that is unconditionally stable, allows more flexibility in choosing separately the timestep and the number of points along the radius. The implicit method requires to solve, in each timestep, the linear system:

$$\begin{aligned}&\Big(\frac{\alpha\Delta t}{\Delta r^2}\Big(\frac{1}{i}-1\Big)\Big)T_{i-1}^{n+1}+\Big(1+\frac{2\alpha\Delta t}{\Delta r^2}\Big)T_i^{n+1}+\\&+\Big(-\frac{\alpha\Delta t}{\Delta r^2}\Big(\frac{1}{i}+1\Big)\Big)T_{i+1}^{n+1}=T_i^n \qquad i=1,\dots,M-1,\end{aligned} \tag{5.24}$$

$$\Big(1+\frac{6\alpha\Delta t}{\Delta r^2}\Big)T_0^{n+1}-\frac{6\alpha\Delta t}{\Delta r^2}T_1^{n+1}=T_0^n \qquad i=0, \tag{5.25}$$

$$\begin{aligned}&\Big(1+\frac{4h\alpha\Delta t}{kM}+\frac{2\alpha\Delta t}{\Delta r^2}+\frac{2h\alpha\Delta t}{\Delta rk}\Big)T_M^{n+1}-\\&-\frac{2\alpha\Delta t}{\Delta r^2}T_{M-1}^{n+1}=T_M^n+\Big(\frac{4h\alpha\Delta t}{kM}+\frac{2h\alpha\Delta t}{\Delta rk}\Big) \qquad i=M.\end{aligned} \tag{5.26}$$

It should be noted that the droplet temperature increases until the evaporation temperature is reached but, according to this model without evaporation during the initial heating, this temperature is imposed from an equilibrium equation derived from the uniform temperature approximation (Eq.(3.103)). This approximation is based on the assumption of constant equilibrium evaporation temperature of the droplet (see Fig.3.3). Eq.(3.103) is a nonlinear equation in the variable T_d that one can solve with any preferred method (Newton, secant method, regula falsi etc...). A first order implicit Euler method and a second order Adams-Bashforth method have been adopted for temporal discretization (Eq.(3.100)) resulting in:

$$\begin{aligned}&\frac{R_d^{n+1}-R_d^n}{\Delta t}=-\frac{\dot m_v}{\rho_w 4\pi[R_d^2]^{n+1}}\\&R_d^{n+2}=R_d^{n+1}+\frac{3}{2}\Delta t\frac{\dot m_v^{n+1}}{\rho_w 4\pi(R_d^{n+1})^2}-\frac{1}{2}\Delta t\frac{\dot m_v^n}{\rho_w 4\pi(R_d^n)^2}.\end{aligned} \tag{5.27}$$

The droplet mass can be computed by solving with Eq.(3.101).

5.1.3 Complete

In the case a non-uniform droplet temperature is assumed during the evaporation period of the first drying stage, the partial differential equation Eq.(3.105) applied to the time-dependent spatial domain $0 \le r \le R_d$ must be solved simultaneously with its boundary conditions, Eq.(3.5.6), and additional equations Eq.(3.100) and

Eq.(3.91) [24]. The numerical solution is complicated by the presence of a variable spatial domain $0 \leq r \leq R_d$ due to droplet evaporation. The procedure adopted to solve this mouving boundary problem is described in [10] and in appendix A of [24]. The main idea is to apply a coordinate transformation in order to remove the time dependence from the boundary condition. A Landau's transformation is adopted:

$$\begin{cases} t = \tau = \psi(\xi, \tau), \\ r = \xi R_d(\tau) = \phi(\xi, \tau). \end{cases}$$

With $0 \leq \xi \leq 1$.

$$T_d(r,t) \rightarrow T_d(\xi,\tau), \tag{5.28}$$

$$\begin{aligned} \frac{\partial}{\partial r} &= \frac{1}{R_d(\tau)} \frac{\partial}{\partial \xi}, \\ \frac{\partial^2}{\partial r^2} &= \frac{1}{R_d^2(\tau)} \frac{\partial^2}{\partial \xi^2}, \\ \frac{\partial}{\partial t} &= \frac{\partial}{\partial \tau} - \frac{\xi}{R_d(\tau)} \frac{dR_d(\tau)}{d\tau} \frac{\partial}{\partial \xi}. \end{aligned} \tag{5.29}$$

Substituting Eq.(5.28) and using the above formulas in Eq.(3.105) with $k_d \neq f(r)$ we get:

$$\frac{\partial T_d(\xi,\tau)}{\partial \tau} = \Big[\frac{\xi}{R_d(\tau)} \frac{dR_d(\tau)}{d\tau} + \frac{2\alpha_d}{\xi R_d^2(\tau)}\Big] \frac{\partial T_d(\xi,\tau)}{\partial \xi} + \frac{\alpha_d}{R_d^2(\tau)} \frac{\partial^2 T_d(\xi,\tau)}{\partial \xi^2}. \tag{5.30}$$

Restoring $t = \tau$ we obtain

$$\frac{\partial T_d(\xi,t)}{\partial t} = \Big[\frac{\xi}{R_d(t)} \frac{dR_d(t)}{dt} + \frac{2\alpha_d}{\xi R_d^2(t)}\Big] \frac{\partial T_d(\xi,t)}{\partial \xi} + \frac{\alpha_d}{R_d^2(t)} \frac{\partial^2 T_d(\xi,t)}{\partial \xi^2}. \tag{5.31}$$

The boundary conditions become:

$$\begin{cases} \frac{\partial T_d(\xi,t)}{\partial \xi} = 0 \qquad for \qquad \xi = 0 \\ h(T_g - T_d) 4\pi R_d^2(t) = -k_d \frac{\partial T_d}{\partial \xi} 4\pi R_d(t) + h_{fg} \dot{m}_v \qquad for \qquad \xi = 1 \end{cases}$$

The spatial domain, now in the transformed variable, $0 \leq \xi \leq 1$ and time are discretized as follows:

$$\begin{aligned} \xi &= i\Delta\xi \qquad i = 0, \ldots, M, \\ t &= n\Delta t \qquad n = 0, \ldots, N. \end{aligned} \tag{5.32}$$

The Euler implicit discretization of Eq.(5.30) gives:

$$\frac{(T_d)_i^{n+1}-(T_d)_i^n}{\Delta t}=\left[\frac{i\Delta\xi}{R_d^{n+1}}\frac{R_d^{n+1}-R_d^n}{\Delta t}+\frac{2\alpha_d^n}{i\Delta\xi(R_d^{n+1})^2}\right]\frac{(T_d)_{i+1}^{n+1}-(T_d)_{i-1}^{n+1}}{2\Delta\xi}+ \frac{\alpha_d^n}{(R_d^{n+1})^2}\frac{(T_d)_{i+1}^{n+1}-2(T_d)_i^{n+1}+(T_d)_{i-1}^{n+1}}{\Delta\xi^2}. \tag{5.33}$$

We denote by:

$$A_i^{n+1}=\frac{(\alpha_d)^n\Delta t}{\Delta\xi^2[(R_d)^{n+1}]^2}, \tag{5.34}$$

$$B_i^{n+1}=\frac{i(R_d^{n+1}-R_d^n)}{2R_d^{n+1}}+\frac{A_i^{n+1}}{i}, \tag{5.35}$$

and substitute these expressions into Eq.(5.33). The following linear system to solve is obtained:

$$(T_d)_i^n=(B_i^{n+1}-A_i^{n+1})(T_d)_{i-1}^{n+1}+(1+2A_i^{n+1})(T_d)_i^{n+1}-(B_i^{n+1}+A_i^{n+1})(T_d)_{i+1}^{n+1}. \tag{5.36}$$

Following the same approach described in section 5.1.2, an approximation of Eq.(5.31) in $\xi=0$ is needed. The term $\frac{1}{\xi}\frac{\partial T_d(\xi,t)}{\partial\xi}|_{\xi=0}$ is set equal to $\frac{\partial^2 T_d(\xi,t)}{\partial^2\xi}|_{\xi=0}$ and then, for $i=0$ along with the symmetry boundary condition $(T_d)_{-1}=(T_d)_1$, Eq.(5.31) gives the condition for the first node:

$$(T_d)_0^{n+1}=\frac{6A_0^{n+1}}{1+6A_0^{n+1}}(T_d)_1^{n+1}+\frac{1}{1+6A_0^{n+1}}(T_d)_0^n. \tag{5.37}$$

The boundary condition for $\xi=1$ is discretized as follows:

$$h^{n+1}[T_{air}-(T_d)_M^{n+1}]4\pi(R_d^{n+1})^2=-k_d^{n+1}\frac{(T_d)_{M+1}^{n+1}-(T_d)_{M-1}^{n+1}}{2\Delta\xi}4\pi R_d^{n+1}+h_{fg}^{n+1}\dot{m}_v^{n+1}. \tag{5.38}$$

Thus, evaluating Eq.(5.31) for $i=M$, and using Eq.(5.38), the condition for the last node is evaluated as:

$$(T_d)_M^{n+1}=\frac{(T_d)_M^n+2A_M^{n+1}(T_d)_{M-1}^{n+1}+2\Delta\xi(B_M^{n+1}+A_M^{n+1}\left[Bi^{n+1}T_{air}-\frac{h_{fg}^{n+1}\dot{m}_v^{n+1}}{4\pi k_d^{n+1}(R_d)^{n+1}}\right]}{1+2A_M^{n+1}+2\Delta\xi Bi^{n+1}(B_M^{n+1}+A_M^{n+1})}, \tag{5.39}$$

where $Bi^{n+1}=\frac{h^{n+1}R_d^{n+1}}{k_d^{n+1}}$ is the Biot number. The linear system is now solved using a backward elimination Gauss algorithm [12],[25],[24].

$$(T_d)_i^{n+1}=a_i^{n+1}(T_d)_{i+1}^{n+1}+b_i^{n+1}, \tag{5.40}$$

Being a and b the Gauss coefficients:

$$For \quad 1 \leq i \leq M-1:$$

$$a_i^{n+1} = \frac{A_i^{n+1} + B_i^{n+1}}{1 + 2A_i^{n+1} - a_{i-1}^{n+1}(A_i^{n+1} - B_i^{n+1})} \quad b_i^{n+1} = \frac{(T_d)_i^n + (A_i^{n+1} - B_i^{n+1} b_{i-1}^{n+1}}{1 + 2A_i^{n+1} - a_{i-1}^{n+1}(A_i^{n+1} - B_i^{n+1})},$$

$$For \quad i = 0:$$

$$a_0^{n+1} = \frac{6A_0^{n+1}}{1 + 6A_0^{n+1}} \quad b_0^{n+1} = \frac{1}{1 + 6A_0^{n+1}}(T_d)_0^n,$$

$$For \quad i = M:$$

$$a_M^{n+1} = 0 \quad b_M^{n+1} = \frac{(T_d)_M^{n+1} + 2A_M^{n+1} b_{M-1}^{n+1} + 2\Delta\xi(B_M^{n+1} + A_M^{n+1}\left[Bi^{n+1} T_{air} - \frac{h_{fg}^{n+1}\dot{m}_v^{n+1}}{4\pi k_d^{n+1} R_d^{n+1}}\right]}{1 + 2A_M^{n+1}(1 - a_{M-1}^{n+1}) + 2\Delta\xi Bi^{n+1}(B_M^{n+1} + A_M^{n+1})}.$$

Then, starting from the last node, where $(T_d)_M^{n+1} = b_M^{n+1}$, we can evaluate at each time step the droplet temperature for all points along the radius. Finally using Eq.(5.3) and Eq.(5.4) it is possible to evaluate mass transfer and droplet shrinkage rates. The algorithm for the solution of this problem is shown in Fig5.4.

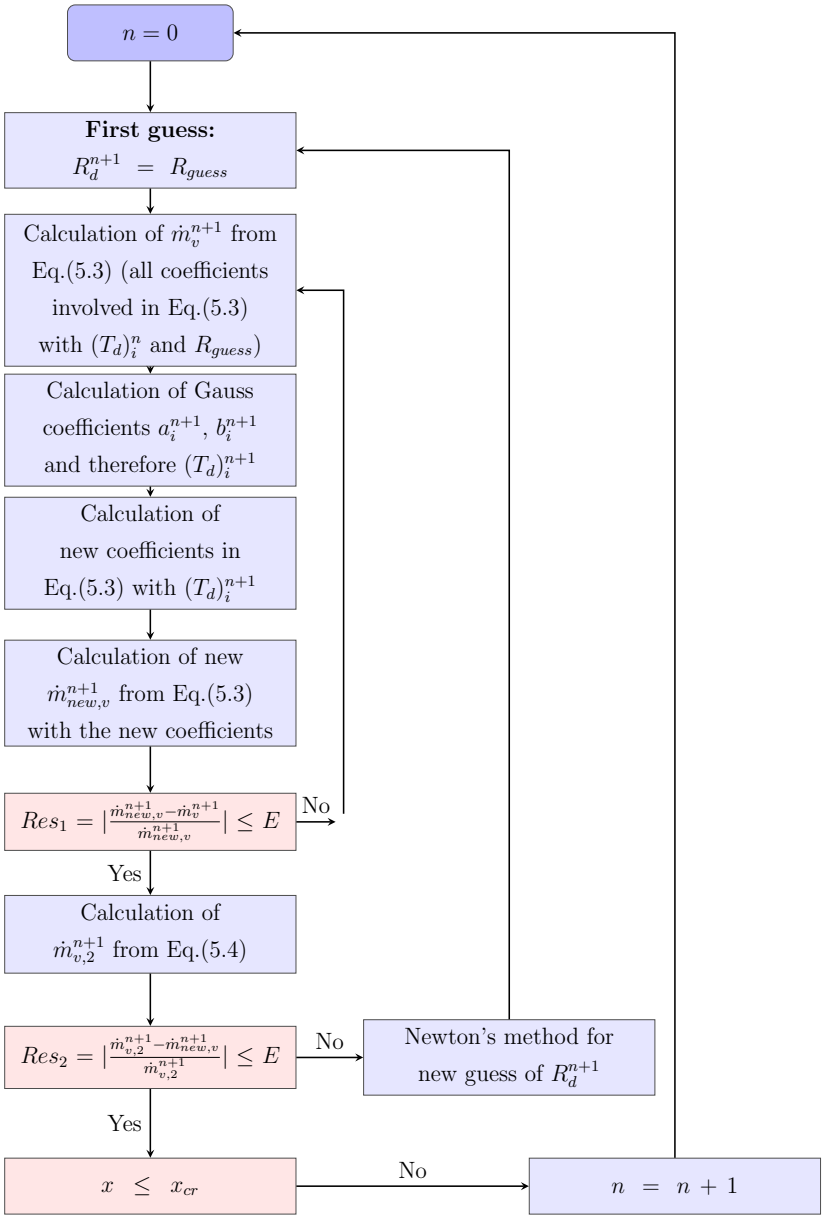


Figure 5.2: *Flow chart for the numerical solution.*

5.2 Second Stage

The numerical solution of the set of PDEs described in section 3.5.7 is complex by the presence of variable spatial domain for each particle region due to evaporating moving crust-wet core interface and by unknown temperature of this interface. Similarly to the droplet evaporation period, applying the Landau's transformation [10] allows fixing the position of the crust-wet core interface. For the region of particle

wet core we use:

$$\begin{cases} t = \tau = \psi_1(\xi_1, \tau), \\ r = \xi_1 R_i(\tau) = \phi_1(\xi_1, \tau). \end{cases}$$

Consequently:

$$\begin{aligned} & T_{wc}(r,t) \to T_{wc}(\xi_1, \tau), \\ & \frac{\partial}{\partial r} = \frac{1}{R_i(\tau)} \frac{\partial}{\partial \xi_1}, \\ & \frac{\partial^2}{\partial r^2} = \frac{1}{R_i^2(\tau)} \frac{\partial^2}{\partial \xi_1^2}, \\ & \frac{\partial}{\partial t} = \frac{\partial}{\partial \tau} - \frac{\xi_1}{R_i(\tau)} \frac{dR_i(\tau)}{d\tau} \frac{\partial}{\partial \xi_1}. \end{aligned} \tag{5.41}$$

For the region of particle crust:

$$\begin{cases} t = \tau = \psi_2(\xi_2, \tau), \\ r = R_i(\tau) + \xi_2[R_p - R_i(\tau)] = \phi_2(\xi_2, \tau). \end{cases}$$

As a result:

$$\begin{aligned} & T_{cr}(r,t) \to T_{cr}(\xi_2, \tau), \\ & \frac{\partial}{\partial r} = \frac{1}{R_p - R_i(\tau)} \frac{\partial}{\partial \xi_2}, \\ & \frac{\partial^2}{\partial r^2} = \frac{1}{[R_p - R_i(\tau)]^2} \frac{\partial^2}{\partial \xi_2^2}, \\ & \frac{\partial}{\partial t} = \frac{\partial}{\partial \tau} - \frac{1 - \xi_2}{R_p - R_i(\tau)} \frac{dR_i(\tau)}{d\tau} \frac{\partial}{\partial \xi_2}. \end{aligned} \tag{5.42}$$

The transformation of equation Eq.(3.107) gives:

$$\frac{\partial T_{wc}(\xi_1, t)}{\partial t} = \Big[\frac{\xi_1}{R_i(t)} \frac{dR_i(t)}{dt} + \frac{2\alpha_{wc}}{\xi_1 R_i^2(t)}\Big] \frac{\partial T_{wc}(\xi_1, t)}{\partial \xi_1} + \frac{\alpha_{wc}}{R_i^2(t)} \frac{\partial^2 T_{wc}(\xi_1, t)}{\partial \xi_1^2}, \tag{5.43}$$

with $0 \leq \xi_1 \leq 1$. The boundary conditions become:

$$\begin{cases} \frac{\partial T_{wc}(\xi_1, t)}{\partial \xi_1} = 0 \qquad for \quad \xi_1 = 0, \\ T_{wc}(\xi_1, t) = T_{cr}(\xi_2, t) \qquad for \quad \xi_1 = 1. \end{cases}$$

In Eq.(5.2) the condition of heat balance at the crust-wet core interface is omitted since its transformation is discussed separately below. The tranformation of equation Eq.(3.106) gives:

$$\begin{aligned} [R_p - R_i(t)] \frac{\partial T_{cr}(\xi_2, t)}{\partial t} = & \Big[(1 - \xi_2) \frac{dR_i(t)}{dt} + \frac{2\alpha_{cr}}{R_i(t) + \xi_1[R_p - R_i(t)]}\Big] \frac{\partial T_{cr}(\xi_2, t)}{\partial \xi_2} + \\ & + \frac{\alpha_{cr}}{R_p - R_i(t)} \frac{\partial^2 T_{cr}(\xi_2, t)}{\partial \xi_2^2}, \end{aligned} \tag{5.44}$$

with $0 \leq \xi_2 \leq 1$.

$$\begin{cases} T_{cr}(\xi_2, t) = T_{wc}(\xi_1, t) \qquad for \quad \xi_2 = 0, \\ h[T_g - T_{cr}(\xi_2, t)] = \frac{k_{cr}}{R_p - R_i(t)} \frac{\partial T_{cr}(\xi_2, t)}{\partial \xi_2} = 0 \qquad for \quad \xi_2 = 1. \end{cases}$$

The set of equations described above are solved using the fully implicit finite differences scheme with fixed time step described in [24] and [10]. According to Landau's transformation the spatial domain and time are discretized as follows:

$$\begin{aligned} \xi_1 &= i\Delta\xi_1 \qquad i = 0, \ldots, M1, \\ \xi_2 &= j\Delta\xi_2 \qquad j = 0, \ldots, M2, \\ t &= n\Delta t \qquad n = 0, \ldots, N. \end{aligned} \tag{5.45}$$

The Euler implicit discretization of Eq.(5.43) gives:

$$\begin{aligned} \frac{(T_{wc})_i^{n+1} - (T_{wc})_i^n}{\Delta t} = \Bigg[\frac{i\Delta\xi_1}{R_i^{n+1}} \frac{R_i^{n+1} - R_i^n}{\Delta t} + \\ + \frac{2\alpha_{wc}^n}{i\Delta\xi_1 (R_i^{n+1})^2} \Bigg] \frac{(T_{wc})_{i+1}^{n+1} - (T_{wc})_{i-1}^{n+1}}{2\Delta\xi_1} + \frac{\alpha_{wc}^n}{(R_i^{n+1})^2} \frac{(T_{wc})_{i+1}^{n+1} - 2(T_{wc})_i^{n+1} + (T_{wc})_{i-1}^{n+1}}{\Delta\xi_1^2}. \end{aligned} \tag{5.46}$$

If we denote by:

$$E_i^{n+1} = \frac{(\alpha_{wc})^n \Delta t}{\Delta\xi_1^2 [(R_i)^{n+1}]^2}, \tag{5.47}$$

$$F_i^{n+1} = \frac{i(R_i^{n+1} - R_i^n)}{2R_i^{n+1}} + \frac{E_i^{n+1}}{i}, \tag{5.48}$$

and substitute these expressions into Eq.(5.46) we get the following linear system to solve:

$$(T_{wc})_i^n = (F_i^{n+1} - E_i^{n+1})(T_{wc})_{i-1}^{n+1} + (1 + 2E_i^{n+1})(T_{wc})_i^{n+1} - (F_i^{n+1} + E_i^{n+1})(T_{wc})_{i+1}^{n+1}, \tag{5.49}$$

with $1 \leq i \leq M1 - 1$. Following the same approach described in section 5.1.2, an approximation of Eq.(5.43) in $\xi_1 = 0$ is needed. The term $\frac{1}{\xi_1} \frac{\partial T_{wc}(\xi_1, t)}{\partial \xi_1}|_{\xi_1 = 0}$ is set equal to $\frac{\partial^2 T_{wc}(\xi_1, t)}{\partial^2 \xi_1}|_{\xi_1 = 0}$ and then, for $i = 0$ and using the symmetry boundary condition $(T_{wc})_{-1} = (T_{wc})_1$, Eq.(5.43) gives the condition for the first node:

$$(T_{wc})_0^{n+1} = \frac{6E_0^{n+1}}{1 + 6E_0^{n+1}} (T_{wc})_1^{n+1} + \frac{1}{1 + 6E_0^{n+1}} (T_{wc})_0^n. \tag{5.50}$$

Assuming that the temperature at crust-wet core interface is a parameter (it will be calculated later in this section), the linear system is solved using a backward elimination Gauss algorithm [12],[25],[24]:

$$(T_{wc})_i^{n+1} = d_i^{n+1} (T_{wc})_{i+1}^{n+1} + e_i^{n+1}, \tag{5.51}$$

being d and e the Gauss coefficients:

$$For \quad 1 \leq i \leq M1-1:$$

$$d_i^{n+1} = \frac{E_i^{n+1}+F_i^{n+1}}{1+2E_i^{n+1}-d_{i-1}^{n+1}(E_i^{n+1}-F_i^{n+1})} \quad e_i^{n+1} = \frac{(T_{wc})_i^n + (E_i^{n+1}-F_i^{n+1})e_{i-1}^{n+1}}{1+2E_i^{n+1}-d_{i-1}^{n+1}(E_i^{n+1}-F_i^{n+1})},$$

$$For \quad i=0:$$

$$d_0^{n+1} = \frac{6E_0^{n+1}}{1+6E_0^{n+1}} \quad e_0^{n+1} = \frac{1}{1+6E_0^{n+1}}(T_{wc})_0^n.$$

Then, if we suppose that $(T_{wc})_{M1}^{n+1}$ is known, it is possible to evaluate all $(T_{wc})_i^{n+1}$ in reversed order, beginning from $i = M1-1$ and finishing by $i=0$. What concerns the crust region (Eq.(5.42)) the Euler implicit discretization gives:

$$\begin{aligned}\frac{(T_{cr})_j^{n+1}-(T_{cr})_j^n}{\Delta t} = \Bigg[(1-j\Delta\xi_2)\frac{(R_j)^{n+1}-(R_j)^n}{\Delta t}+ \\ + \frac{2(\alpha_{cr})^n}{(R_j)^{n+1}+j\Delta\xi_2[R_p-(R_j)^{n+1}]}\Bigg]\frac{(T_{cr})_{j+1}^{n+1}-(T_{cr})_{j-1}^{n+1}}{2\Delta\xi_2[R_p-(R_j)^{n+1}]}+ \\ + \frac{(\alpha_{cr})^n}{[R_p-(R_j)^{n+1}]^2}\frac{(T_{cr})_{j+1}^{n+1}-2(T_{cr})_j^{n+1}+(T_{cr})_{j-1}^{n+1}}{\Delta\xi_2^2}.\end{aligned} \tag{5.52}$$

If we denote by:

$$\begin{aligned}s^{n+1} &= R_p-(R_j)^{n+1}, \\ G_j^{n+1} &= \frac{(\alpha_{cr})^n\Delta t}{\Delta\xi_2^(s^{n+1})^2}, \\ H_j^{n+1} &= \frac{(1-j\Delta\xi_2)[(R_j)^{n+1}-(R_j)^n]}{2\Delta\xi_2 s^{n+1}}+\frac{\Delta\xi_2 s^{n+1}G_j^{n+1}}{(R_j)^{n+1}+j\Delta\xi_2 s^{n+1}},\end{aligned} \tag{5.53}$$

and substitute these expressions into Eq.(5.52) we get the following linear system to solve:

$$(T_{cr})_j^n = (H_j^{n+1}-G_jn+1)(T_{cr})_{j-1}^{n+1}+(1+2G_j^{n+1})(T_{cr})_j^{n+1}-(H_j^{n+1}+G_jn+1)(T_{cr})_{j+1}^{n+1}, \tag{5.54}$$

with $1 \leq j \leq M2$. Because it is assumed that the value of temperature at the crust-wet core interface, $T_{cr}(0,t)$ is a parameter, only the boundary condition at the particle outer surface is considered. The discretization of this boundary condition yields:

$$\begin{aligned}h^{n+1}[T_g-(T_{cr})_{M2}^{n+1}] &= \frac{(k_{cr})^{n+1}}{s^{n+1}}\frac{(T_{cr})_{M2+1}^{n+1}-(T_{cr})_{M2-1}^{n+1}}{2\Delta\xi_2} \\ (T_{cr})_{M2+1}^{n+1} &= (T_{cr})_{M2-1}^{n+1}+2\Delta\xi_2(Bi_{cr})^{n+1}[T_g-(T_{cr})_{M2}^{n+1}],\end{aligned} \tag{5.55}$$

where $(Bi_{cr})^{n+1} = \frac{h^{n+1}s^{n+1}}{k_{cr}^{n+1}}$ is the crust Biot number. Substituting Eq.(5.55) in Eq.(5.54) the condition for the last node is evaluated as follows:

$$(T_{cr})_{M2}^{n+1} = \frac{(T_{cr})_{M2}^{n} + 2G_{M2}^{n+1}(T_{cr})_{M2-1}^{n+1} + 2\Delta\xi_2(Bi_{cr})^{n+1}(H_{M2}^{n+1} + G_{M2}^{n+1})T_g}{1 + 2[G_{M2}^{n+1} + \Delta\xi_2(Bi_{cr})^{n+1}(H_{M2}^{n+1} + G_{M2}^{n+1})]}. \quad (5.56)$$

The linear system is solved using a backward elimination Gauss algorithm [12],[25],[24]:

$$(T_{cr})_j^{n+1} = f_j^{n+1}(T_{cr})_{j+1}^{n+1} + g_j^{n+1}, \quad (5.57)$$

where f and g are the Gauss coefficients:

$For \quad 2 \leq j \leq M2 - 1:$

$$f_j^{n+1} = \frac{G_j^{n+1} + H_j^{n+1}}{1 + 2G_j^{n+1} - f_{j-1}^{n+1}(G_j^{n+1} - H_j^{n+1})} \quad g_j^{n+1} = \frac{(T_{cr})_j^n + (G_j^{n+1} - H_j^{n+1})g_{j-1}^{n+1}}{1 + 2G_j^{n+1} - f_{j-1}^{n+1}(G_j^{n+1} - H_i^{n+1})},$$

$For \quad j = 1:$

$$f_1^{n+1} = \frac{G_1^{n+1} + H_1^{n+1}}{1 + 2G_1^{n+1}} \quad g_1^{n+1} = \frac{(T_{cr})_1^n + (G_1^{n+1} - H_1^{n+1})(T_{cr})_0^{n+1}}{1 + 2G_1^{n+1}},$$

$For \quad j = M2:$

$$f_{M2}^{n+1} = 0 \quad g_{M2}^{n+1} = \frac{(T_{cr})_{M2}^n + 2\Delta\xi_2(Bi_{cr})^{n+1}(H_{M2}^{n+1} + G_{M2}^{n+1})T_g + 2G_{M2}^{n+1}g_{M2-1}^{n+1}}{1 + 2[G_{M2}^{n+1} + \Delta\xi_2(Bi_{cr})^{n+1}(H_{M2}^{n+1} + G_{M2}^{n+1})] - 2G_{M2}^{n+1}f_{M2-1}^{n+1}},$$

then, starting from the last node, where $(T_{cr})_{M2}^{n+1} = g_{M2}^{n+1}$, it is possible to evaluate $(T_{cr})_j^{n+1}$ in reverse order, beginning from $j = M2$ and finishing by $j = 1$. There is also a supplementary condition, which must be satisfied for the both crust and wet core regions of the particle. This condition is the energy balance at the crust-wet core interface. This is the second equation in boundary conditions (3.5.7). That equation can be discretized as follows:

$$(k_cr)_0^{n+1}\frac{T_{cr}(\Delta\xi_2^*)^{n+1} - (T_{cr})_0^{n+1}}{\Delta\xi_2^* s^{n+1}} = (k_{wc})_{M1}^{n+1}\frac{(T_{wc})_{M1}^{n+1} - T_{wc}(1 - \Delta\xi_1^*)^{n+1}}{\Delta\xi_1^*(R_i)^{n+1}} + \frac{(h_{fg})^{n+1}\dot{m}_v^{n+1}}{4\pi[(R_i)^{n+1}]^2}, \quad (5.58)$$

where $\Delta\xi_1^*$ and $\Delta\xi_2^*$ are dimensionless distances from the crust-wet core interface. These parameters are connected by the following relationship:

$$\Delta\xi_1^*(R_i)^{n+1} = \Delta\xi_2^* s^{n+1}. \quad (5.59)$$

Assuming that $\Delta\xi_2^* = \Delta\xi_2$, and since $T_{cr}(\Delta\xi_2^*)^{n+1}$ and $T_{wc}(1 - \Delta\xi_1^*)^{n+1}$ must be equidistant from the both sides of the crust-wet core interface in the spatial domain (see Fig.5.3), $\Delta\xi_1^* = \Delta\xi_2\frac{s^{n+1}}{(R_i)^{n+1}}$. This value must be obtained through the help of an interpolation. Now it is possible to evaluate the residual in the following equation:

$$Res^{n+1} = 1 - \frac{(k_{wc})_{M1}^{n+1}\frac{(T_{wc})_{M1}^{n+1} - T_{wc}(1-(\Delta\xi_1^*)^{n+1})}{\Delta\xi 1^*(R_i)^{n+1}} + \frac{(h_{fg})^{n+1}(\dot{m}_v)^{n+1}}{4\pi[(R_i)^{n+1}]^2}}{(k_{cr})_0^{n+1}\frac{T_{cr}(\Delta\xi_2^*)^{n+1} - (T_{cr})_0^{n+1}}{\Delta\xi_2^* s^{n+1}}}. \quad (5.60)$$

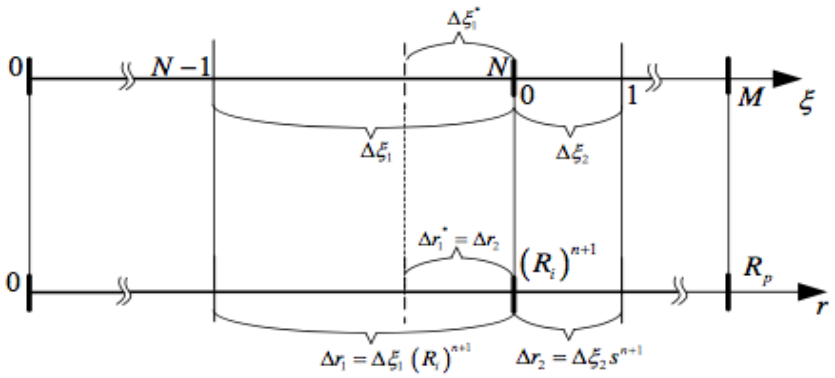


Figure 5.3: *Determination of* $\Delta\xi_1^*$ *[24].*

Numerical expressions for evaluation of mass transfer and wet core shrinkage rates are obtained by discretization of equations Eq.(3.108) and Eq.(3.109).

$$\frac{(R_i)^{n+1} - (R_i)^n}{\Delta t} = -\frac{1}{4\pi\varepsilon(\rho_{wc}^w)^{n+1}[(R_i)^{n+1}]^2}(\dot{m}_v)^{n+1} \tag{5.61}$$

$$(\dot{m}_v)^{n+1} = -\frac{8\pi\varepsilon(D_v)^{n+1}M_w p_g R_p (R_i)^{n+1}}{R_u[(T_{cr})_{M2} + (T_{wc})_{M1}]^{n+1}s^{n+1}} * \tag{5.62}$$
$$* \ln\left[\frac{p_g - (p_{v,i})^{n+1}}{p_g - \left[\frac{R_u}{M_w(h_D)^{n+1}4\pi(R_p)^2}\dot{m}_v^{n+1} + \frac{p_v^\infty}{T_g}\right](T_{cr})_{M2}^{n+1}}\right].$$

The numerical algorithm is presented in the figure 5.4:

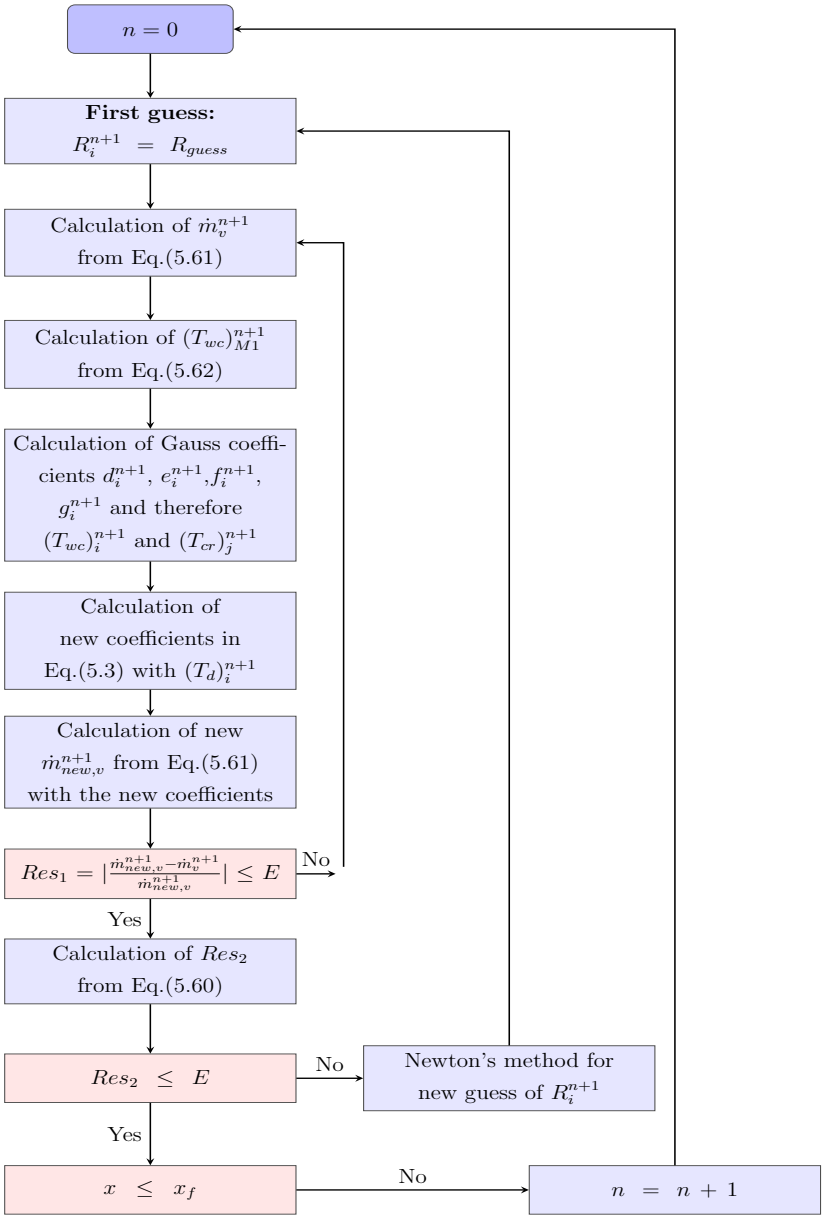


Figure 5.4: *Flow chart for the numerical solution.*

Part V

The case study

Chapter 6

Original and modified geometry

The choice of the spray dryer geometry comes from the master thesis from which this work has been constructed. In the following section is presented the original geometry on which several tests have been conducted and represents a well validated case. Most of the informations were obtained from [2], [13] and [16].

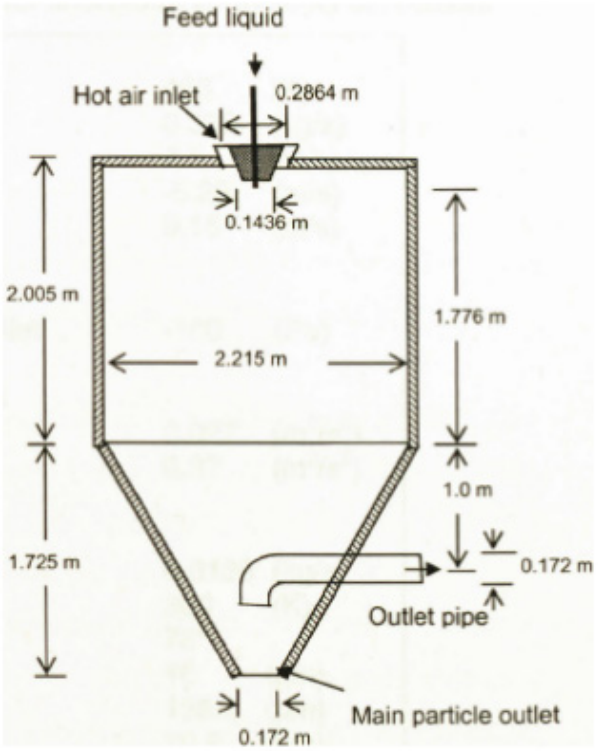


Figure 6.1: *Anandharamakrishnan's geometry [2].*

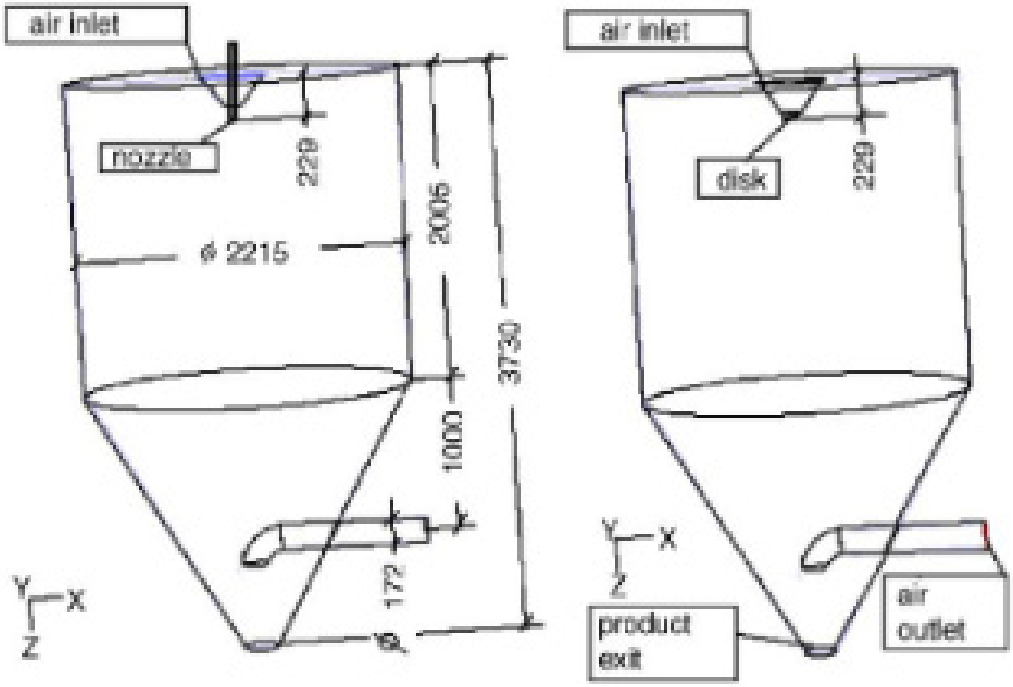


Figure 6.2: *Huang's geometry [16].*

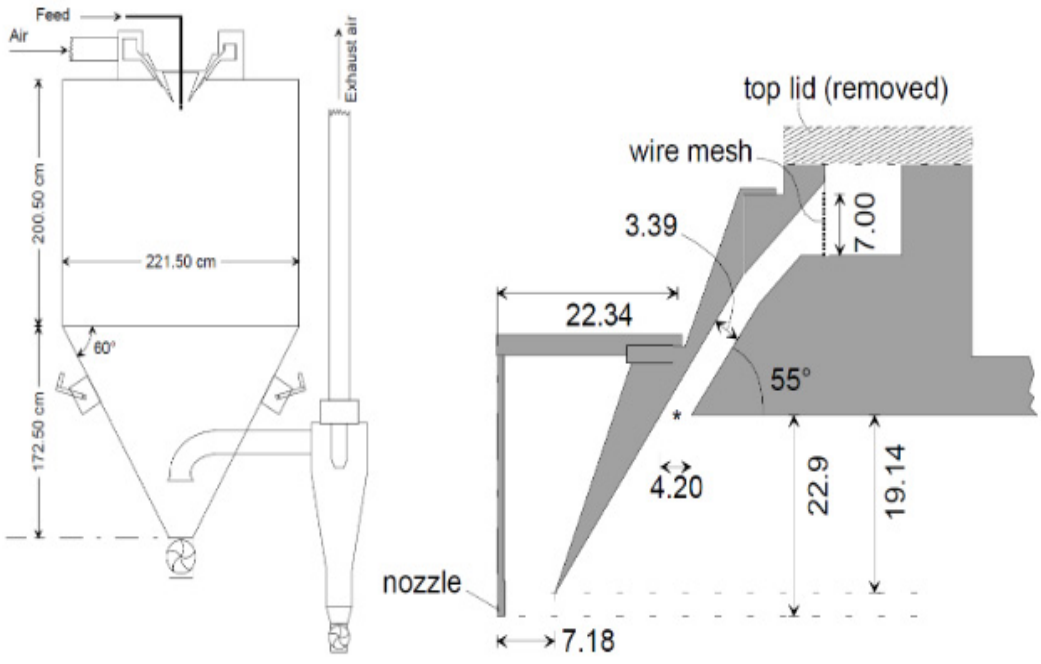


Figure 6.3: *Kieviet's geometry [13].*

The original geometry used, for the spray dryer CFD simulation, has been the result of a very accurate study, especially regarding the inlet region and the outlet

pipe from which dried particles are absorbed and for which the literature did not provide details. The geometry presented in Fig.6.4 was able to reproduce the inlet data-setup in Table 6.1 and to validate the velocity and temperature profiles along some radial sections available in literature [2]. Figures 6.5 and 6.6 show some details regarding the critical inlet and outlet parts of the all geometry.

Inlet Air		
Air inlet temperature	468	(K)
Air mass flow rate	0.336	(kg/s)
Air axial velocity	7.5	(m/s)
Air radial velocity	-5.25	(m/s)
Air total velocity	9.15	(m/s)
Outlet Condition		
Outflow and reference at outlet	-100	(Pa)
Turbulence inlet condition		
Turbulence k-value	0.027	(m^2/s^2)
Turbulence ϵ-value	0.37	(m^2/s^3)
Liquid spray from nozzle		
Liquid feed rate (spray rate)	0.0139	(kg/s)
Feed Temperature	300	(K)
Spray angle	76	(deg)
Minimum droplet diameter	10	(μm)
Maximum droplet diameter	138.0	(μm)
Average droplet diameter	70.5	(μm)
Droplet velocity at nozzle exit	59	(m/s)
Rosin-Rammler parameter	2.05	
Chamber wall conditions		
Chamber wall thickness	0.002	(m)
Wall material	Steel	
Overall wall-heat transfer coefficient	3.5	(W/m^2K)
Air temperature outside wall	300	(K)
Interaction between wall and droplet	Escape	

Table 6.1: *Boundary Condition from [2].*

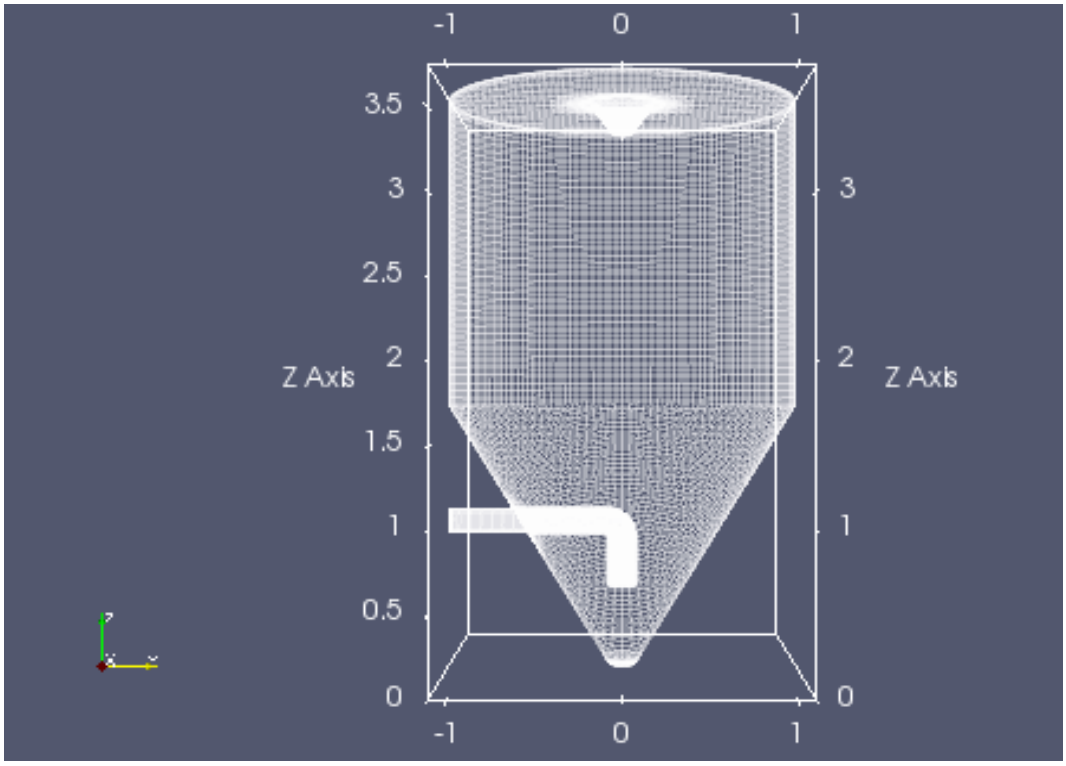


Figure 6.4: *Overall geometry.*

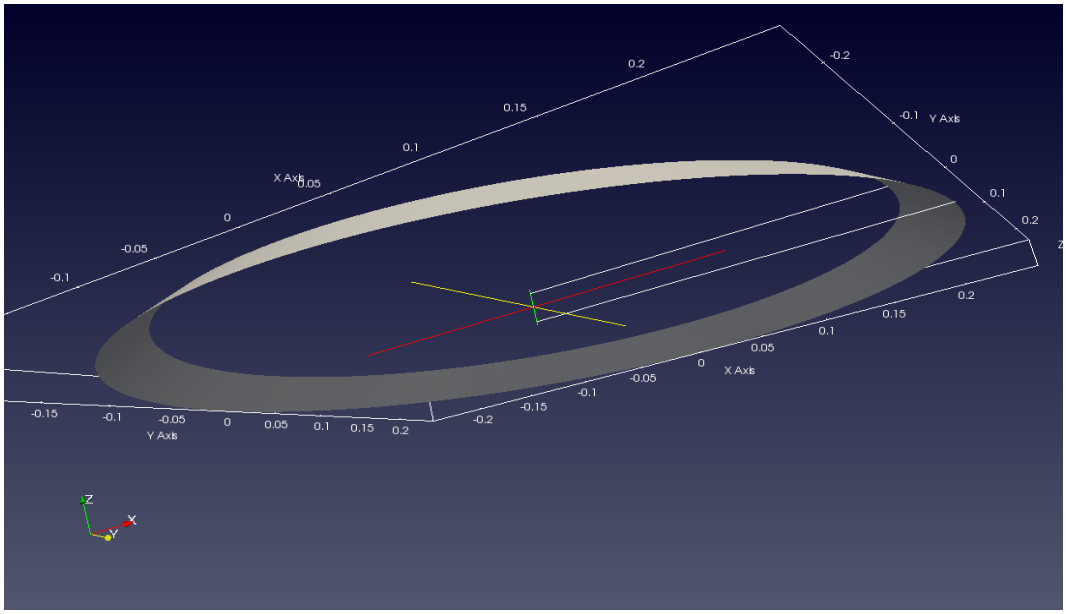


Figure 6.5: *Air inlet patch detail.*

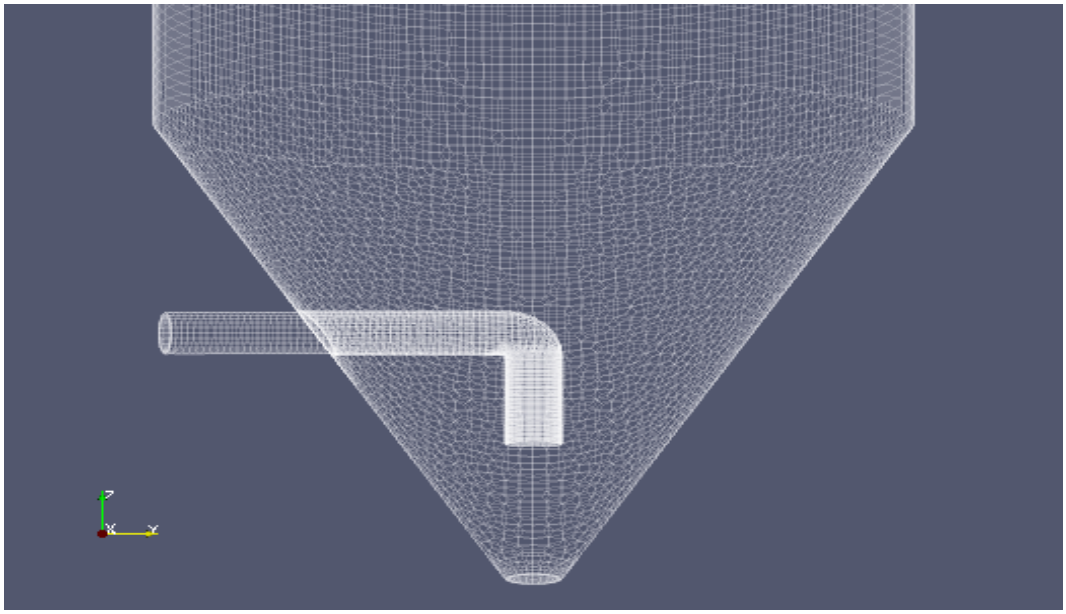


Figure 6.6: *Outlet pipe detail.*

Since this work, regardind the CFD , deals with an implementation in OpenFOAM® of an evaporation model, we decided to use a simplified geometry that allowed us to manipulate the case study with more simplicity. Indeed this geometry is easier to construct, to mesh and therefore encourages reproducibility by users. The modified geometry is a cylinder that respects the proportions of the original geometry and it is presented in Fig.(6.8). Please note that the reference system is such that the z axis is upward oriented.

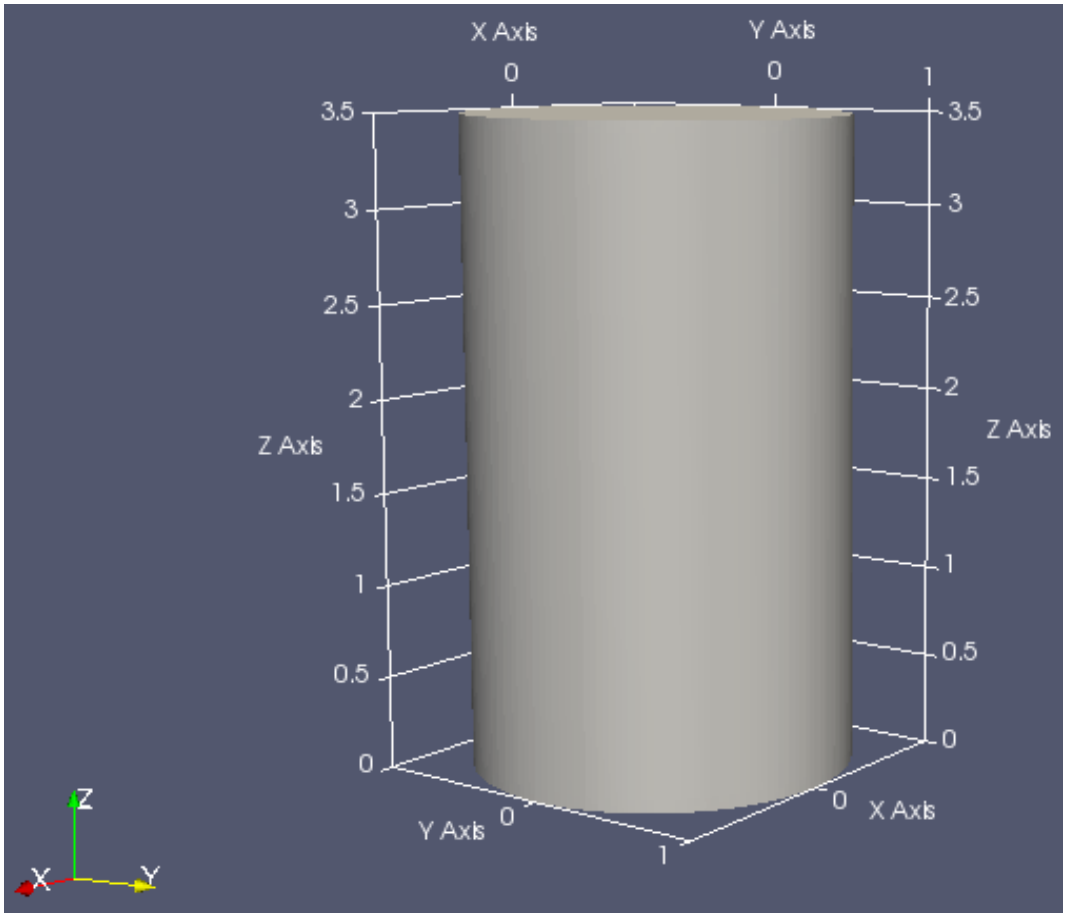


Figure 6.7: *Modified geometry.*

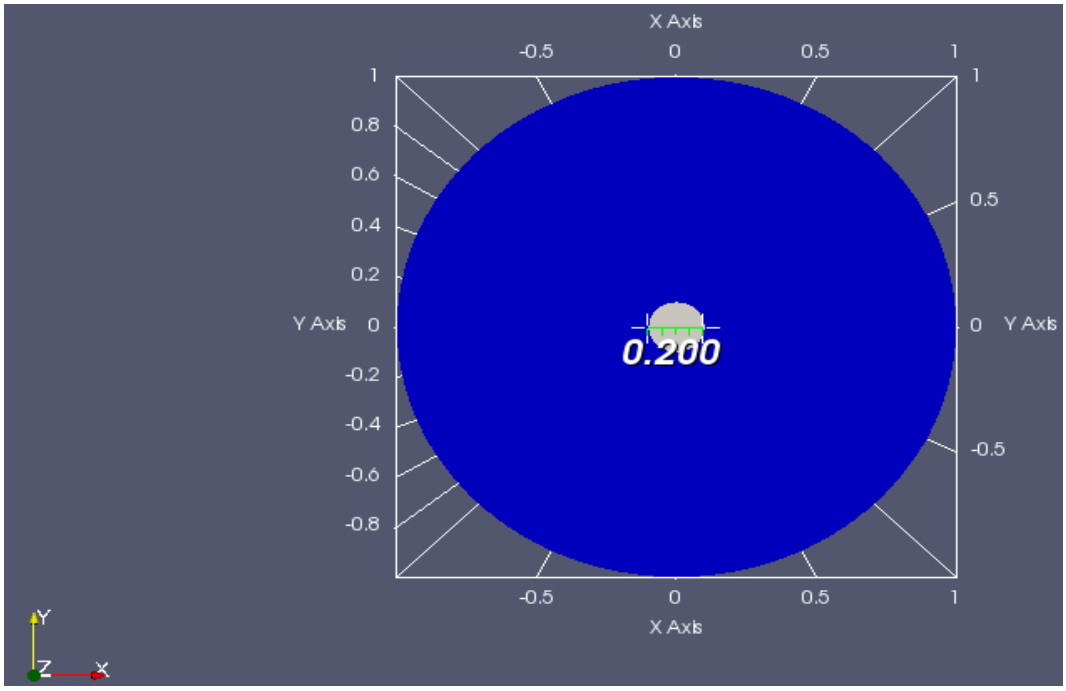


Figure 6.8: *Inlet patch detail.*

The dimensions of the geometry are listed in the table below:

Cylinder	Heigth= 3.5 $[m]$
	Diameter= 2 $[m]$
Inlet	Diameter= 0.2 $[m]$
Outlet	Diameter= 0.2 $[m]$

Table 6.2: *Modified geometry dimensions.*

6.0.1 Case with only the flow

The case concerning only the air flow is here presented. The objective is to compute a statistical steady developed flow that will be the basis for further simulations.

The solver: *BuoyantPimpleFoam*

As described within the source code of the solver (freely distributed with the download of OpenFOAM®), this is a transient solver for buoyant, turbulent flow of compressible fluids for ventilation and heat transfer. This solver, in addition to the mass conservation and momentum conservation equations in the compressible form (see Chapter 2 for details), also solves the energy equation in terms of enthalpy or internal energy. Therefore, before running the case, we need to define the thermodynamical properties of the working fluid and the temperature field. As a remark about the implementation in OpenFOAM® of the momentum conservation equation, the pressure gradient and gravity force terms are rearranged in the following form:

$$\begin{aligned} -\nabla p + \rho \boldsymbol{g} &= -\nabla(p_{rgh} + \rho \boldsymbol{g} \cdot \boldsymbol{r}) + \rho \boldsymbol{g} \\ &= -\nabla p_{rgh} - (\boldsymbol{g} \cdot \boldsymbol{r}) \nabla \rho \end{aligned} \tag{6.1}$$

Where $p_{rgh} = p - \rho \boldsymbol{g} \cdot \boldsymbol{r}$ and $\boldsymbol{r}$ is the position vector. The name of the solver suggests that the *PIMPLE* algorithm is used. *PIMPLE* is one of the pressure based solver for Navier-Stokes equations implemented in OpenFOAM®. It is an hybrid between the *SIMPLE* (Semi-Implicit Method for Pressure-Linked Equations) and *PISO* (Pressure Implicit with Splitting Operators) and is formulated for very large timesteps and pseudo-transient simulations [5]. The temporal dicretization scheme used for this time dependent simulation is the first order accurate Euler implicit method with adjustable time-step. Implicit numerical methods are unconditionally stable hence

the CFL number does not have to be bounded under certain low values. However using Euler implicit method doesn't mean that a time step of any size can be chosen. The timestep must be chosen in such a way that it resolves the time-dependent features, and it maintains the solver stability. The solver uses the $URANS$ equations with a $k-\omega$ SST turbulence model and the CFL could be set greater than 1 since all turbulence scales are modelled and only the transient behaviour of the mean flow can be observed. However the maximum CFL number in this simulation was set equal to 1 in order to gain stability and to be able to capture the very transient behaviour of this flux. Moreover, this solver requires the dictionary *thermophysicalProperties* that contains the definition of the physical properties of the working fluid. Within the subdictionary *thermoType*, the thermophysical models is specified. Thermophysical models are concerned with energy, heat and physical properties and they are constructed in OpenFOAM® as a pressure-temperature (p-T) system from which other properties are computed. OpenFOAM® includes a large set of precompiled combinations of modelling, built within the code using C++ templates. This coding approach assembles thermophysical modelling packages beginning with the equation of state and then adding more layers of thermophysical modelling that derive properties from the previous layer(s) [29]. The *thermoType* model used in this thesis is described in the following list:

- *heRhoThermo*: a thermophysical model based on density ρ.
- *const*: a transport model which assumes a constant dynamic viscosity μ and a Prandtl number $Pr = \frac{c_p \mu}{k}$.
- *hConst*: a thermodynamic model which assumes a constant c_p.
- *perfectGas*: it concerns the equation of state of the working fluid. In this case the perfect gas equation is used to compute the density field ρ.
- *sensibleEnthalpy*: it selects the energy variable (enthalpy in this case). The word sensible means that in the (sensible) energy heat of formation is not included

In this configuration, the calculated flow physical variables are the turbulent kinetic energy k, the specific dissipation ω, the pressure p, the dynamic pressure p_{rgh}, the velocity components U_x, U_y, U_z, the temperature T, the turbulent viscosity ν_T (that is not a physical property), the effective turbulent thermal diffusivity α_T (that is not a physical property). The boundary and initial conditions for all these calculated variables are presented in the next section.

The linear solver, already implemented in OpenFOAM®, for the solution of the previous variables are really standard. Depending on if the matrix is symmetric (e.g pressure) or asymmetric (e.g velocity, turbulent kinetic energy, specific dissipation, enthalpy ...) the solvers are:

- GAMG (geometric-algebraic multi-grid) with the smoother Gauss Seidel for symmetric matrices with tolerance 10^{-8}
- SmoothSolver with the smoother symGaussSeidel for symmetric matrices with tolerance 10^{-8}

Initial and Boundary conditions

The geometry is given on STL format and this file is composed by 5 patches:

- *Inlet*: This is the patch through which the hot air enters the computational domain;
- *Outlet*: This is the patch through which the hot air goes out the computational domain;
- *Wall_lower*: This patch is the inferior base of the cylinder with an hole corresponding to the *Outlet* patch;
- *Wall_upper*: This patch is the superior base of the cylinder with an hole corresponding to the *Inlet* patch;
- *Wall_side*: This is the patch that includes the lateral surface of the cylinder;

According to this division inside the STL, Table 6.3 summarizes the initial and boundary conditions used for this simulation.

The *zeroGradient* boundary condition simply extrapolates the quantity to the patch from the nearest cell value by setting the gradient equal to zero for the variable of interest in the direction perpendicular to the boundary. What concerns k, ω, ν_T, α_T the turbulent wall function *kqRWallFunction*, *omegaWallFunction*, *nutkWallFunction*, *alphatWallFunction* are respectively the wall functions for each variable regarding the $k\omega$ SST turbulent model. These wall functions impose a wall-value to each variable; for example the value of the specific dissipation ω becomes:

$$\omega_{wall} = 10\frac{6\nu}{\beta y^2}, \tag{6.2}$$

where $\beta = 0.075$, ν is the kinematic viscosity and y is the distance of the first cell center normal to the wall.

It should be noted that for all the wall patches the *WallHeatTransfer* condition is applied with a value of *alphaWall* very low in order to simulate an adiabatic wall condition. This is a small variation from the conditions described in Tab.6.1 but this is a good approximation for this case. The Unit Measure of *alphaWall* in OpenFOAM® is $[\frac{W}{m^2}]$ that is a bit unusual for this physical properties therefore it is likely to impose a very low value of heat flux per unit area through the wall patches.

Concerning the dynamic pressure p_{rgh} the boundary condition applied to the wall is *fixedFluxPressure*. This boundary condition adjusts the pressure gradient such that the flux on the boundary is that specified by the velocity boundary condition. At the outlet a depression of $100[Pa]$ satisfies the condition for the original geometry.

The air inlet temperature $T = 470$ $[K]$ and the mass flow rate of 0.4 $[\frac{Kg}{s}]$ comes from literature values in Tab.6.1.

Table 6.3: *Initial and boundary conditions.*

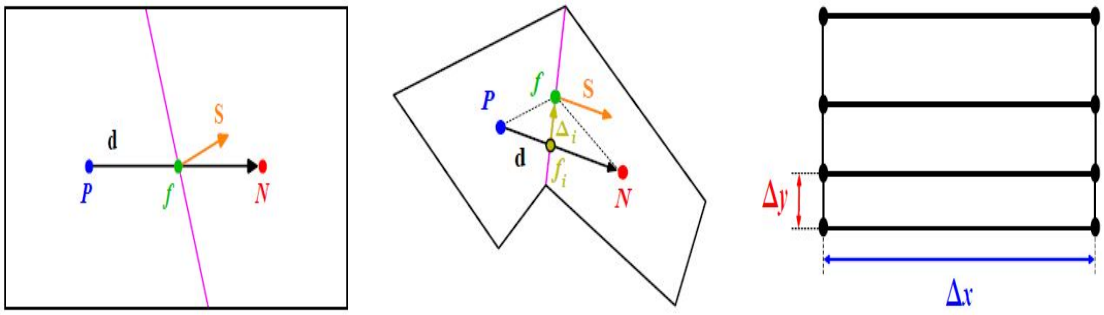
	Walls	Inlet	Outlet
Temperature: **T**	-*wallHeatTransfer* initial value: uniform 300 $[K]$ $T_{ref} = 300$ $[K]$ $\alpha_{Wall} = 0.001$ $[\frac{W}{m^2}]$	-*fixed value* value: uniform 468$[K]$	-*zeroGradient*
Thermal diffusivity: $\alpha_{\mathbf{T}}$	-*alphatWallFunction* $Pr_T = 0.85$ initial value: uniform 0 $[\frac{Kg}{ms}]$	-*calculated* initial value: uniform 0 $[\frac{Kg}{ms}]$	-*zeroGradient*
Turbulent kinetic energy: **k**	-*kqRWallFunction* initial value: uniform 0.027 $[\frac{m^2}{s^2}]$	-*fixedValue* initial value: uniform 0.027 $[\frac{m^2}{s^2}]$	-*zeroGradient*
Specific dissipation: ω	-*omegaWallFunction* initial value: uniform 4.28 $[\frac{1}{s}]$	-*fixedValue* initial value: uniform 4.28 $[\frac{1}{s}]$	-*zeroGradient*
Turbulent viscosity: $\nu_{\mathbf{T}}$	-*nutkWallFunction* initial value: uniform 0 $[\frac{Kg}{ms}]$	-*calculated* initial value: uniform 0 $[\frac{Kg}{ms}]$	-*calculated* initial value: uniform 0 $[\frac{Kg}{ms}]$
Velocity: **U**	-*fixedValue* initial value: uniform 0 $[\frac{m}{s}]$	-*flowRateInletVelocity* mass flow rate 0.4 $[\frac{Kg}{s}]$ $\rho_{inlet} = 1$ $[\frac{K}{m^3}]$	-*zeroGradient*
Pressure: p_{rgh}	-*fixedFluxPressure* initial value: uniform 101325 $[Pa]$	-*zeroGradient*	-*fixedValue* initial value: uniform 101225 $[Pa]$
Pressure: p	-*calculated* initial value: uniform 101225 $[Pa]$	-*calculated* initial value: uniform 101225 $[Pa]$	-*zeroGradient*

Mesh

Since the geometry has no particular features, the mesh process is quite fast and it is easy to get a mesh with very good quality. The open source mesher utilized is *cfMesh* which is a cross-platform library for automatic mesh generation that is built on top of OpenFOAM® [14]. In particular with the utility *cartesianMesh* an unstructered mesh is obtained, consisting of predominantly hexahedral cells with polyhedra in the transition regions between the cells of different size. *cfMesh* doesn't require a background mesh and seemed to be faster than the other utilities that come with OpenFOAM®. The mesh quality is controlled by three main parameters that can be extrapolated using the OpenFOAM® utility *checkMesh*:

- *Mesh orthogonality*: This parameter is related to angular deviation of the cell face normal vector from the vector connecting two consecutive cell centres. Usually this parameter is kept below 70.
- *Mesh skewness*: Skewness is the deviation of the vector that connects two cell centres from the face centres. Usually this parameter is kept below 8.
- *Aspect ratio*: Aspect ratio AR is the ratio between the longest side Δx and the shortest side Δy of the cell. Large AR are good if gradients in the largest direction are small.

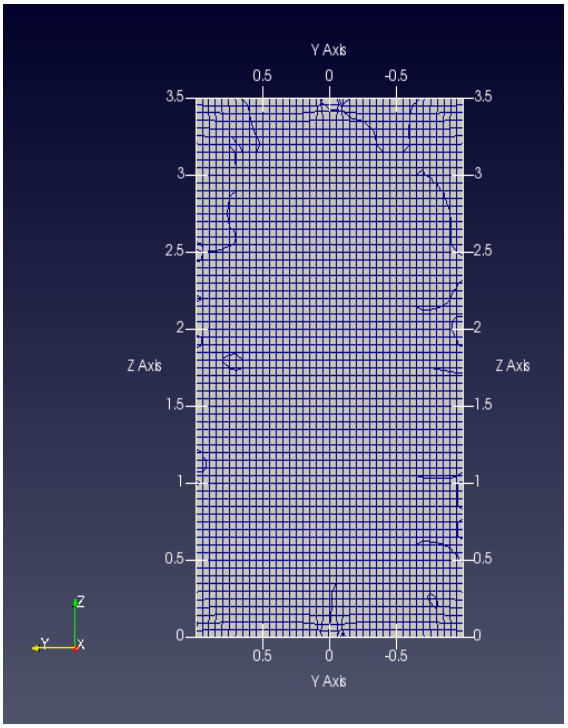
All these mesh features can be viewed in Fig.6.9.



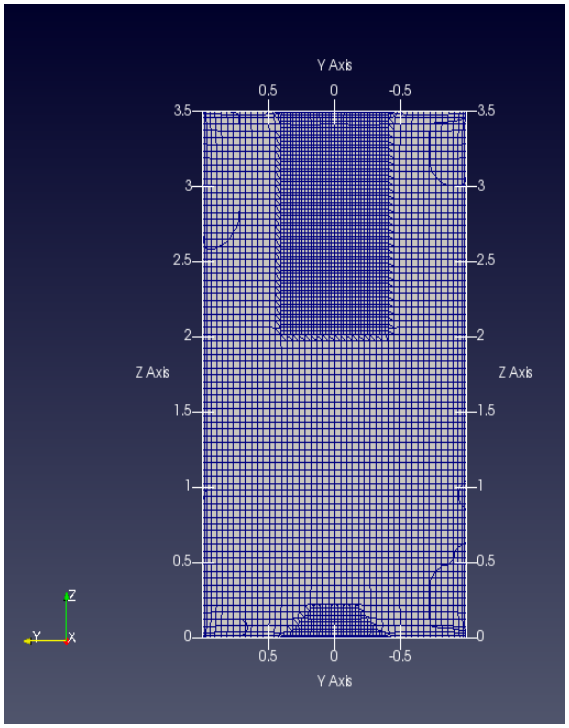
(a) Mesh orthogonality (b) Mesh skewness (c) Aspect ratio

Figure 6.9: *Main mesh quality parameters.*

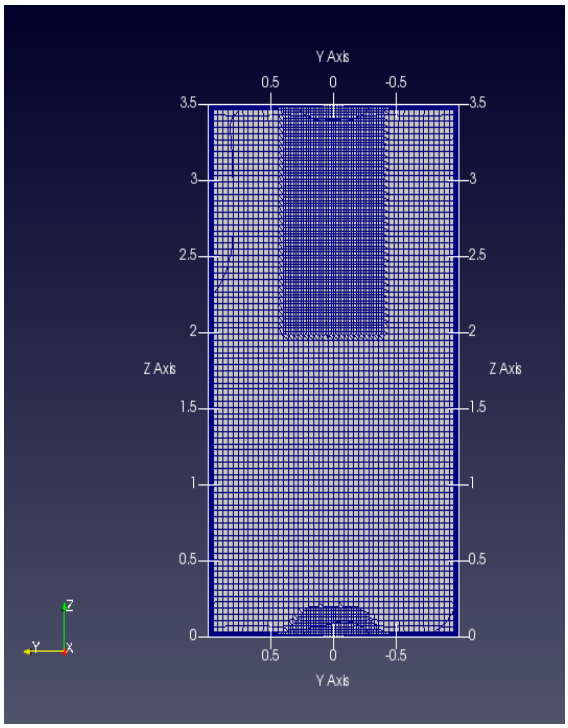
Three different meshes have been tested in order to perform a convergence study. Figure 6.10 shows a slice of all meshes and it is possible to see an increasing refinement level near the inlet and outlet region of the computational domain.



(a) Coarse mesh



(b) Fine mesh



(c) Very fine mesh

Figure 6.10: *Meshes with different refinement levels.*

The main features of all these meshes are reported in Tab.6.4

Table 6.4: *Mesh features.*

	Coarse mesh	**Fine mesh**	**Very fine mesh**
Cells	93184	214804	457084
HexaHedra	92176	209440	448648
Prisms	224	672	2048
Pyramids	560	840	760
Tetrahedra	224	336	304
Polyhedra	0	3516	5324
Non-orthogonality	Max= 35.6423 Average= 2.5941	Max= 37.886 Average= 3.5087	Max= 53.8612 Average= 4.2559
Max skewness	0.503481	0.925743	1.60281
Max *AR*	6.650601	7.20404	27.3225
Number of layers	0	3	5
y+ *Wall lower(average)*	346.627	112.479	5.111
y+ *Wall upper(average)*	119.939	15.302	0.171
y+ *Wall side (average)*	142.605	41.828	2.177

$y+$ values are only indicatives since we are not interested in resolving the boundary layer but rather in the mixing in the internal region of the cylinder. However the refinement purpose was to increase the accuracy of the solutiohn and the scalable wall functions for the $k-\omega\ SST$ turbulence model ensures good results independently from the grid spacing.

Chapter 7

Particles implementation in OpenFOAM

The model for the two-stage drying process previously described have been implemented and verified in OpenFOAM®. In particular the first stage has been verified with the evaporation model existing in *ReactingParcelFoam*. In order to test only the behaviour of the drying process the same velocity and temperature field have been used when running with *ReactingParcelFoam* and the newly developed version of the dryin process.

7.1 Background fluid flow field

The background fluid flow simulation has been run for a time long enough in order to let the initial transient develop and vanish. Figures 7.1 and 7.2 show the signal of temperature and of the vertical velocity component as a function of time. This signal comes from two points inside the computational domain which locations are shown in Fig.7.3; After 160 seconds it can be seen that a steady state condition is reached since the there are no more oscillations of the plotted variables in time. However we decided to run the simulation until 170 seconds and to use this final solution as starting point for all further simulations with particles.

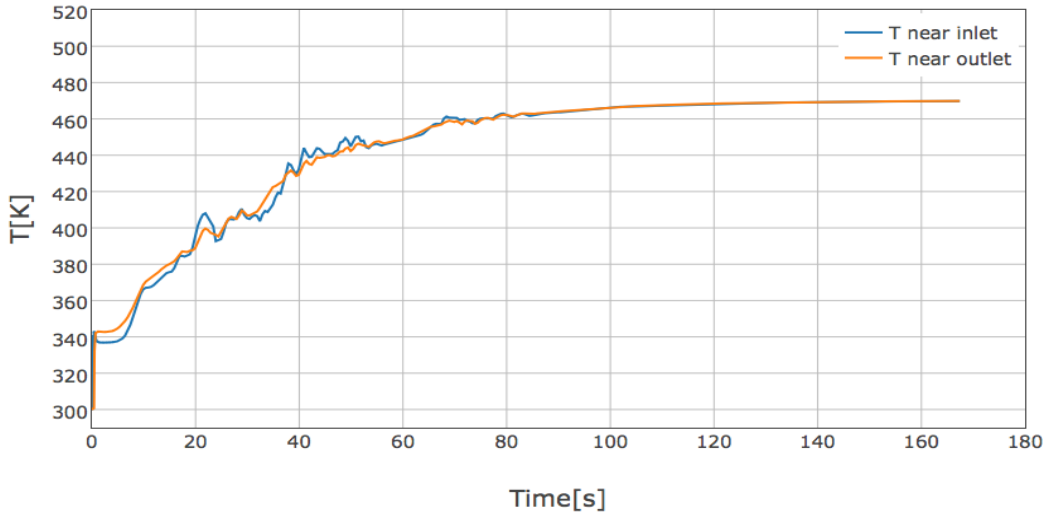


Figure 7.1: *Temperature as a function of time extracted at the two positions shown in Fig.7.3.*

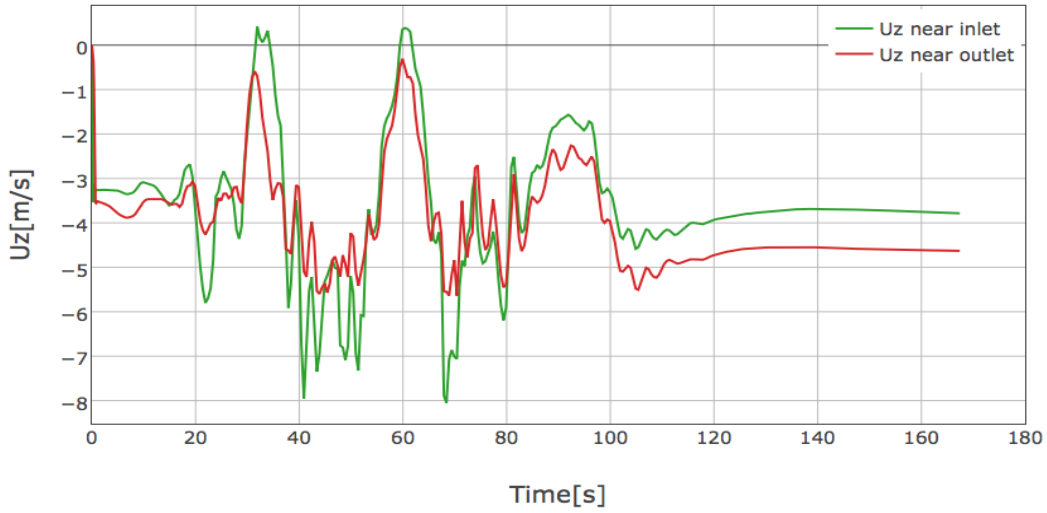


Figure 7.2: *Velocity component in the z-direction as a function of time extracted at the two positions shown in Fig.7.3.*

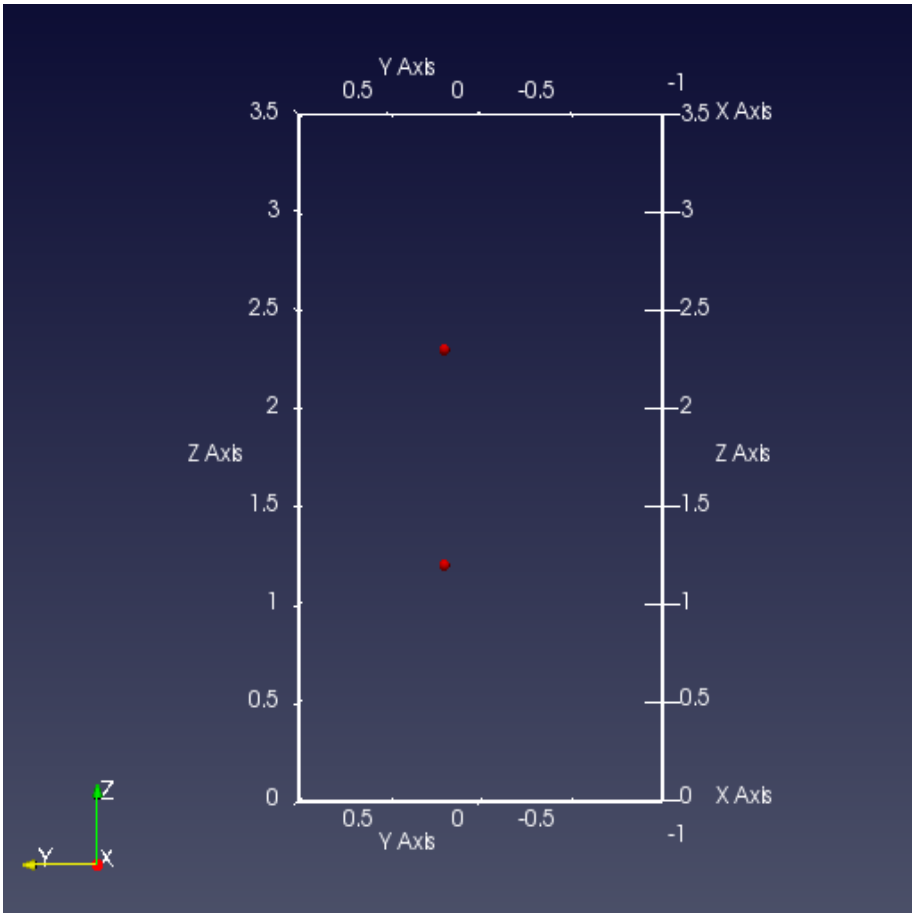
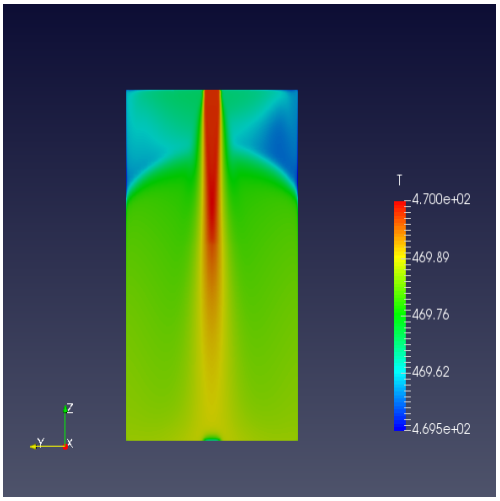
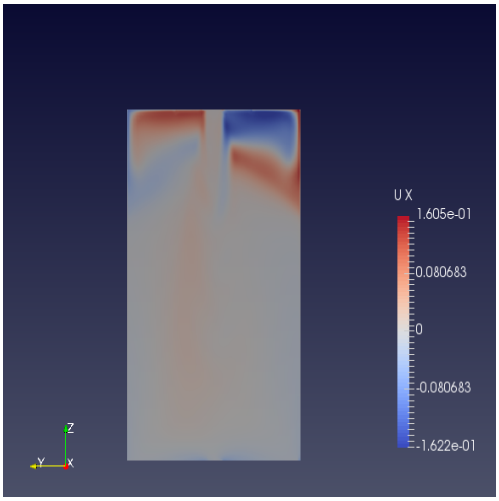


Figure 7.3: *Location of two points inside the domain used for the acquisition of velocity and temperature temporal signal.*

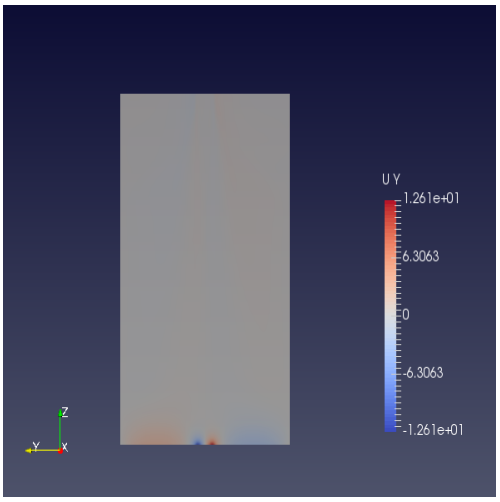
A better view of the fluid motion and of the thermodynamic inside the cylinder can be obtained through the analysis of the velocity and temperature fields plotted in Fig 7.4 for solution at $t = 170$ $[s]$. Despite the colour visualisation of the temperature field, it is possible to say that all the domain has a temperature of about 470 $[K]$. Moreover the analysis of velocity components show that there is considerable recirculation zone near the bottom region that brings the air up almost until the top.



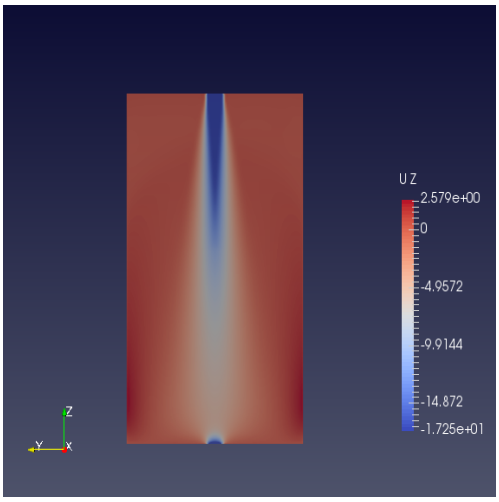
(a) Temperature field.



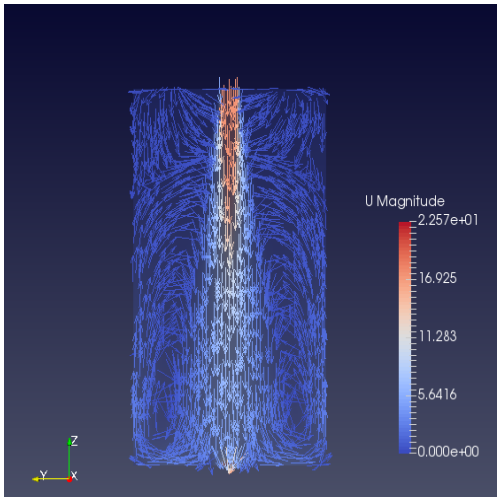
(b) x velocity component.



(c) y velocity component.



(d) z velocity component.



(e) Magnitude of velocity.

Figure 7.4: *Eulerian fields for the final solution at* $t = 170s$.

Fig.7.5 shows the plot of the final residuals for velocity components and dynamic pressure, ensuring that each time step the convergence criteria (10^{-8} for the plotted variables is reached.

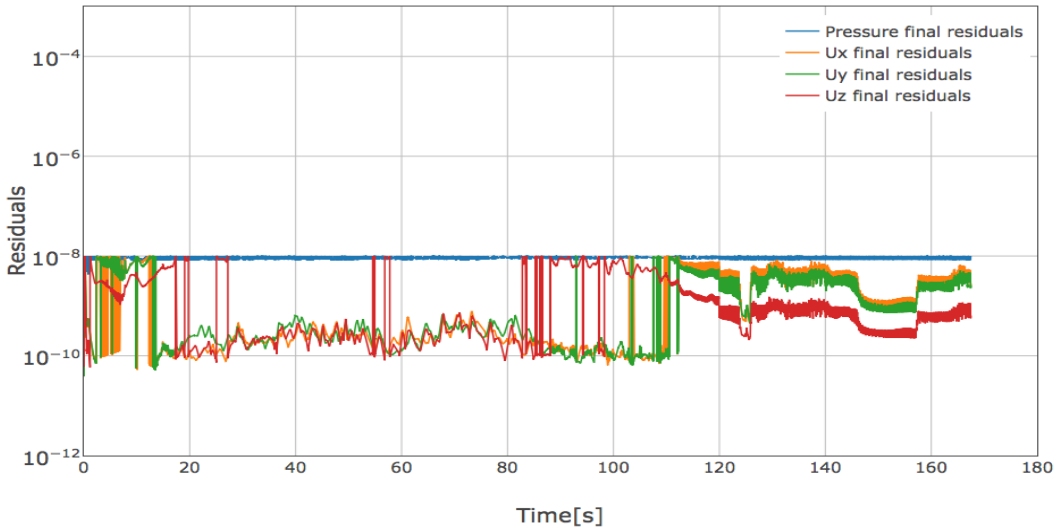


Figure 7.5: *Initial and final residuals: convergence criteria are reached through final residuals.*

At this point particles are injected in the computational domain, and their properties are computed through two different Lagrangian solvers:

- *ReactingParcelFoam*: This is a transient solver for compressible, turbulent flow with a reacting, multiphase particles cloud [5].
- *MA_BuoyantKinematicParcelFoam*: This is a new transient solver that uses the model for the drying kinetic described in chapter 3 and in chapter 5.

Let us underline that this solver continues to follow the evolution of the background flow field, hence Eulerian fields are still computed each time step through the use of the *PIMPLE* algorithm. The boundary conditions and the simulation set up for these Eulerian fields are always the same described in the previous Chapter.

7.2 OpenFOAM® solver: *ReactingParcelFoam*

ReactingParcelFoam is a Lagrangian solver that is able to model several physical processes like combustion, radiation, chemistry reactions and with the inclusion of

a Lagrangian cloud of particles. The purpose within this thesis is to use this solver to model the evaporation of pure liquid droplets and compare the results with the new two-stage drying process solver. Also this solver requires a *thermophysicalProperties* dictionary that is almost the same used for the previous simulation with *buoyantPimpleFoam* but with some modifications regarding the transport and thermodynamic models which take into account for temperature dependencies of the air dynamic viscosity and specific heat. However, in this case another dictionary named *reactingCloudProperties* for evaporating droplets is required. Inside this dictionary it possible to find the core of the Lagrangian set up and the parameters that are significant for this thesis are shown in the list below:

- *coupling*: This keyword enables or not the coupling between the continuous and the dispersed phase. For this work the coupling is always set to false and this means that a one-way coupling is performed.

- *interpolationSchemes*: This subdictionary refers to the interpolation scheme between cell centres values and particles-droplets positions. The option *cell*, that assumes cell-centre values constant over the cell, has been used for all variables except the velocity U for which the *cellPoint* option has been chosen. (*cellPoint* concerns with a linear weighted interpolation using cell values).

- *integrationSchemes*: This is related to the integration with time of particles velocity and temperature. By default these are setted to Euler for velocity and analytical for temperature.

- *subModels*: In this section it is possible to specify a number of models but the only relevant for this work are those about the forces applied to the particles (*sphereDrag*), the injection inside the domain (*coneInjectionModel*) and the particles interaction with wall (*stick*).

  - The *sphereDrag* model imposes a drag coefficient depending on Reynolds particle number.

$$\begin{cases} C_D = 0.424Re & Re > 1000 \\ C_D = 24(1 + \frac{1}{6}Re^{\frac{2}{3}}) & Re < 1000 \end{cases}$$

  - *coneInjectionModel*: The *coneInjectionModel* is a multi-point injection model in which users specifies the time of start of injection, the list of injector positions and directions (along injection axes), parcel velocities,

inner and outer spray cone angles, parcel diameters also obtained by distribution model and number of parcel to inject per injector. Please, note that parcel means a group of particles, but in this thesis they are imposed as the same. The injections parameters have been taken from literature (see Tab.6.1) and adapted for the application with this simplified geometry. The table below summarizes the choices about the main injection parameters:

Table 7.1: *Injection parameters.*

Start of injection (SOI)	0
Duration	2 s
parcelsPerInjector	4000
Umag	59 $\frac{m}{s}$
positionAxis	$(0, 0, 3.45)$ $(0, 0, -1)$
thetaInner	0
thetaOuter	76
sizeDistribution	type: fixedValue fixedValueDistribution: value $70\mu m$
heatTransferModel	*RanzMarshall*
phaseChangeModel	*liquidEvaporation*

– *stick*: For all wall patches it has been set a stick boundary condition that assigns a zero velocity to the particles that impact on the wall.

As a final remark the interactions, such as the collisions between the particles in the dispersed phase, are not taken into account in any cases tested in this thesis.

7.3 *MA_BuoyantKinematicParcelFoam*

This solver is a new OpenFOAM® application implemented for this thesis with the purpose to add to the already existing *BuoyantPimpleFoam* solver for the solution of the Eulerian fields, the evolution of a Lagrangian cloud of particles. This allowed us to include the specific evaporation model described in chapter 2. Despite the great importance of knowing the structure of the code to implement a new model, this is not of great interest from a physical point of view and we will avoid its description. On the other hand, it is needed to describe how the code works to understand

where our evaporation model (see section 4.1.1) is implemented. OpenFOAM® handles the simulation of the particles dynamics reducing as much as possible the interactions with the solution of the fluid equations: the advantage of this choice is that the kind of the underlying fluid simulation has a very little influence on how the particles-classes work. Let us consider now the general case of a time dependent simulation, we can identify these different steps:

- the governing equations of the fluid are solved in a standard way with a given *Eulerian* time step Δt.
- here the Lagrangian loop starts. To evaluate the new particle position the given Eulerian time step is divided in a certain number of Lagrangian time steps δt. For each Lagrangian time step an interpolation of the Eulerian fields (stored in cell centres) is made in the position of the particle. Then, considering the list of forces acting on the particle, the particle velocity $\boldsymbol{u}_p$ is evaluated.
- knowing the particle velocity and a given Lagrangian time step an estimation is made for the new particle position $\boldsymbol{x}^*$.

$$\boldsymbol{x}^* = \boldsymbol{x}(t) + \delta t \boldsymbol{u}_p(t) \tag{7.1}$$

  Here the code checks if the particle crosses a cell boundary. If not, the new particle position is actually the previous estimation and the Lagrangian time step is definitely fixed.

$$\boldsymbol{x}(t + \delta t) = \boldsymbol{x}^* \tag{7.2}$$

  If yes, the code performs a series of additional checks (for examples if it is a domain boundary or a partition between different parallel domains) and the effective Lagrangian time step is a fraction of the initial one.

$$\begin{aligned} &\delta t \rightarrow \delta t^* \\ &\boldsymbol{x}^* = \boldsymbol{x}_n + \delta t^* \boldsymbol{u}_p \end{aligned} \tag{7.3}$$

- for the new position, the new particle velocity is computed according to the steps described at the beginning of the loop (interpolation of Eulerian fields, evaluation of forces acting on particle).
- until the particle has moved for a time equal to a Eulerian time step (that is the sum of all Lagrangian time steps) the loop is repeated.

The evaporation model is integrated within the Lagrangian loop, precisely after the interpolation of the Eulerian fields to the particle position, when the Lagrangian timestep is fixed. This timestep is used for the time integration of the droplets energy and radius shrinking equations (see chapter 3 for further details) resulting in the evaluation of new droplets diameter, temperature and moisture content. The Lagrangian loop and, the point in which the new evaporation model is included, is shown in Fig.7.6.

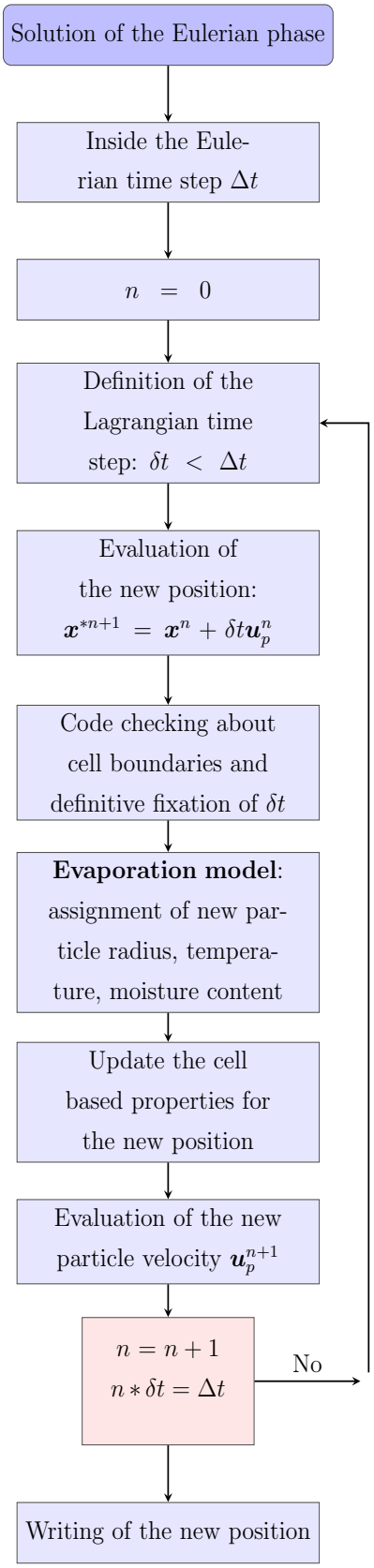


Figure 7.6: *Flow chart for the Lagrangian loop.*

Part VI

Results

Chapter 8

Results

8.1 First drying stage

As described in Chapter 5, three different models for the first drying stage of liquid drops with insoluble solids have been tested. In this section we present the results obtained from these models obtained with a Python code and a comparison with experimental data. Details about the experimental set up and the model implemented in literature can be found in [26]. There is a main difference from Nesic's model: unlike us the author solves the diffusion equation for the solid species in order to evaluate a solid concentration profile inside the droplet and to have a good estimation for the critical moisture content for which a solid shell begins to build. As pointed by [3], for the description of the insoluble solid phase processes it would be better to use the notion of a number density distribution $n(t, r)$, which describes the number of nanoparticles in an infinitesimal volume. This number is affected by different processes such as the drying of the droplet, mass transfer of solid particles (diffusion), and particle formation in the interior of the droplet. All these aspects are not taken into account in our model and the first stage finishes when a critical moisture content calculated from a simple expression based on solid porosity is reached. The insoluble solid dispersed in droplets is the silica SiO_2 that experimentally showed evaporation rates and temperature levels very similar to water. However silica at a concentration of 40% turns into a gel structure that stops all internal recirculation flows induced by the air drag force. This results in an slight increase in temperature during the first evaporation stage that our model is not able to capture. The are a lot of thermophysical properties for air, water and solid and some of which are not easy to extrapolate from literature to reproduce the same experimental set up. The main choices for this model are listed in table 8.1:

Droplet		
Initial droplet radius	1	mm
Initial droplet mass	4.3	mg
Initial solid content	0.3	
Air		
Velocity	1.4	$\frac{m}{s}$
Temperature	451	K
Density	1	$\frac{Kg}{m^3}$
Thermal diffusivity	$36.94e^{-6}$	$\frac{m^2}{s}$
Kinematic viscosity	$2.5e^{-5}$	$\frac{m^2}{s}$
Relative humidity	0.4%	
Thermal conductivity	0.0326	$\frac{W}{mK}$
Water		
Specific heat	4187	$\frac{J}{KgK}$
Density	1000	$\frac{Kg}{m^3}$
Solid (silica)		
Specific heat	770	$\frac{J}{KgK}$
Density	939	$\frac{Kg}{m^3}$
Porosity	0.3	

Table 8.1: *First drying stage set up.*

8.1.1 Lumped

The main assumption in this model is that the droplet temperature is not a function of the radius and it can vary only with time. This approximation leads to the well known simplified lumped model. Indeed there is no possibility to have information about the droplet centre temperature and this could be a problem for large drops for which it might exist a consistent difference. Evaporation is enabled from the beginning of the simulation and it can be seen that the equilibrium temperature is reached with great accuracy if compared with experimental data. Even if the lumped approximation should be rigorously applied only if the Biot number is less than 0.1, in this case with an initial Biot number of 0.14, the surface temperature results seem to be correct. Information about the Fourier number, that is the other

parameter that makes it possible to understand if the applicability of the lumped model is correct, will be given in the next two sections, where it will be clear that we are committing an error disregarding the temperature profiles within the droplet. The energy equation that governs the evolution in time of the droplet temperature is Eq.(3.90) and two different time integration schemes (first order implicit, second order Adam Bashforth explicit) have been tested for this equation; there are no significant differences between the two schemes and this is also confirmed by the green and red lines shown in Fig.8.1, which are pratically superimposed. However for the implicit algorithm (see flowchart in Fig.5.1), attention should be paid on the choice of the initial guess for droplet radius that must be proportionate to the time step in order to gain faster convergences, and of course, faster results. What concerns the droplet mass reduction there is a little difference from the experimental data and from Nesic's model, probably due to differences between the original experimental set up and the parameters in table 8.1. The shrinkage of the droplet diameter and the reduction of moisture content are plotted in Fig.8.3 and in Fig.8.4. The first evaporation stage ends when a critical moisture content, calculated from $X_{cr} = \frac{\rho_w}{\rho_s}\frac{\varepsilon}{1-\varepsilon}$, is reached. This model is the one chosen for the implementation in OpenFOAM® where, dealing with micron sized droplets, the lumped approximation is in the majority of cases correct, and where the injection of thousands drops, requires the simplest model to reduce the computational costs.

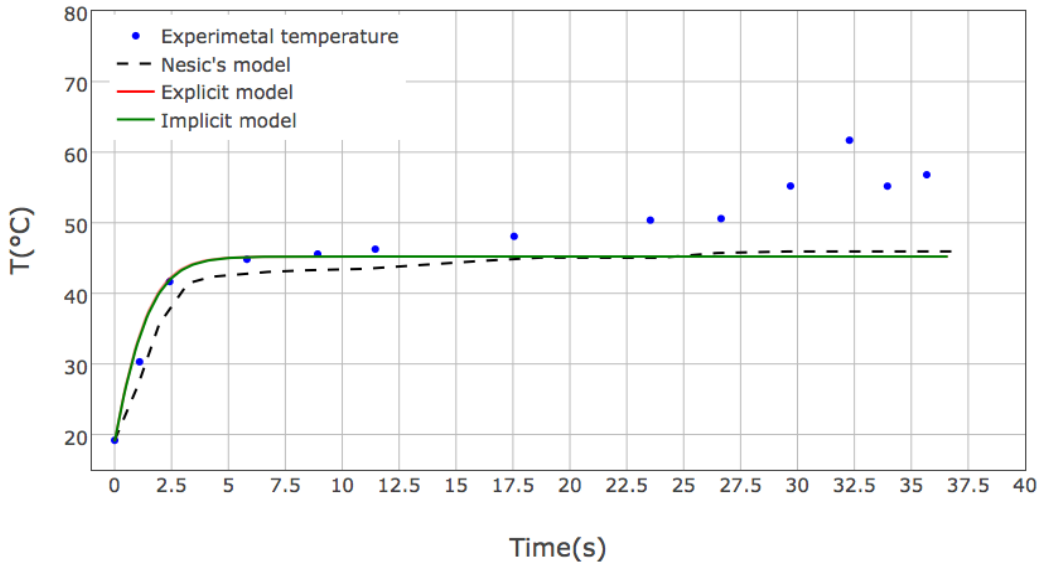


Figure 8.1: *Evolution in time of droplet temperature for the lumped model.*

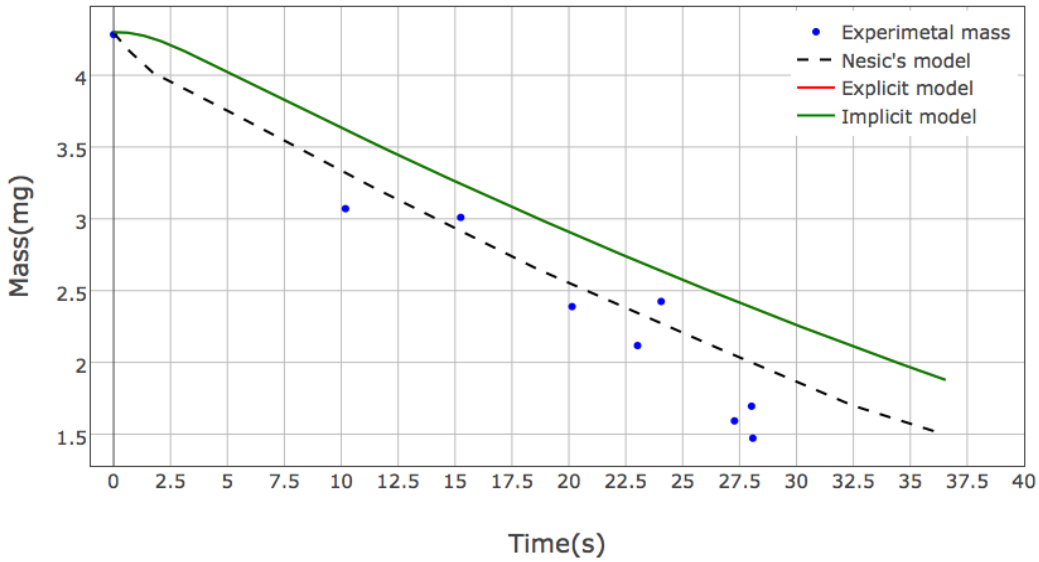


Figure 8.2: *Evolution in time of droplet mass for the lumped model.*

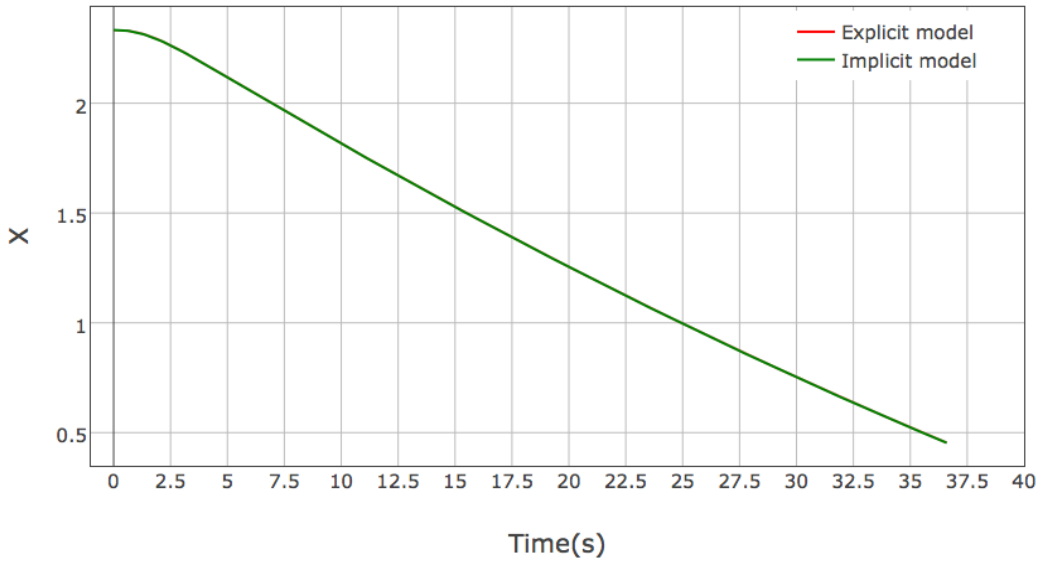


Figure 8.3: *Evolution in time of droplet moisture content for the lumped model.*

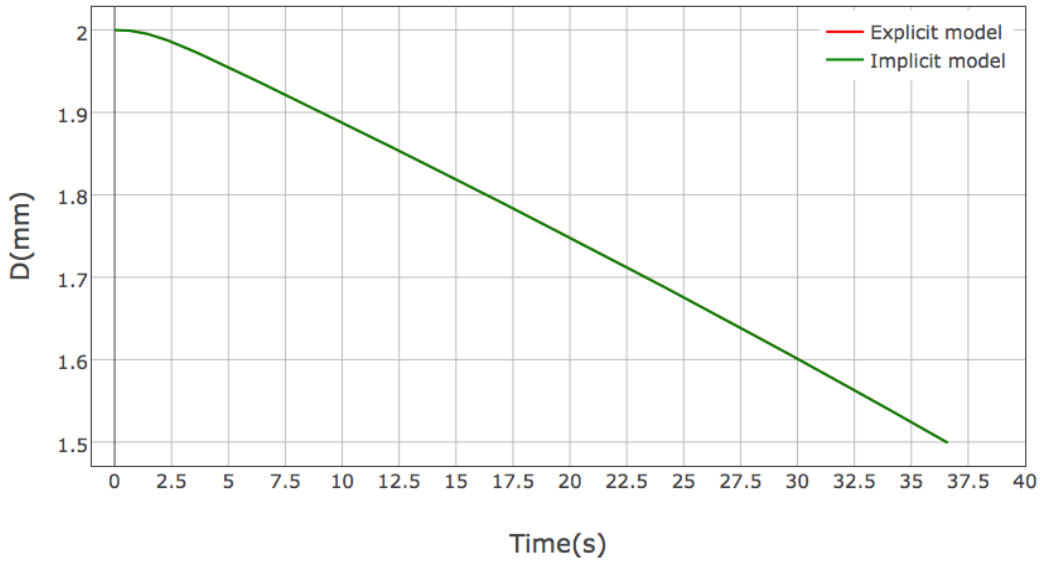


Figure 8.4: *Evolution in time of droplet diameter for the lumped model.*

8.1.2 Initial heating and uniform temperature approach

This model is based on two main assumptions:

- constant evaporation temperature;
- no evaporation during the initial heating period;

At the end of the initial heating period the calculated Fourier number is $Fo = 0.38$ and the Biot number is $Bi = 0.14$; hence it is expected a certain radial dependence of temperature. This can be observed in Fig.(8.5) and in Fig.(8.6) that show the evolution in time of the temperature within the droplet for two different time integration schemes of Eq.(5.13) (Implicit Euler method and Explicit Adam-Bashforth method, see section 4.1.2 for further details). It can be observed that there is a difference of about $5 - 6°C$ between the centre and the surface of the droplet, confirming that the previous model was incomplete because it disregarded the core temperature and it imposed a constant value for the whole drop. However, since there is no evaporation during the initial heating period, these values are overestimated and this will be observed in the next model. There are 50 grid points along the radius and the Δt is set equal to 0.001. The choice of Δt is related to the numerical stability of the explicit method, for which the time step can not be chosen independently from spatial discretization. The stability criteria for the second order Adam-Bashforth

method is not as simple as for traditional first order Euler explicit method and it is out of the scope of this work. Fig.8.7 also shows the analytical solution of Eq.(5.13) together with the two last profiles of Fig.8.5 and in Fig.8.6. A good agreement is seen between numerical and analytical solutions also for the implicit method, that is first order accurate. As a final remark about the analytical solution, it should be underlined that it is obtained through the one term approximation of an infinite series according to the procedure described in [4];

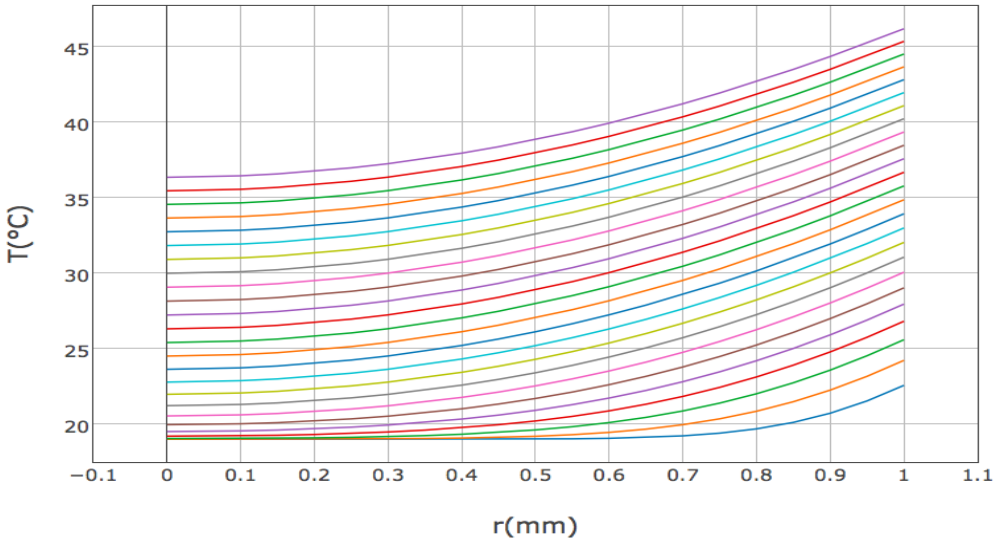


Figure 8.5: *Temperature profiles with the explicit Adam-Bashforth method.*

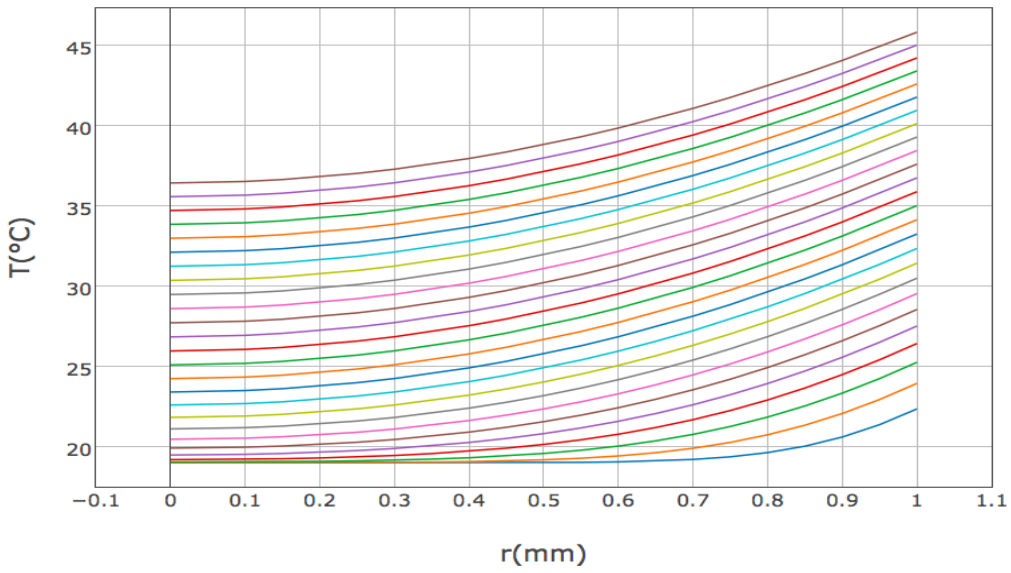


Figure 8.6: *Temperature profiles with the implicit Euler method.*

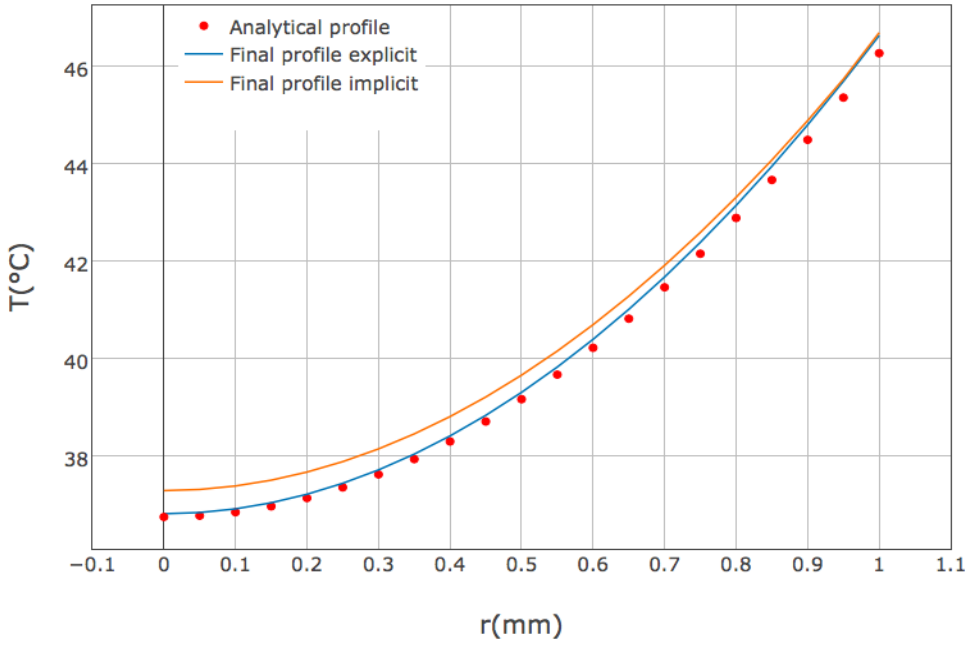


Figure 8.7: *Comparison between analytical solution and last profile for the explicit Adam-Bashforth method.*

Additional results for this model are presented in terms of evolution in time of surface temperature (Fig.8.8), mass (Fig.8.9), diameter (Fig.8.11) and moisture content (Fig.8.10). The effect of excluding evaporation during the initial heating period involves constant values for the mass, the diameter and the moisture content. It is also possible to note in Fig.8.8 that this assumption leads to the result that the evaporation temperature is reached by droplet surface after a shorter time than the experimental data; when evaporation temperature is reached, the evaporation is enabled within the model and it is possible to catch the diameter shrinkage and consequently the mass and moisture content reduction. Also in this case when the critical moisture content is reached the simulation ends. There are very small differences between the implicit and explicit model, and this results in two overlapping lines in Figs.8.8-8.10.

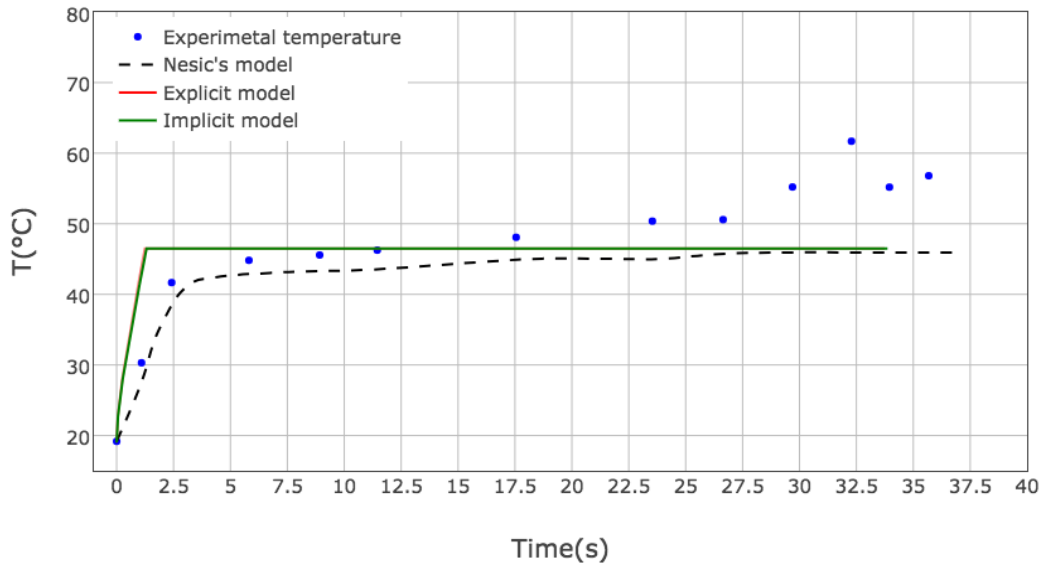


Figure 8.8: *Evolution in time of droplet temperature for the initial heating and uniform evaporation temperature model.*

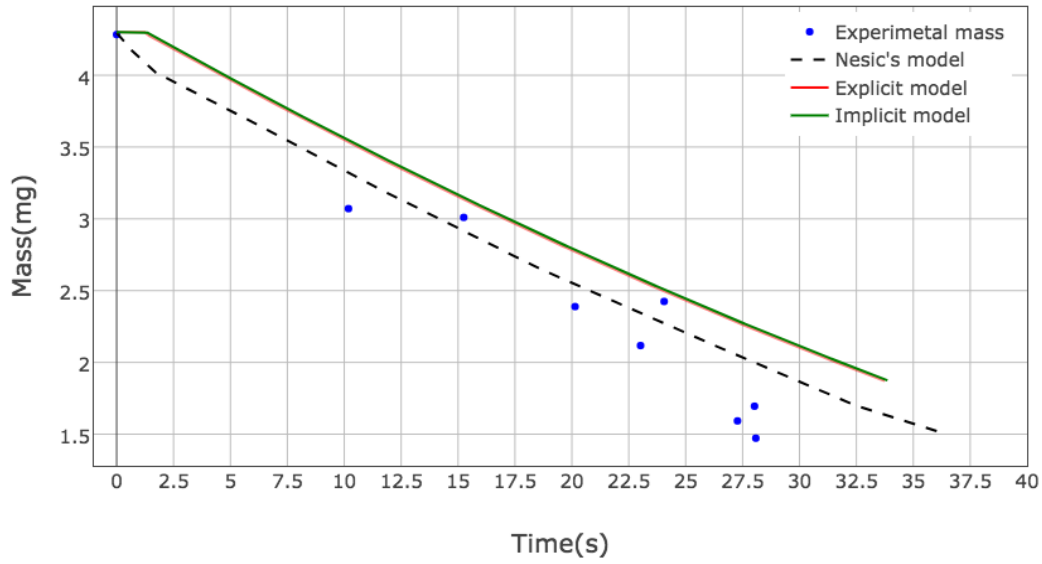


Figure 8.9: *Evolution in time of droplet mass for the initial heating and uniform evaporation temperature model.*

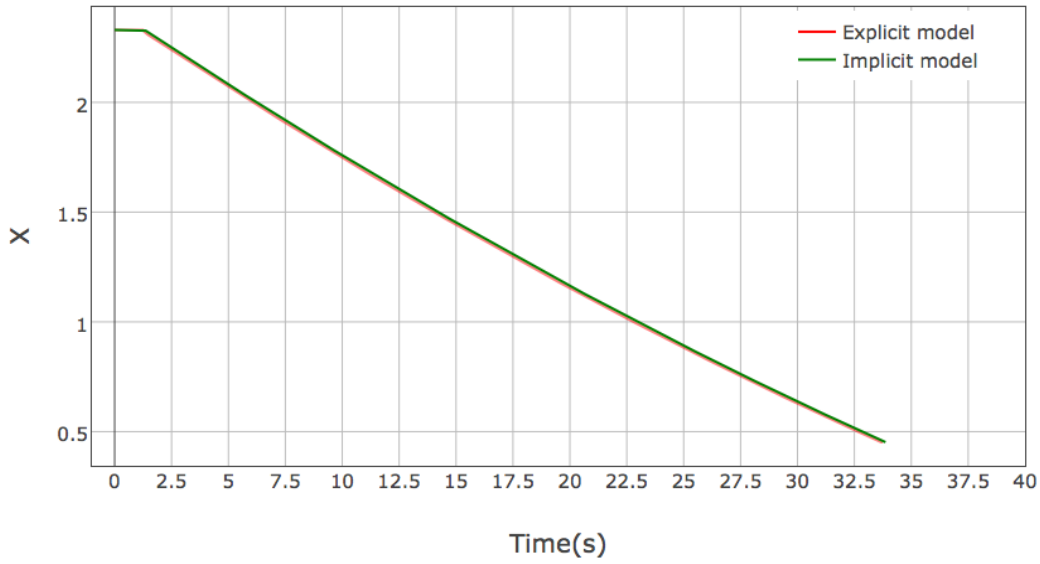


Figure 8.10: *Evolution in time of droplet moisture contentfor the initial heating and uniform evaporation temperature model.*

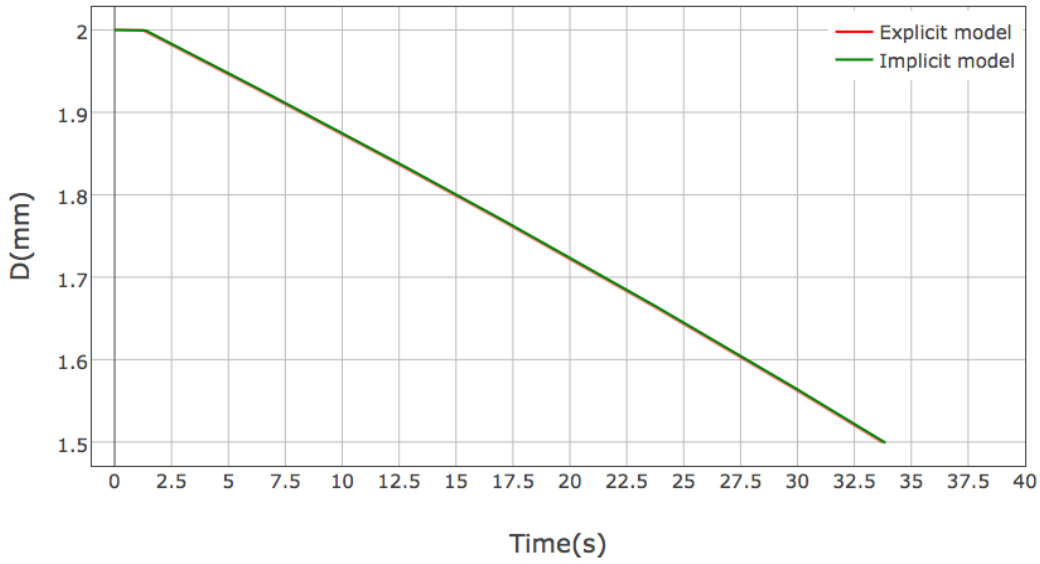


Figure 8.11: *Evolution in time of droplet diameter for the initial heating and uniform evaporation temperature model.*

8.1.3 Complete

This model is able to compute, at each time step, the temperature profiles within the droplet without any simplification (see Fig.8.12). The greatest complexity of this approach is to deal with a moving boundary condition on the droplet surface. Comparing Fig.8.15 with Fig.8.8 it is possible to note that there are no considerable differences regarding the evaluation of the droplet surface temperature. However, Fig.8.15 shows also the evolution in time of centre temperature that is different from the external one during the transient period. This can be justified analysing the trend of the Fourier number and Biot number in Fig.8.13 and in Fig.8.14 respectively. The Fourier number is inversely proportional to the square of droplet radius while the Biot number has simple linear relation. Therefore, during the initial heating period when evaporation is very low and the radius is constant (Fig.8.18), their values do not satisfy the lumped approximation and a difference of about $5 - 6°C$ exists between the centre and the surface but, if compared with Fig.8.5 and Fig.8.6 the boundary temperature values are $10°C$ smaller. However, at the end of this transient period, as the droplet radius start to shrink, the Fourier number increases and also the Biot number starts to decreases. The temperature profiles become flat and from now on the lumped approximation would be correct. For this complex problem only the implicit method for moving boundary problem has been implemented and the numerical algorithm can be found in section 5.1.3 and follows the one described in [10]. As before also results in terms of reduction of droplet mass, diameter and moisture content (Fig.8.16, Fig.8.17, Fig.8.18), are presented but no great differences from the other models can be noted.

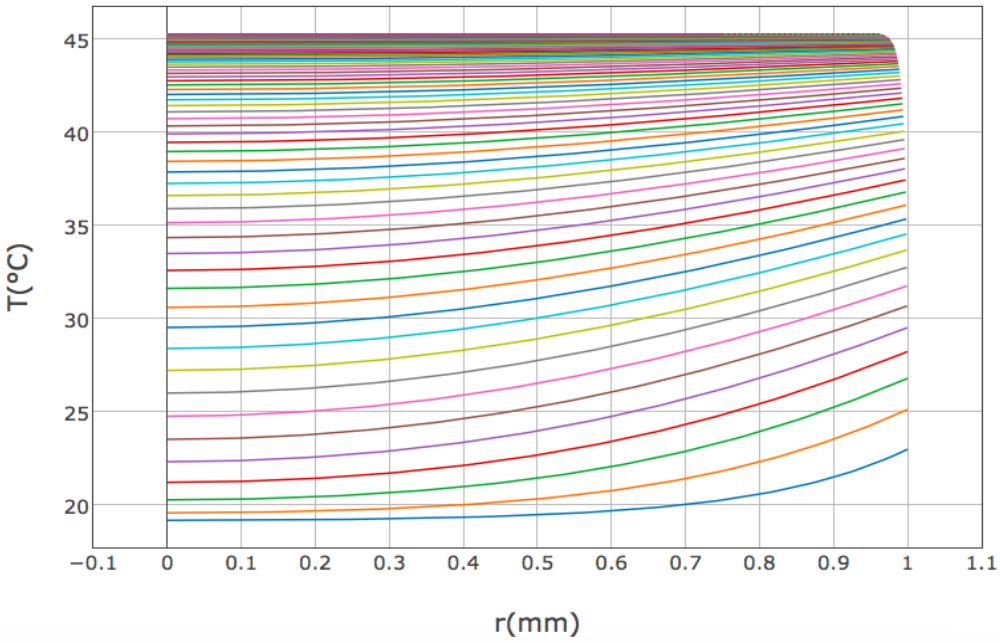


Figure 8.12: *Temperature profiles using the complete model.*

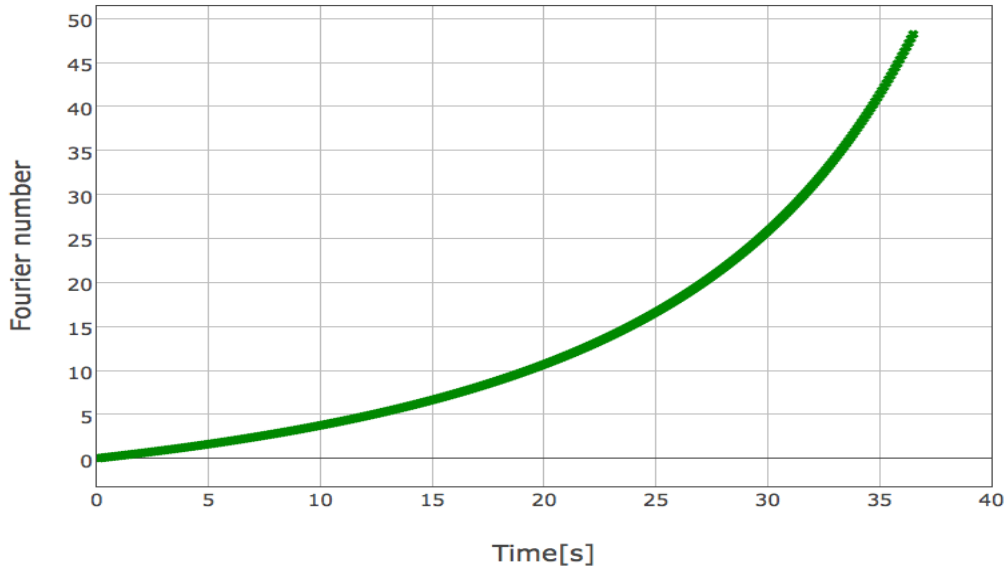


Figure 8.13: *Evolution in time of the Fourier number using the complete model.*

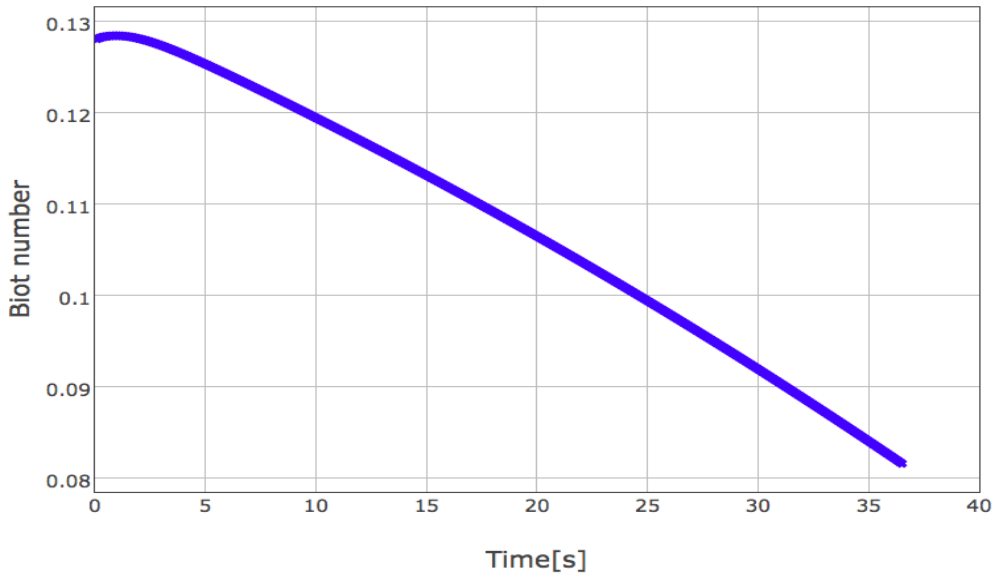


Figure 8.14: *Evolution in time of the Biot number using the complete model.*

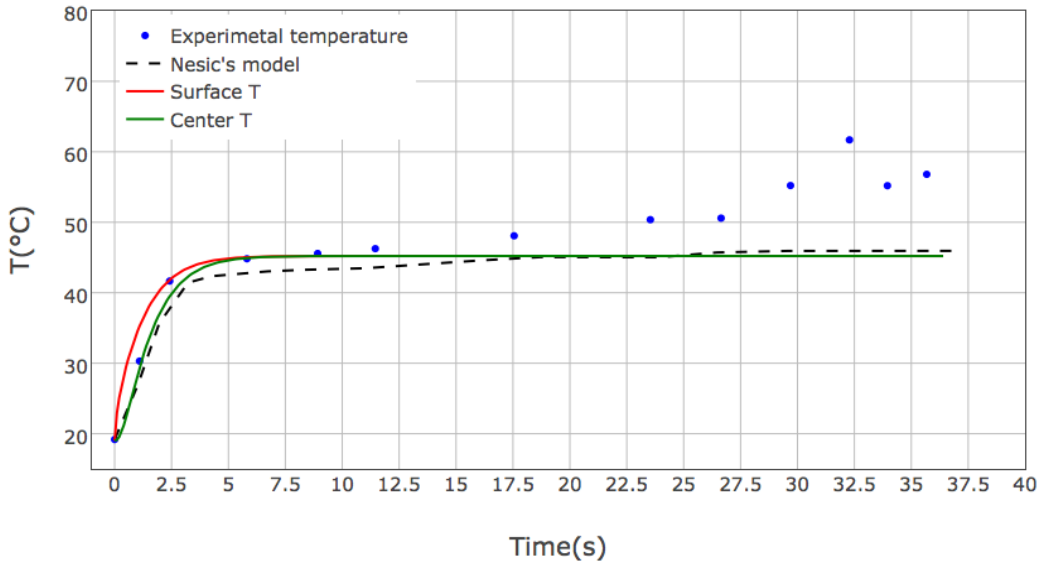


Figure 8.15: *Evolution in time of droplet surface and centre temperature using the complete model.*

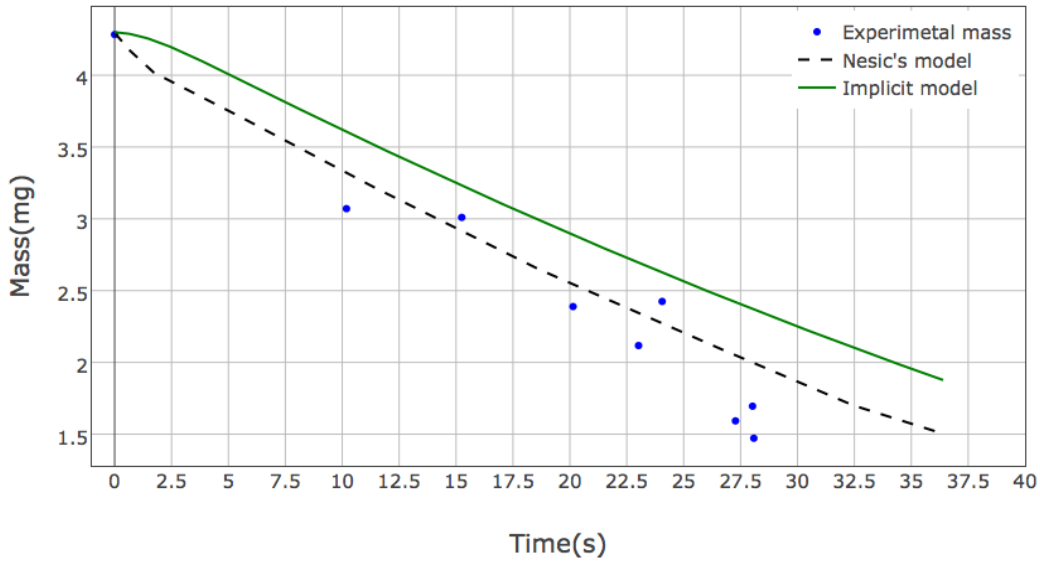


Figure 8.16: *Evolution in time of droplet mass using the complete model.*

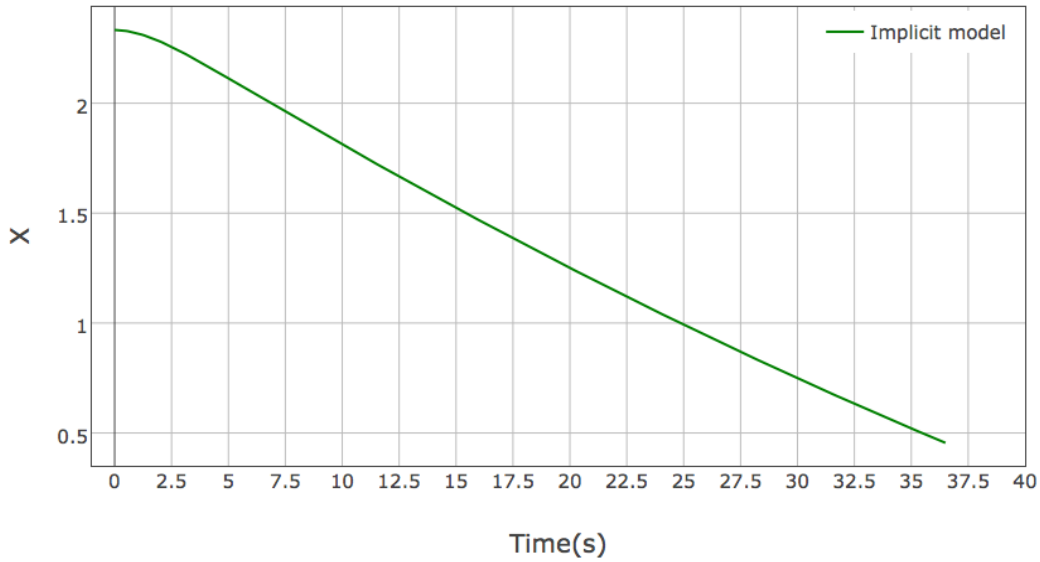


Figure 8.17: *Evolution in time of droplet moisture content using the complete model.*

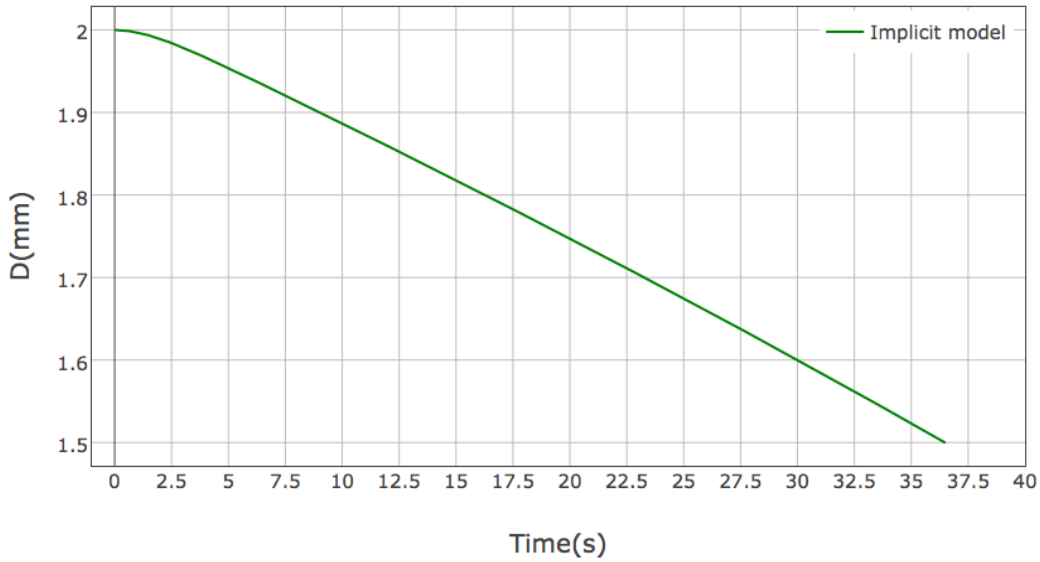


Figure 8.18: *Evolution in time of droplet diameter using the complete model.*

8.2 Second stage

8.2.1 Lumped

Also for the second stage, a lumped approximation has been implemented. When the critical moisture content X_{cr} is reached, the droplet turns into a completely non evaporating particle with a constant mass and radius. This is obtained by disabling the evaporation inside the previous first stage lumped model and the result is a steep temperature increase that is visible in Fig.8.22. As before, results are identical for the implicit and explicit model. Despite its simplicity this model shows a good agreement with experimental data and Nesic's model, whose approach was a little different from the one adopted in this section. Even if Nesic used a lumped approximation and therefore did not take into account the temperature profile inside the droplet, he does not set to zero the vapour mass flow rate but he uses an additional 'crust' diffusion coefficient that is considerably smaller than the convection diffusion coefficient [26]. This new diffusion coefficient models the resistance enhancement to vapour diffusion through the solid crust and in the original paper it is set to 10^{-6}; it should be underline that our approach is a great simplification since the wet core is completely neglected.

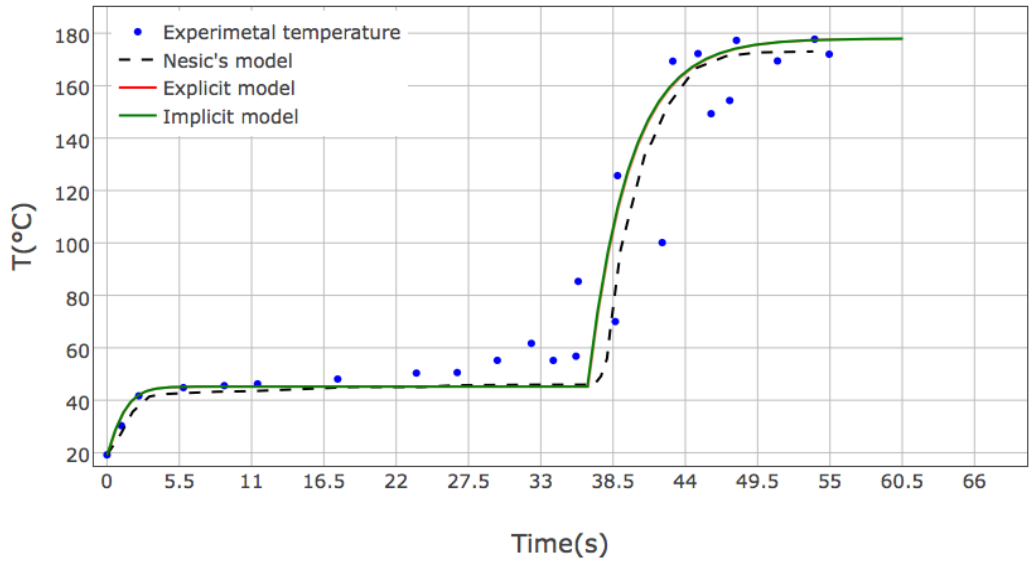


Figure 8.19: *Evolution in time of droplet temperature with second stage included using a lumped model.*

8.2.2 Complete

This approach is accurately described in section 3.5.7, and through the application of a numerical scheme for boundary moving problem ([10]) similar to that used for first stage, it is possible to have informations about the temperature profile inside the droplet, and the rate at which the crust recedes towards the centre. These informations can be found in Fig.8.20 and Fig.8.21, where respectively the last temperature profile and the interface measure from the centre are presented. When a final moisture content is reached, that is in this case $X = 0.05$, the evaporation stops and the droplet behaves like a solid particle which increases its temperature because it is subject to convective thermal exchange with the surrounding air.

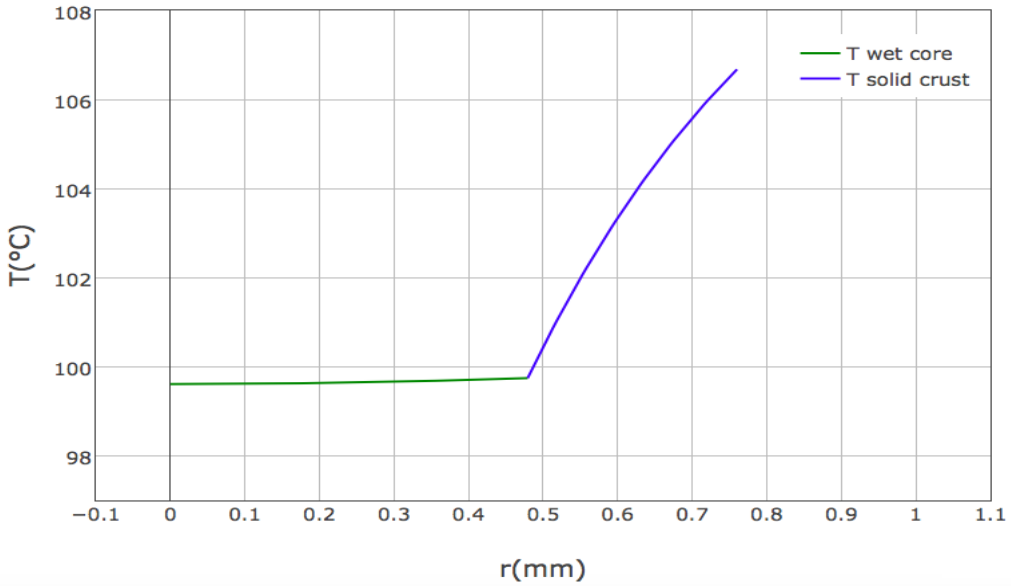


Figure 8.20: *Temperature profile inside the wet core and the crust using the complete second stage model.*

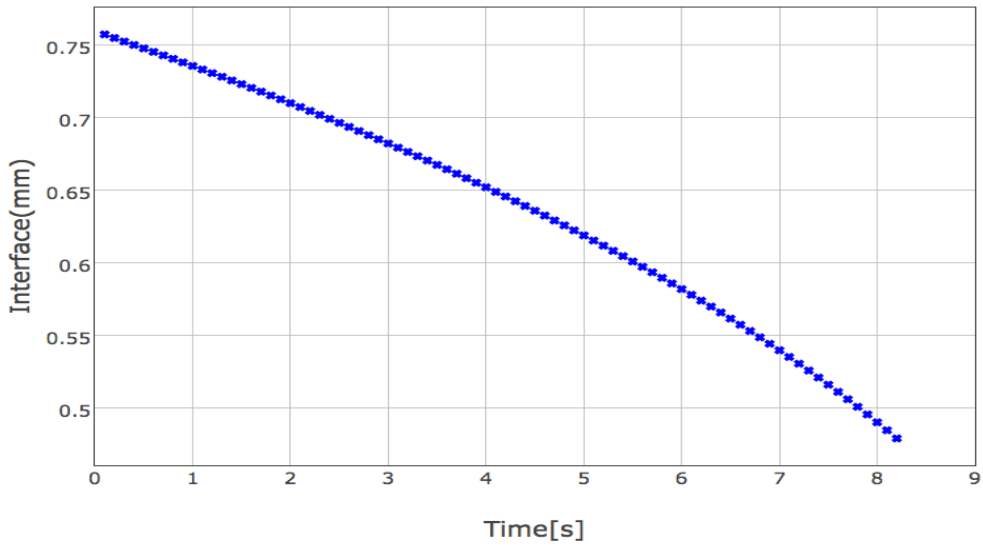


Figure 8.21: *Evolution in time of wet core crust interface using the complete second stage model.*

Fig.8.22 shows the evolution in time of the droplet temperature only during the period in which evaporation is enabled and it is possible to note that there exists a temperature difference between the centre and the surface but the rate at which temperature increases is underestimated if compared with experimental data. This could be due to difficulties to model the vapour diffusion through the crust porosity

especially because there is no information about the diffusion coefficient that governs the vapour Stefan flow used in this case. Fig. 8.23 shows the evolution in time of the droplet surface temperature, and it can be seen a steep increase of temperature when the evaporation stops that is similar to the one described for the previous lumped model.

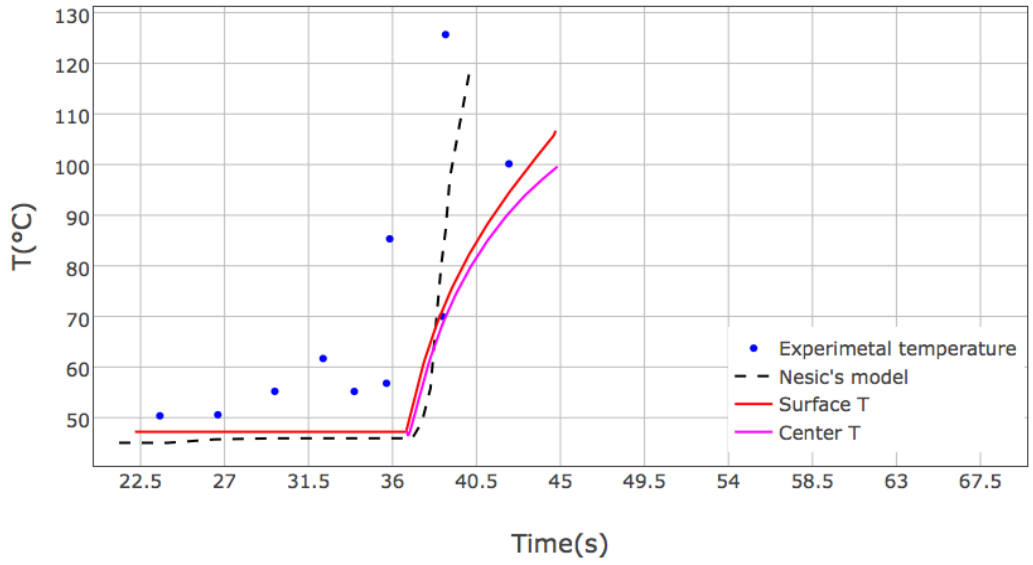


Figure 8.22: *Evolution in time of the droplet temperature during the solid crust formation period. The results are obtained using the complete second-stage model.*

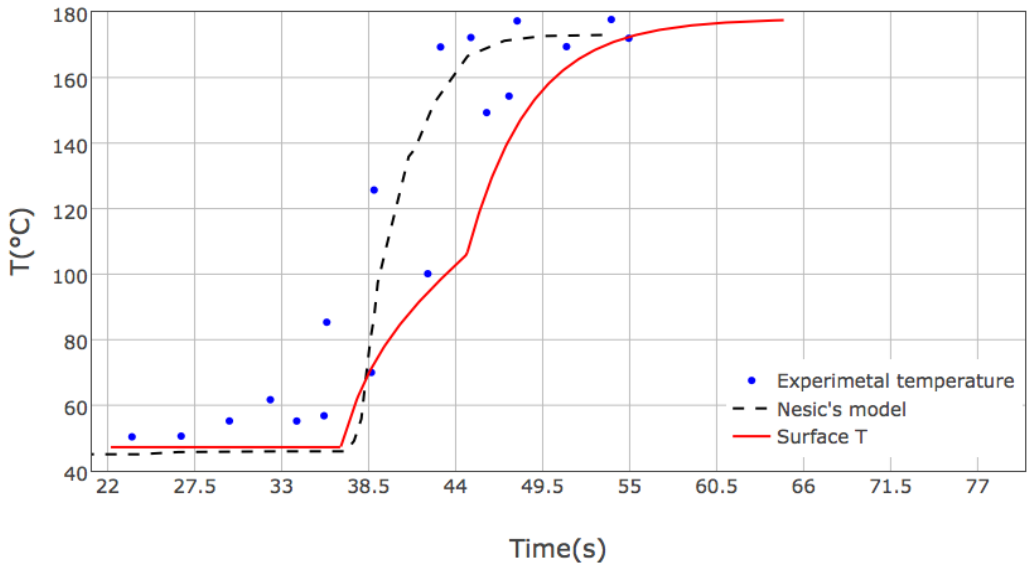


Figure 8.23: *Evolution in time of the droplet surface temperature. Results are obtained using the complete second-stage model.*

8.3 Grid dependency of the background fluid flow solution

To ensure that the solution is independent from the grid resolution, three different meshes have been tested. A convergence study is always needed for CFD simulations but it can involve high computational costs, especially for cases similar to the one studied in this work where the physical simulated time is long. Indeed, if the mesh is refined also the time step must be reduced because of the CFL condition and this leads to very long simulations. To have an order of magnitude of the computational time, the case with the coarser mesh needed two days to reach 170 seconds, but with the finest mesh it took almost 10 days. To test the effect of grid refinement, we present temperature and velocity profiles at the same height of the points in Fig.7.3. What concerns the temperature profiles shown in Figs.8.24-8.25, it is clear that for all meshes at the end of simulation there are no variations along the radius, confirming that all the domain is at 470 $[K]$. Analysing the velocity profiles in Figs.8.26-8.27, a rigorous study on grid refinement dependency would suggest a test with a finer mesh, since, especially for the velocity peak, there are still large differences. However the background fluid flow is not the main objective of this work and, also because

of the high computational costs, we decided not to further refine the mesh and to take the last solution with the finer one as the starting point for the simulation with particles. When compared with the others, the velocity profiles for the finest mesh are also more symmetrical and this is expected for this geometry.

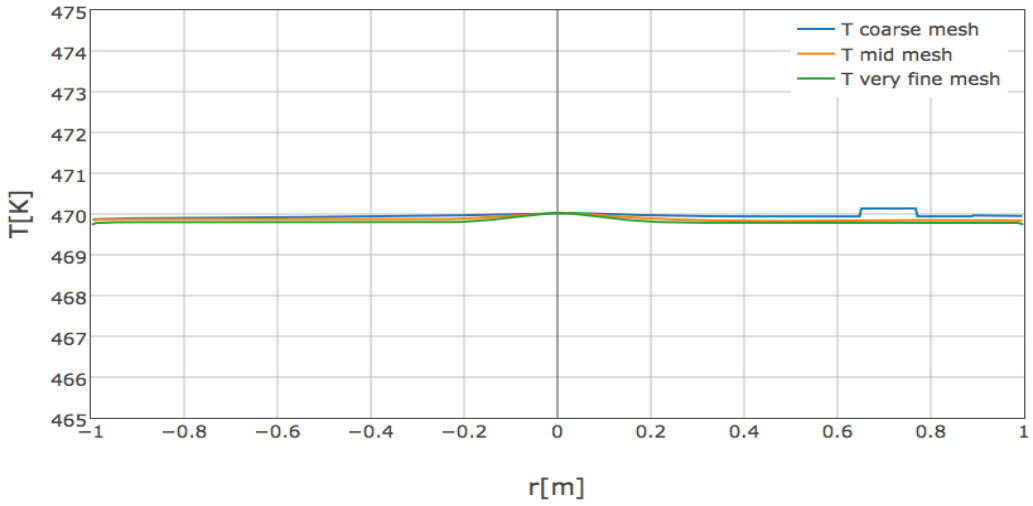


Figure 8.24: *Temperature profiles near the inlet for three different meshes.*

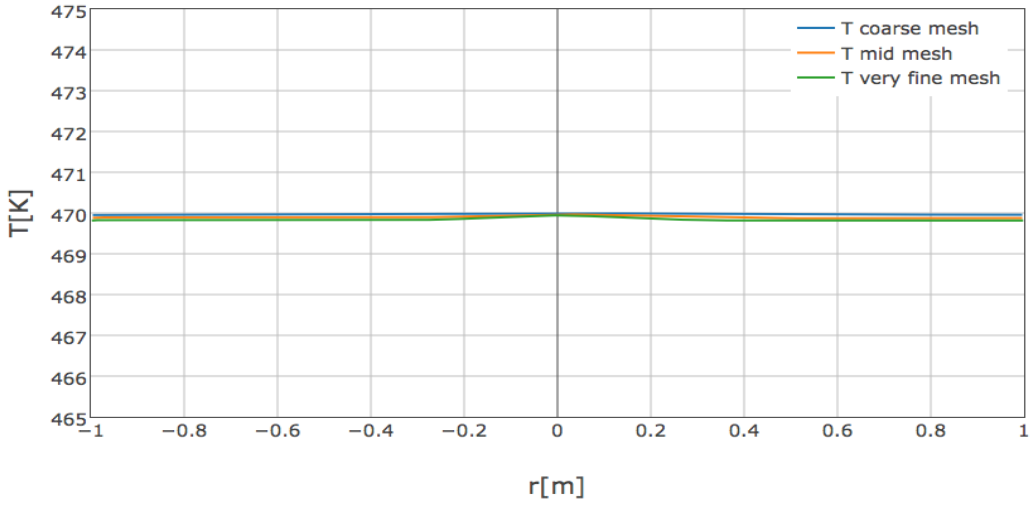


Figure 8.25: *Temperature profiles near the outlet for three different meshes.*

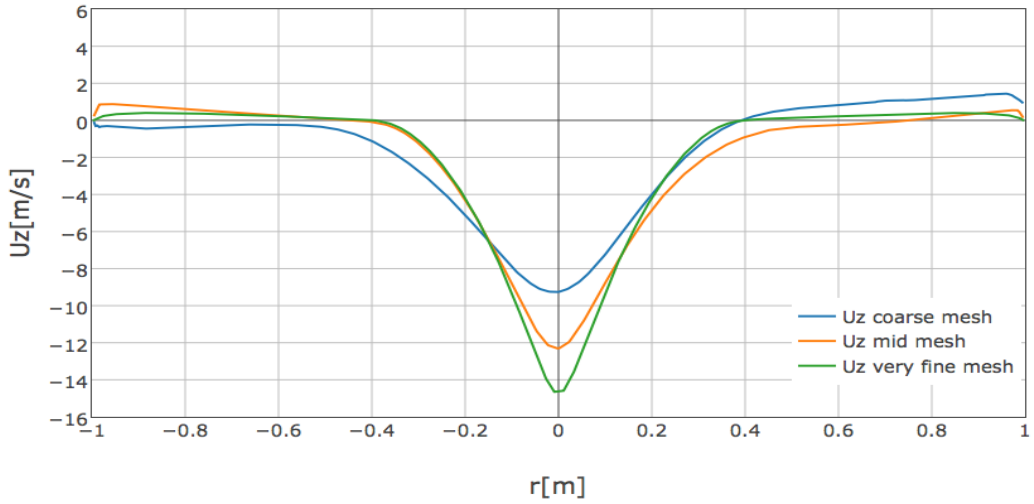


Figure 8.26: *Vertical component of velocity profiles (U_z) near the inlet for three different meshes.*

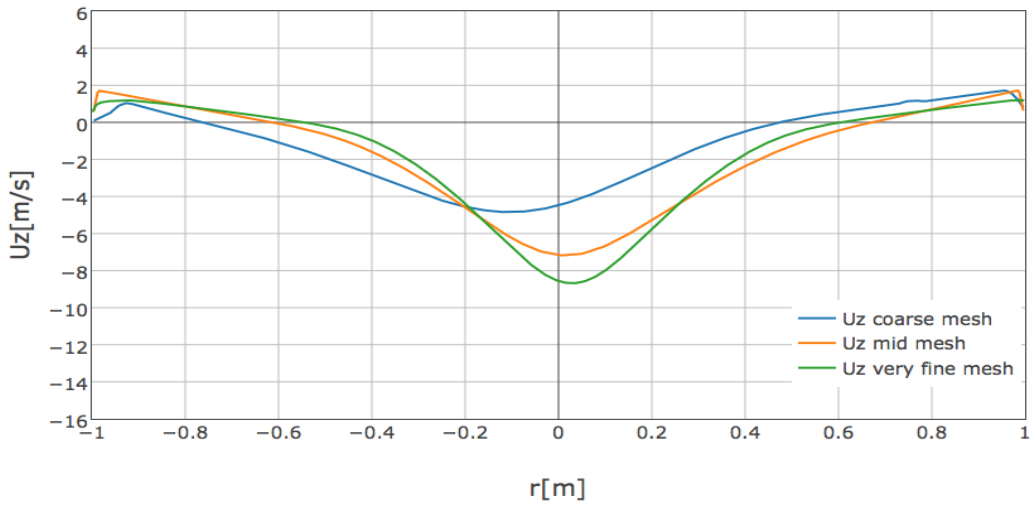


Figure 8.27: *Vertical component of velocity profiles (U_z) near the inlet for three different meshes.*

8.4 Implementation of the particles in OpenFOAM

The main objectives of this section are:

- a comparison between two different Lagrangian solvers on the first drying

stage:

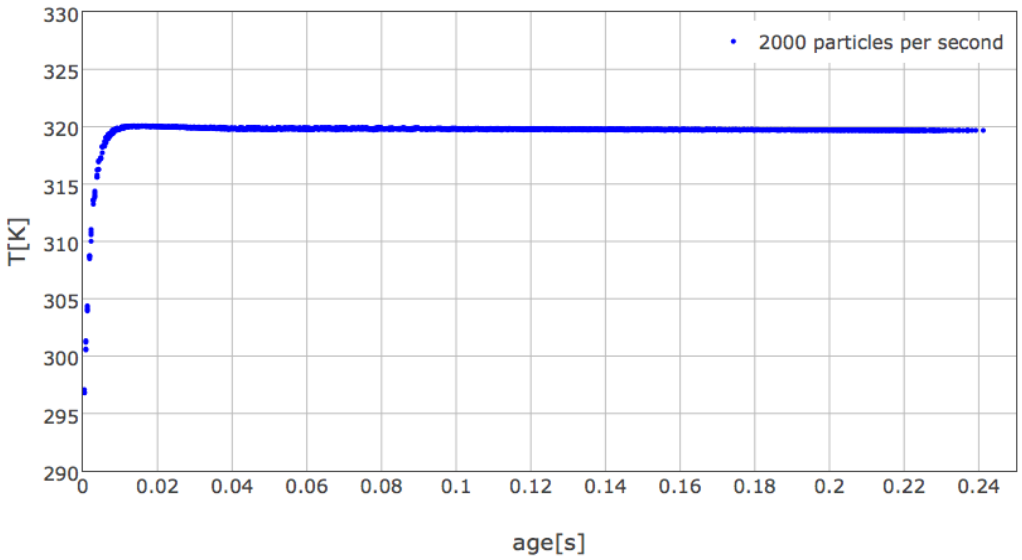
  - *ReactingParcelFoam*: the one already implemented in OpenFOAM®;
  - *MA_BuoyantKinematicParcelFoam*: the one implemented for this thesis that is able to model the two evaporation stages;

- the presentation of some results about the coupling of the first and simplified second drying stage;

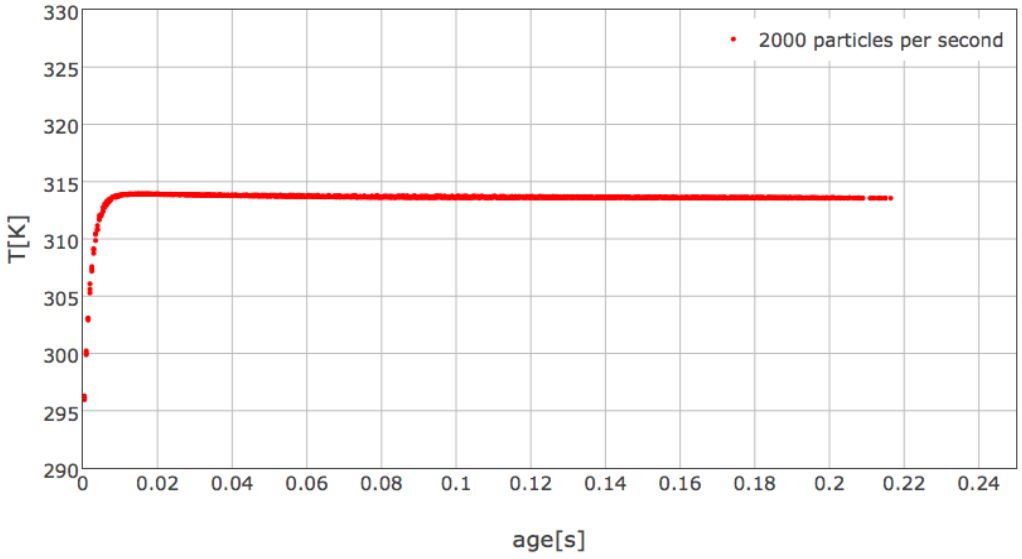
8.4.1 First drying stage

In order to carry out a validation of our Lagrangian solver *MA_BuoyantKinematicParcelFoam*, we decided to make a comparison with the solution that the solver *ReactingParcelFoam* provides for pure liquid droplets. To make this comparison, within our solver, where the specific two stage evaporation model is implemented, the concentration of solids inside the liquid drop has been set to a very low value (0.003). This results in an initial moisture content of 332.3 that is high but far for the ideal infinite value that is characteristic of pure liquid droplets. As described in chapter 2 and in chapter 4, when the critical moisture content is reached, the first drying stage ends and the second one is applied to the dried particles. All the tests have been conducted with 2000 particles per second, injected for 2 seconds inside the computational domain. In order to get into a limit case, we decided to make simulations with our solver with perfectly dry air. This is obtained by setting to 0 the partial pressure of vapour in the drying air but we left the standard settings for *ReactingParcelFoam* (initial mass fraction of $H20 = 0.01$). The solver *MA_BuoyantKinematicParcelFoam* has two variants based on the two algorithms described in section 5.1. When the explicit time integration scheme is applied, it has been modified to first ored explicit Euler method instead of the second order Adam-Bashforth method. The implicit algorithm is quiet complex and not strictly necessary in this case, but tests have been conducted also with this method because it is really similar to the one described in section 5.1.3 that is the most complete approach that can provide information about the temperature inside the droplet. Therefore a simulation carried out with this method is easily extended to the most complete case only with some modifications inside the code. However, with this method we encountered some convergence problems inside an intermediate while loop when the droplet radius was too small, and this resulted in a final moisture content of 20, quite far from critical one that was 0.45. This problem could be solved

by decreasing the time step but clearly this leads to very high computational costs especially for longer simulations with much more particles injected. The explicit method has been more stable, and allowed us to reach the critical moisture content also with smaller computational costs. Since results for the explicit and implicit model are identical at least as long as there were no convergence problems, only results for the explicit method are presented. Fig.8.28 shows the evolution of droplet temperature as a function of droplet age. Age is a parameter that has straightforward meaning since it is the time from which the droplet has been injected in the domain. Hence it is possible to note that the trend of the two curves is very similar but there is a difference regarding the evaporation temperature and the maximum droplet age. These differences may come from a different initial value of drying air humidity that has a strong influence on the temperature. Smaller humidity values enhance evaporation and this leads to smaller equilibrium temperature. It should be underlined that our solver enables evaporation only for droplets with moisture content values greater than the critical one. This corresponds to a droplet diameter of about 20 μm, hence a selection using this cut off value has been made on the *ReactingParcelFoam* cloud. Since the droplets with greater ages have the smallest diameters as shown in Fig.8.29, the difference regarding droplets age is related to the rate at which diameter shrinks, and that is probably related to the evaporation temperature and to the evaporation rate mentioned above.

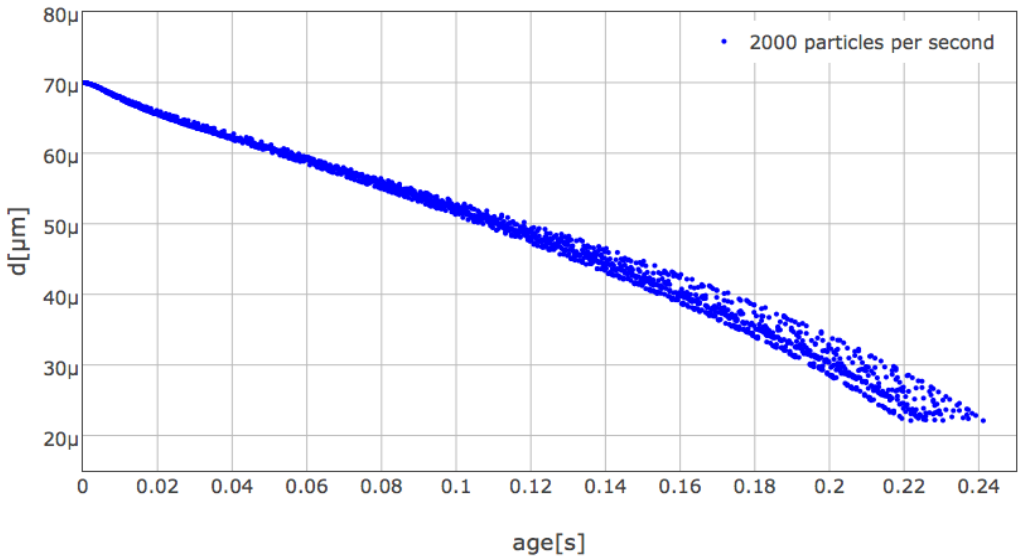


(a) *ReactingParcelFoam*

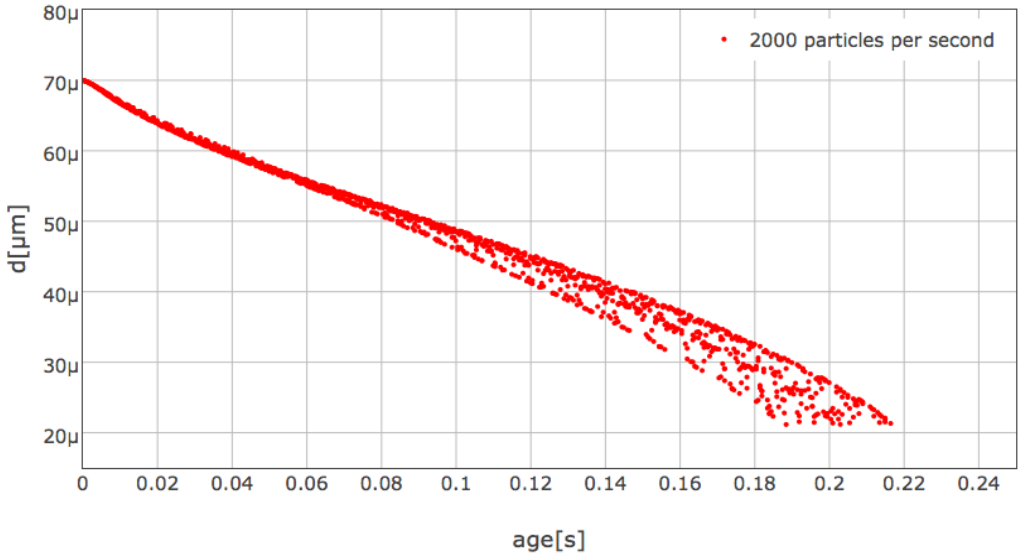


(b) *MA_BuoyantKinematicParcelFoam*

Figure 8.28: *Droplets temperature history.*



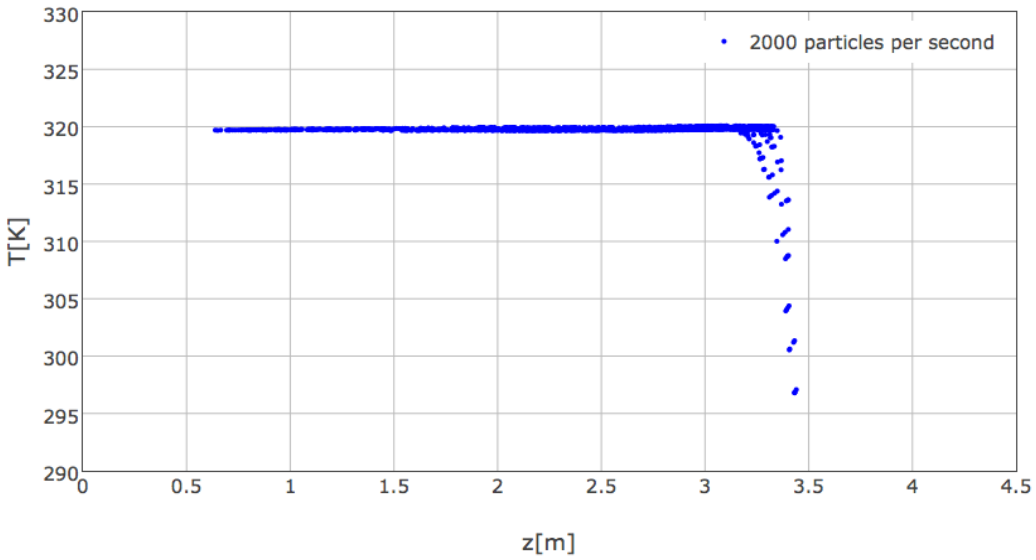
(a) *ReactingParcelFoam*



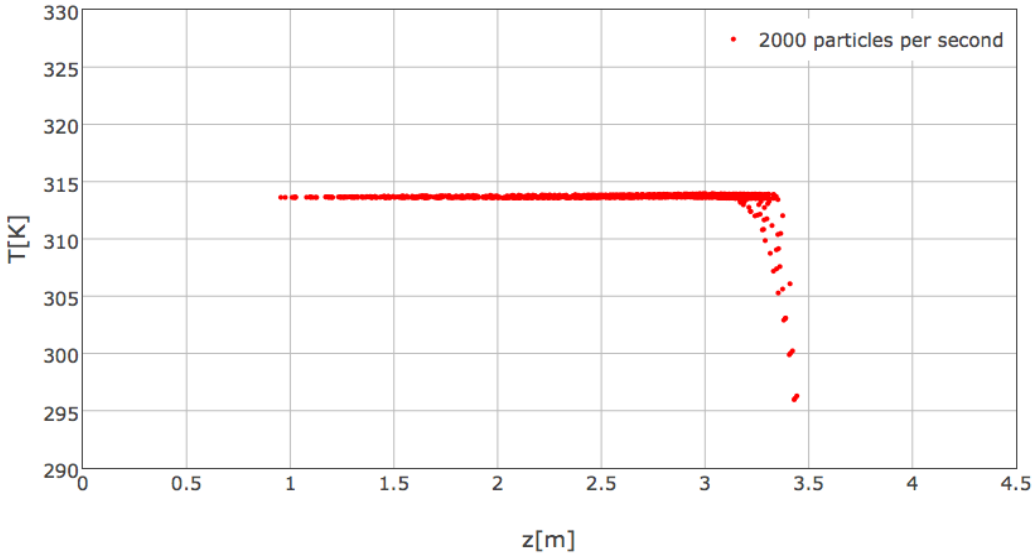
(b) *MA_BuoyantKinematicParcelFoam*

Figure 8.29: *Droplets diameter shrinkage history.*

Fig.8.30 shows that for the two solvers the equilibrium temperature is reached near the injection zone that is located at an height of 3.45 $[m]$.



(a) *ReactingParcelFoam*



(b) *MA_BuoyantKinematicParcelFoam*

Figure 8.30: *Droplets temperature as a function of their vertical coordinate.*

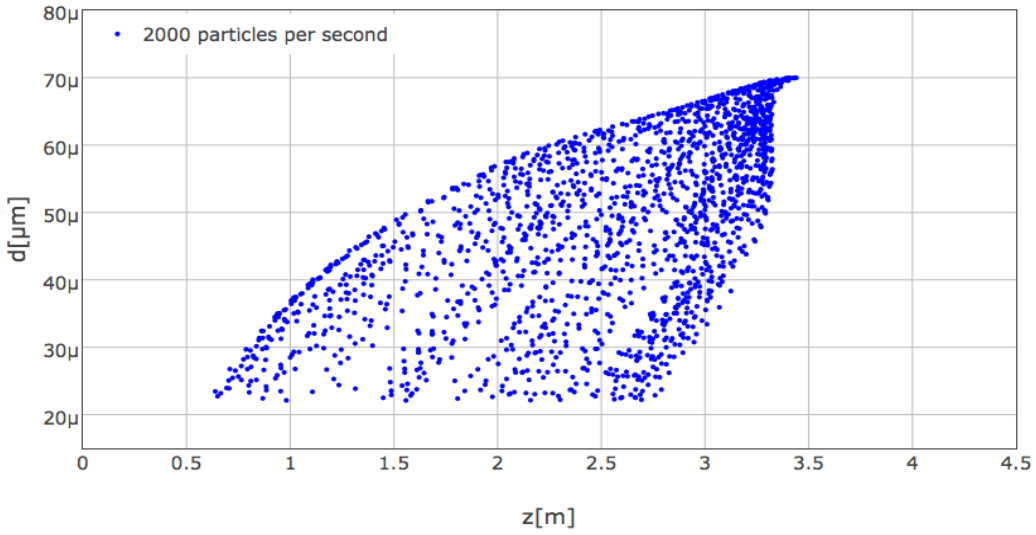
Figures 8.31 and 8.32 suggest some considerations. Two interpretation keys are possible:

- if we consider droplets with the same diameters, Fig.8.31 confirms that the

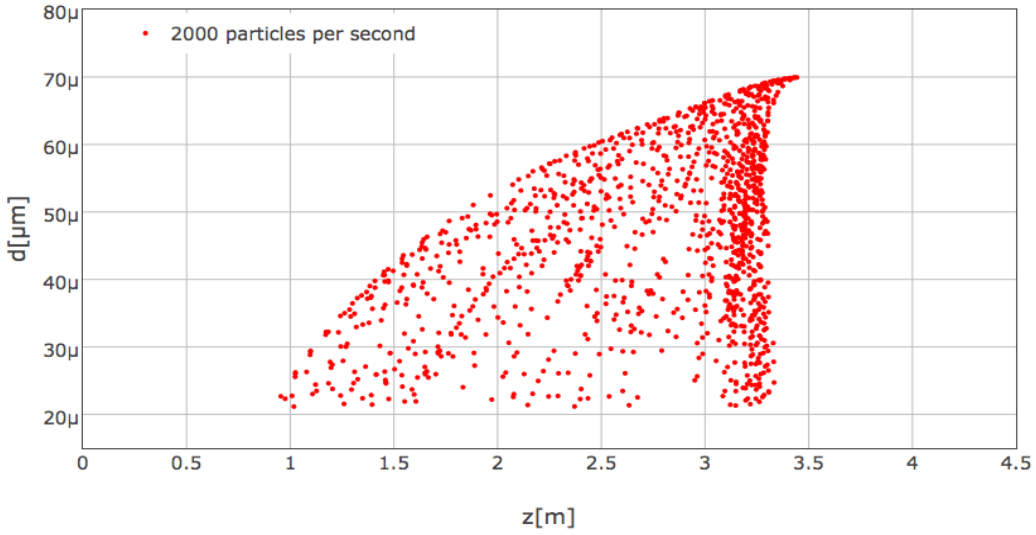
one related to *ReactingParcelFoam* are located in a range of vertical coordinate that is lower than the new solver; this can be explained by keeping in mind that droplets with the same diameter, but computed with the two different solvers, have different ages, and the the biggest values are those for *ReactingParcelFoam* solver;

- if we consider droplets with the same ages, Fig.8.32 confirms that they assumes the same vertical coordinate range in the domain.

From these considerations it is possible to conclude that the small mass differences (due to diameter differences) between the particles with the same age but, computed with the two different solvers, do not affect the dynamic of the droplets.

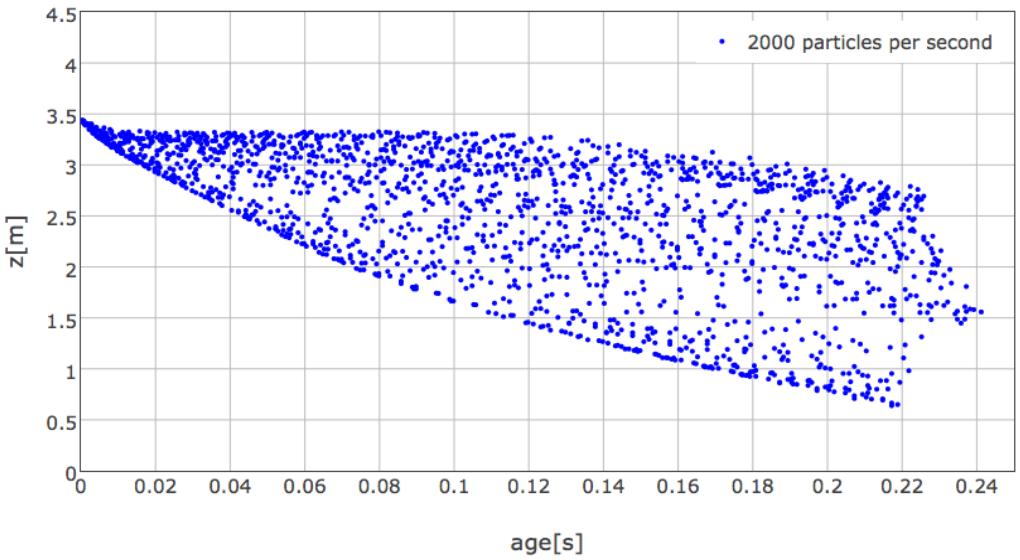


(a) *ReactingParcelFoam*

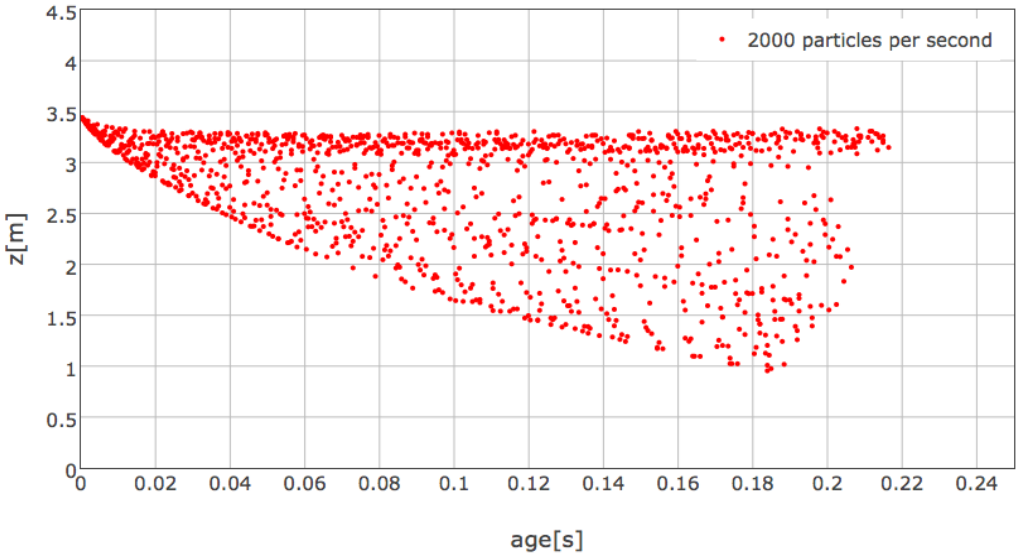


(b) *MA_BuoyantKinematicParcelFoam*

Figure 8.31: *Droplets diameter as a function of their vertical coordinate.*



(a) *ReactingParcelFoam*



(b) *MA_BuoyantKinematicParcelFoam*

Figure 8.32: *Droplets vertical coordinate as function of age.*

Moreover, the new solver implemented provides also information about the moisture content and the rate at which it is reduced from the high initial value of 300 until the critical one of 0.45. This is shown in Fig.8.33.

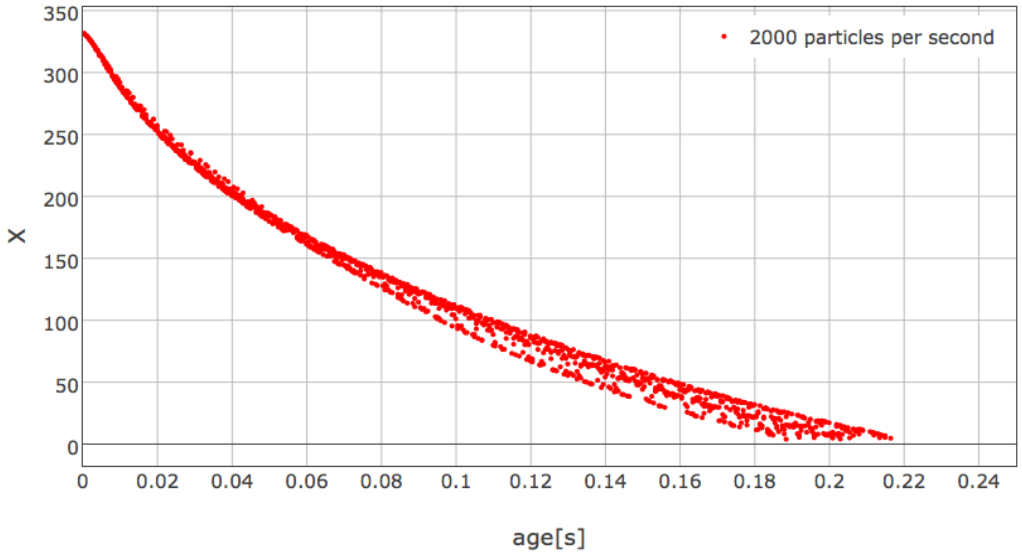


Figure 8.33: *Droplets moisture content history.*

Figures 8.34 and 8.35 show the shape of the droplets cloud. The main difference is in the region near the injection zone, where the droplets cloud for *ReactingParcelFoam* is more concentrated. Indeed there are some differences in the way that the solvers treat the background Eulerian fields, and this may result in different velocity fields that governs the droplet dynamics. The droplets that reached the critical moisture content are still kept within the domain and treated as non evaporating solid particles (Fig.8.36). Because of the bottom recirculation zone, also shown in Fig.7.4, the particles are dragged up to half of the domain.

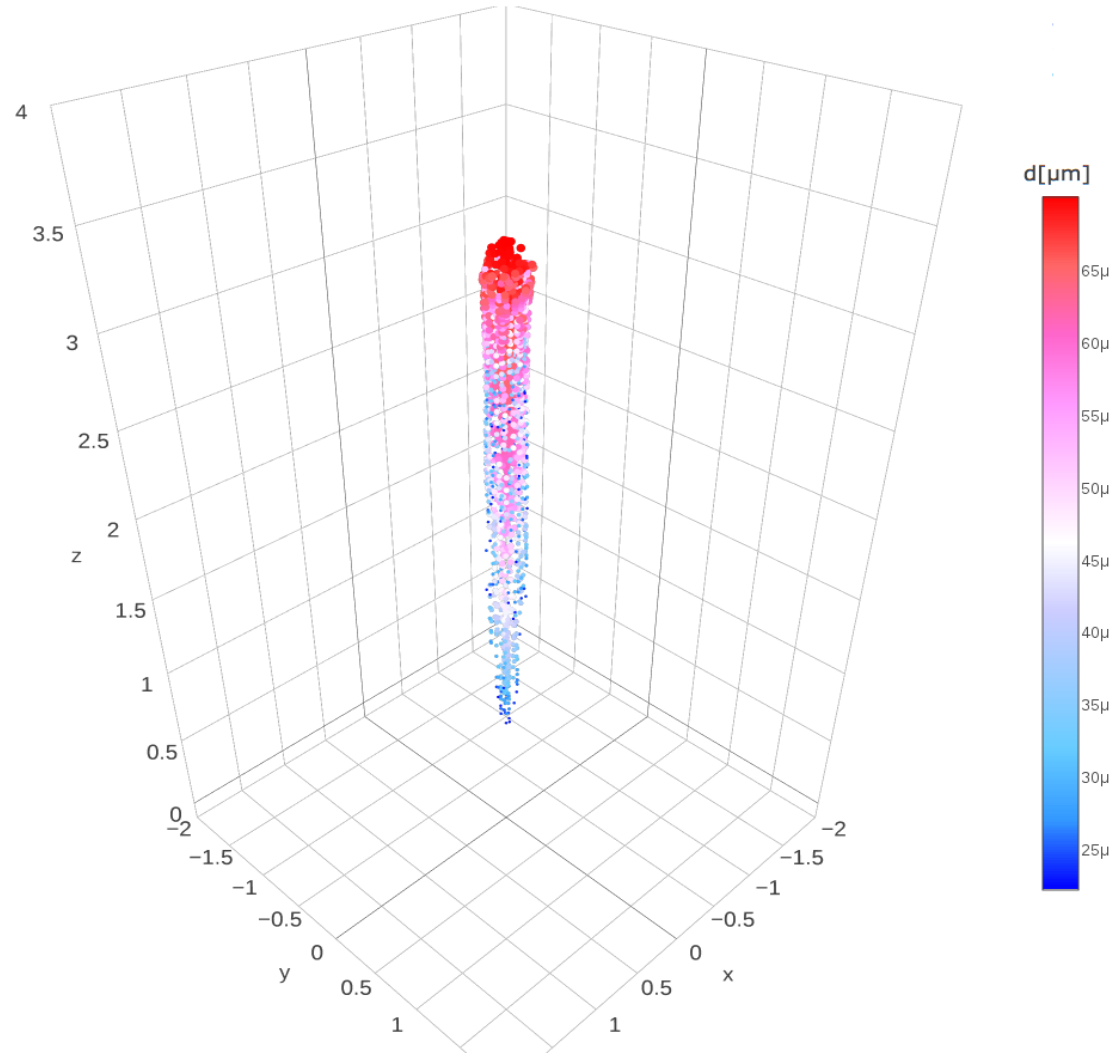


Figure 8.34: *Droplets cloud for ReactingParcelFoam.*

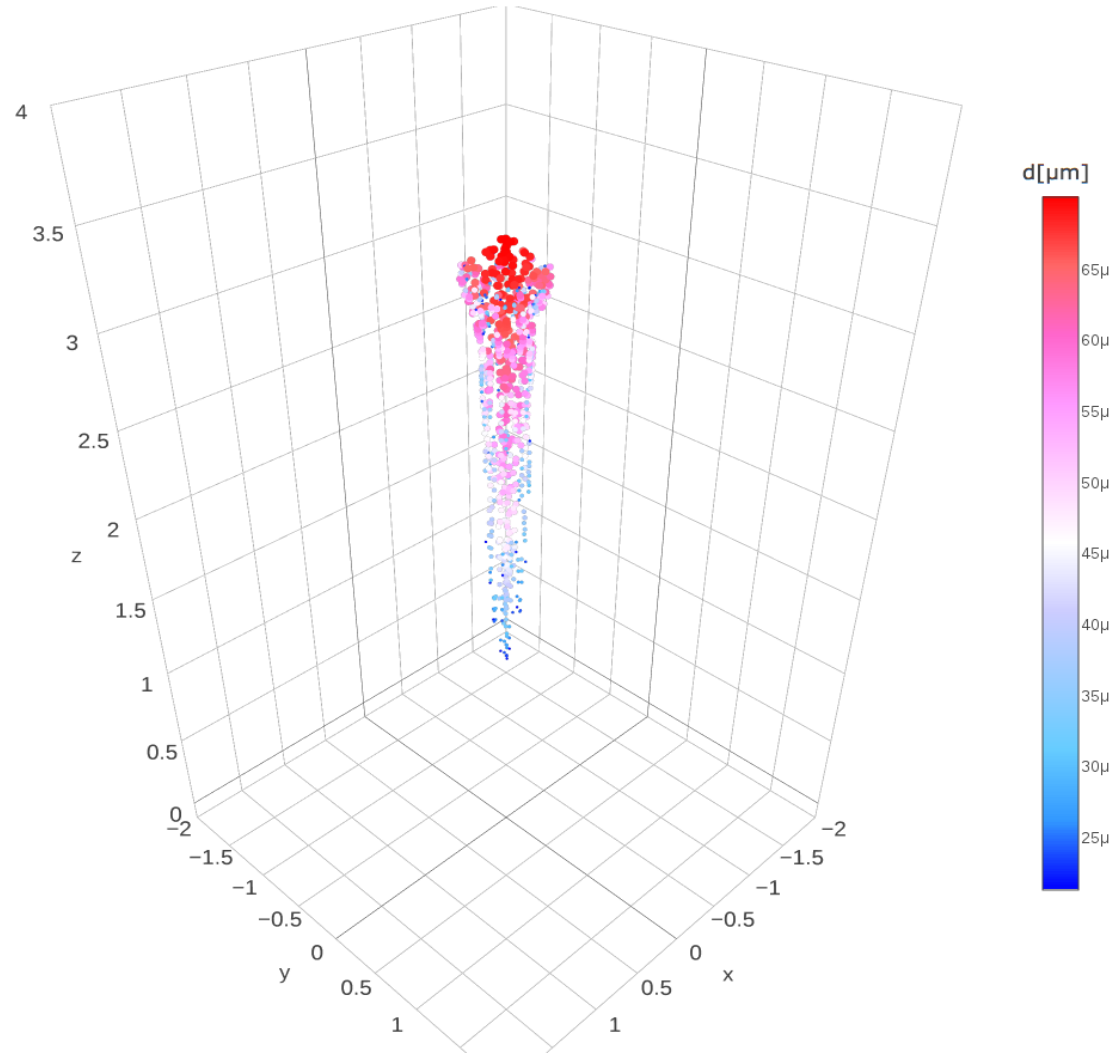


Figure 8.35: *Droplets cloud for MA_BuoyantKinematicParcelFoam.*

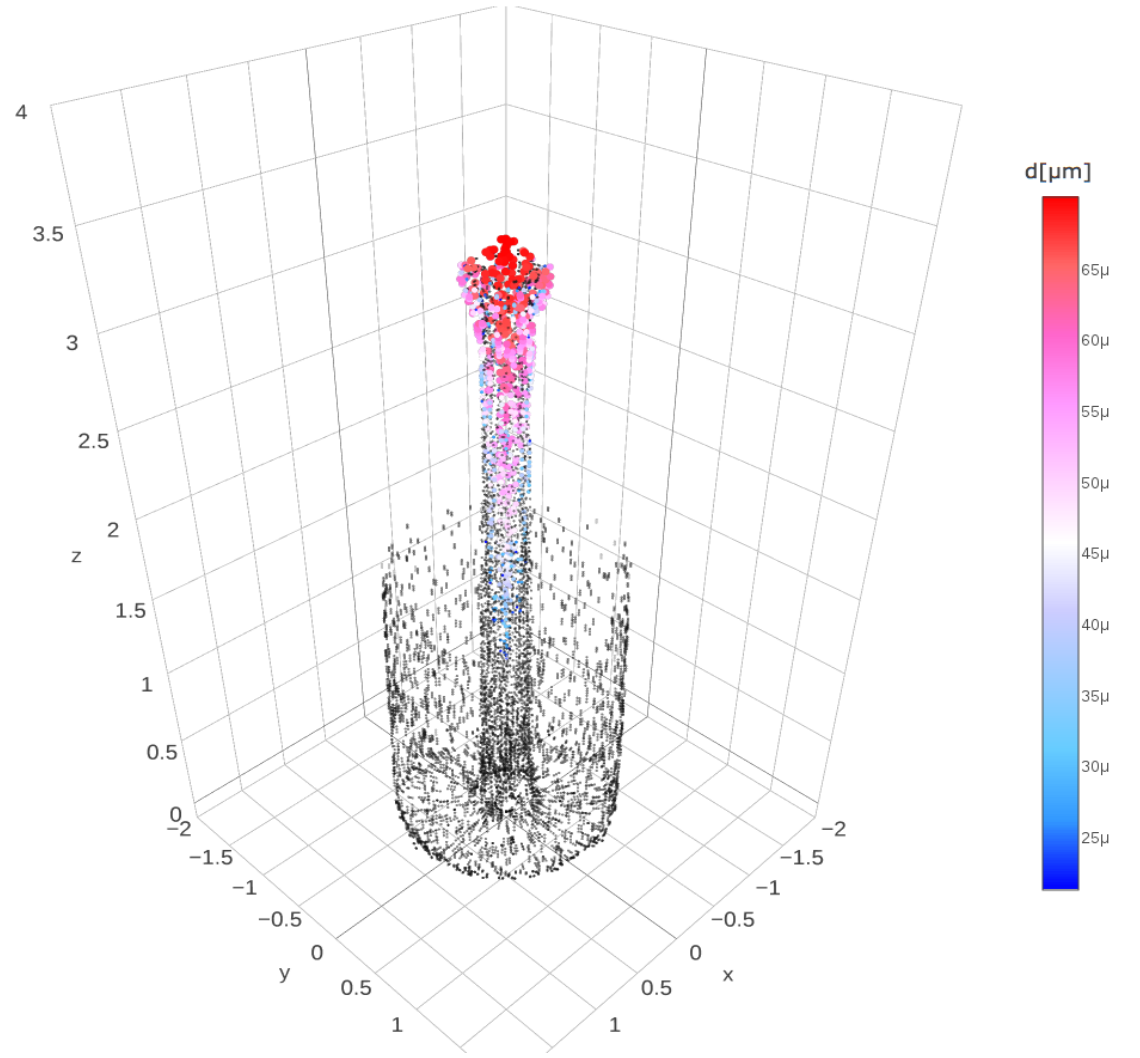


Figure 8.36: *Droplets cloud for MA_BuoyantKinematicParcelFoam. The black dots represent the droplets that reached the critical moisture content and could therefore be regarded as solid particles.*

In order to show the influence on the evaporation temperature and on the droplets diameter history, we also made some tests by setting the value of air relative humidity to 0.2%. Fig.8.37 shows that the equilibrium temperature reaches 319 $[K]$ which is similar to the results from *ReactingParcelFoam*. Also the diameter shrinkage rate (Fig.8.38) reduces and assumes values comparable to those shown in Fig.8.29(a). In coclusion this shows that the relative humidity of the air is an important parameter.

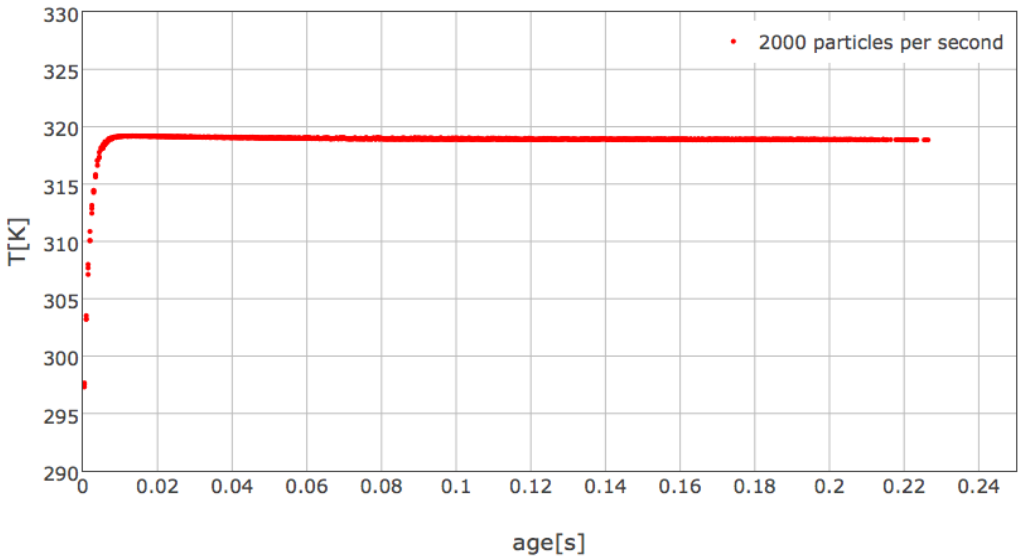


Figure 8.37: *Droplets temperature history with modified drying air relative humidity.*

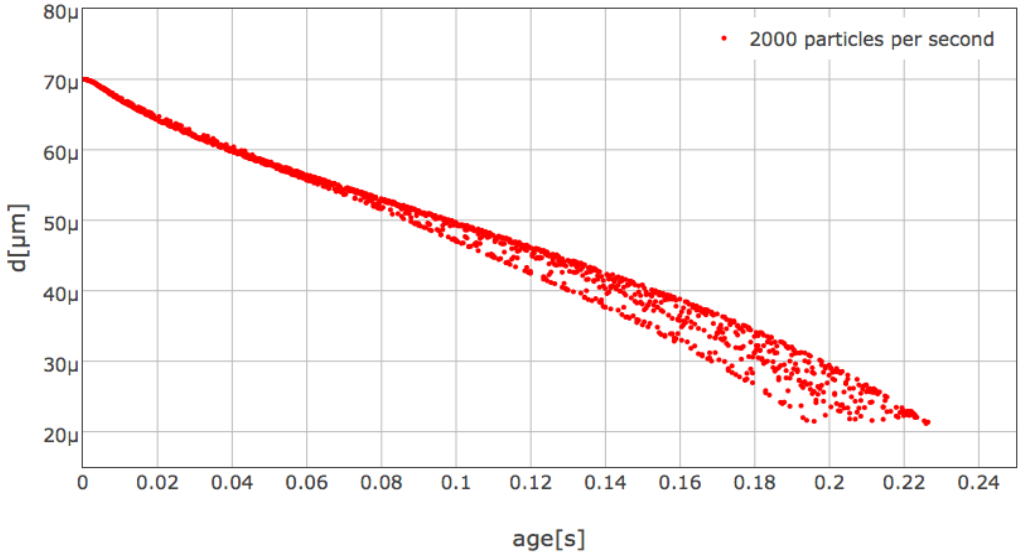


Figure 8.38: *Droplets diameter shrinkage history with modified drying air relative humidity.*

8.4.2 Second drying stage

The results for the second drying stage are shown after making some changes to the set up of the previous case in order to present a more realistic condition. Indeed

the initial solid content within the droplets is set to 0.3, which is the same value used for the all the tests in chapter 4 and it is 100 times smaller than before. As shown in Fig.8.40 and Fig.8.41 the main feature, about the first drying stage, that is different from the previous case is that the droplets reach the critical moisture content very near the injection zone and with a much greater diameter (60% higher). However, when the critical value is reached, the droplets turn into non evaporating particles and their temperature starts to increase until the drying air temperature is reached (Fig.8.39). It should mentioned that this is a very simplified approach and only a first attempt to model in a CFD code a very complex process. Indeed the real physical process involves a solid crust formation and a wet core still exists; the main difficulty is to track the wet core shrinkage and the vapour diffusion through this receding interface. Thus this approach can only provide information about the droplets temperature (whose prediction could be also quite accurate) but usually, in spray drying, the moisture content of drying powder is an important parameter; therefore knowing where the particles with a certain amount of moisture content are inside the domain could be really important to construct an optimized geometry that provides powder with the required moisture content. However temperature information could be important if we are treating heat sensitive products, and in this case the goal is also to avoid increasing the powder temperature too much. However since there is no more evaporation the diameter is kept constant by the solver and this results in the horizontal line in Fig.8.40 and Fig.8.41. Figure 8.43 shows the particles with a moisture content greater than the critical one and finally the complete cloud is presented in Fig.8.44 where the particles treated as solid remains confined in the bottom region. This is a difference from the previous case, see Fig.8.36, but it is consistent since now particles are more massive and the recirculation can not carry them up.

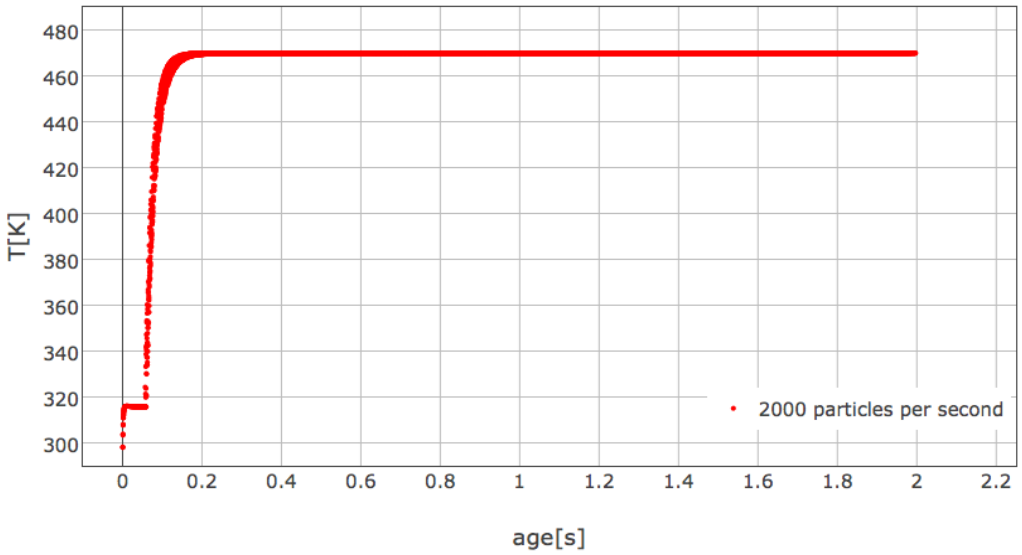


Figure 8.39: *Droplets temperature history.*

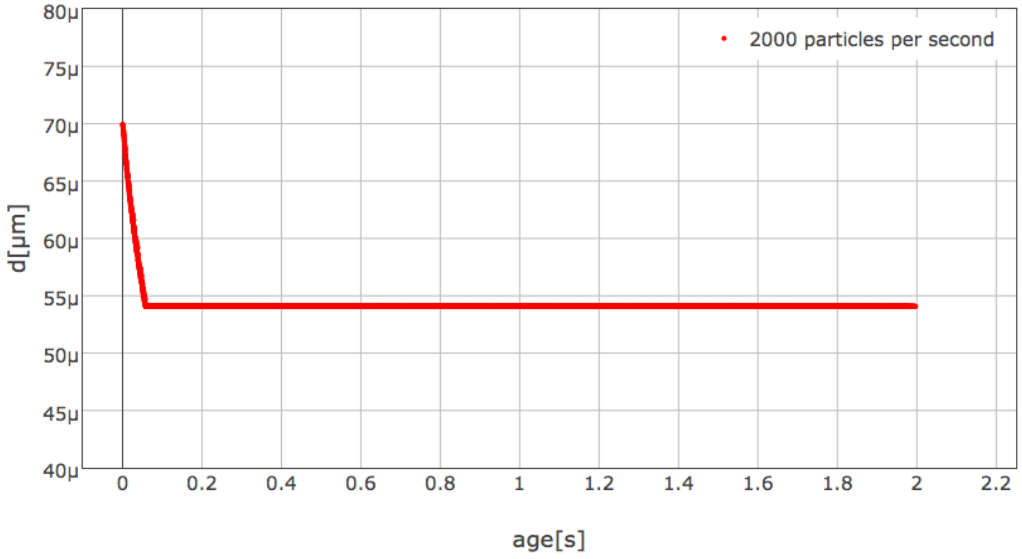


Figure 8.40: *Droplets diameter history.*

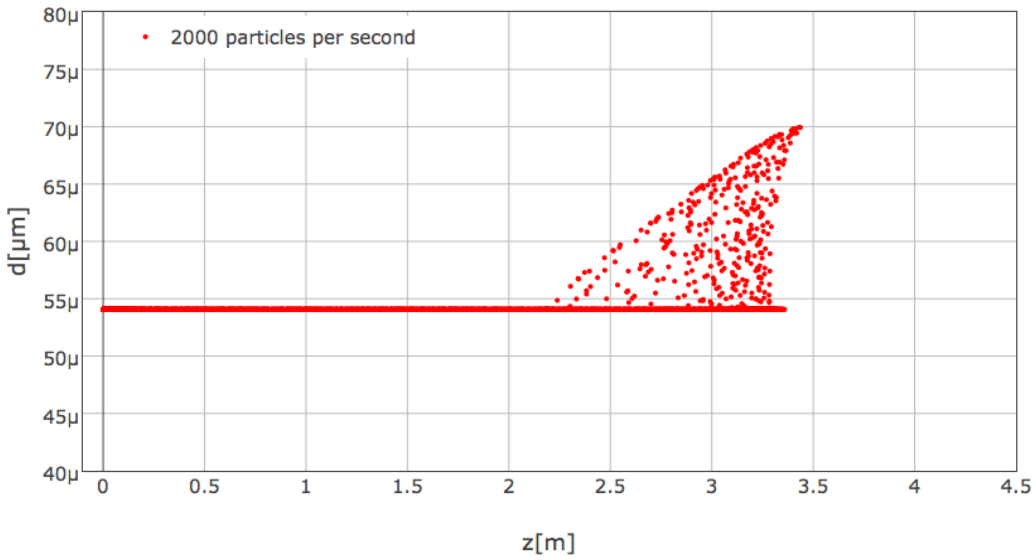


Figure 8.41: *Droplets diameter as a function of their vertical coordinate.*

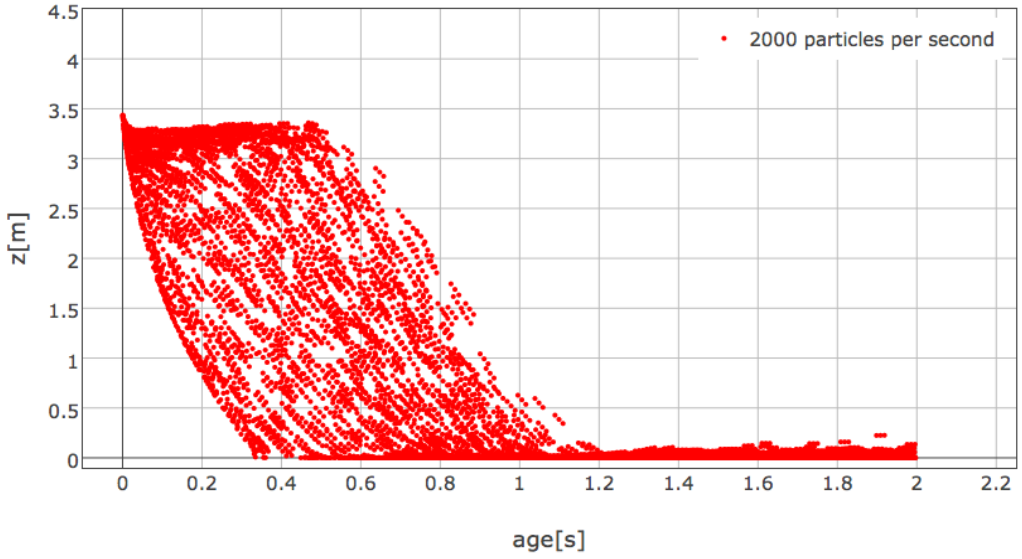


Figure 8.42: *Droplet vertical coordinate as a function of age.*

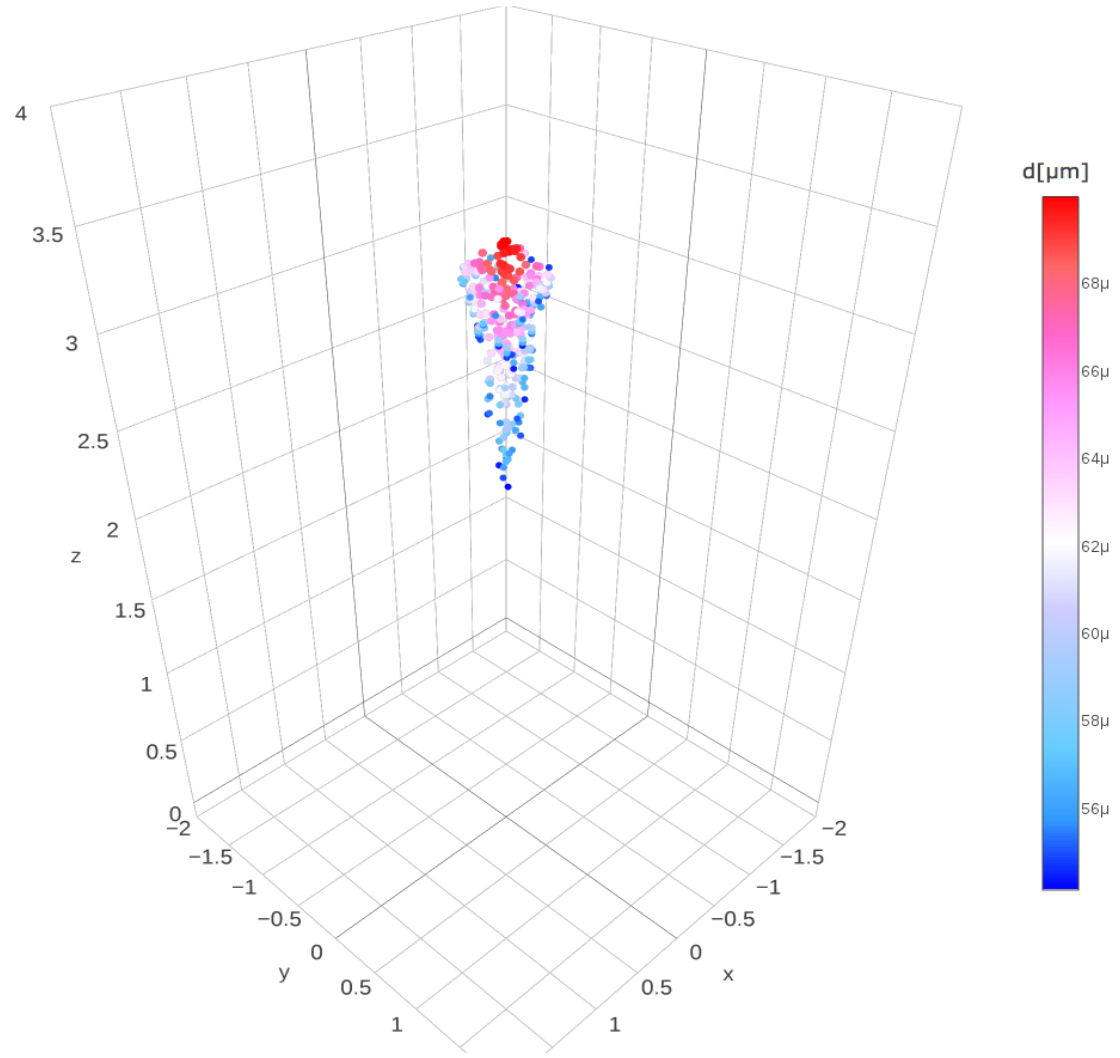


Figure 8.43: *Droplets cloud for MA_BuoyantKinematicParcelFoam. Only particles with* $X > X_{cr}$ *are shown.*

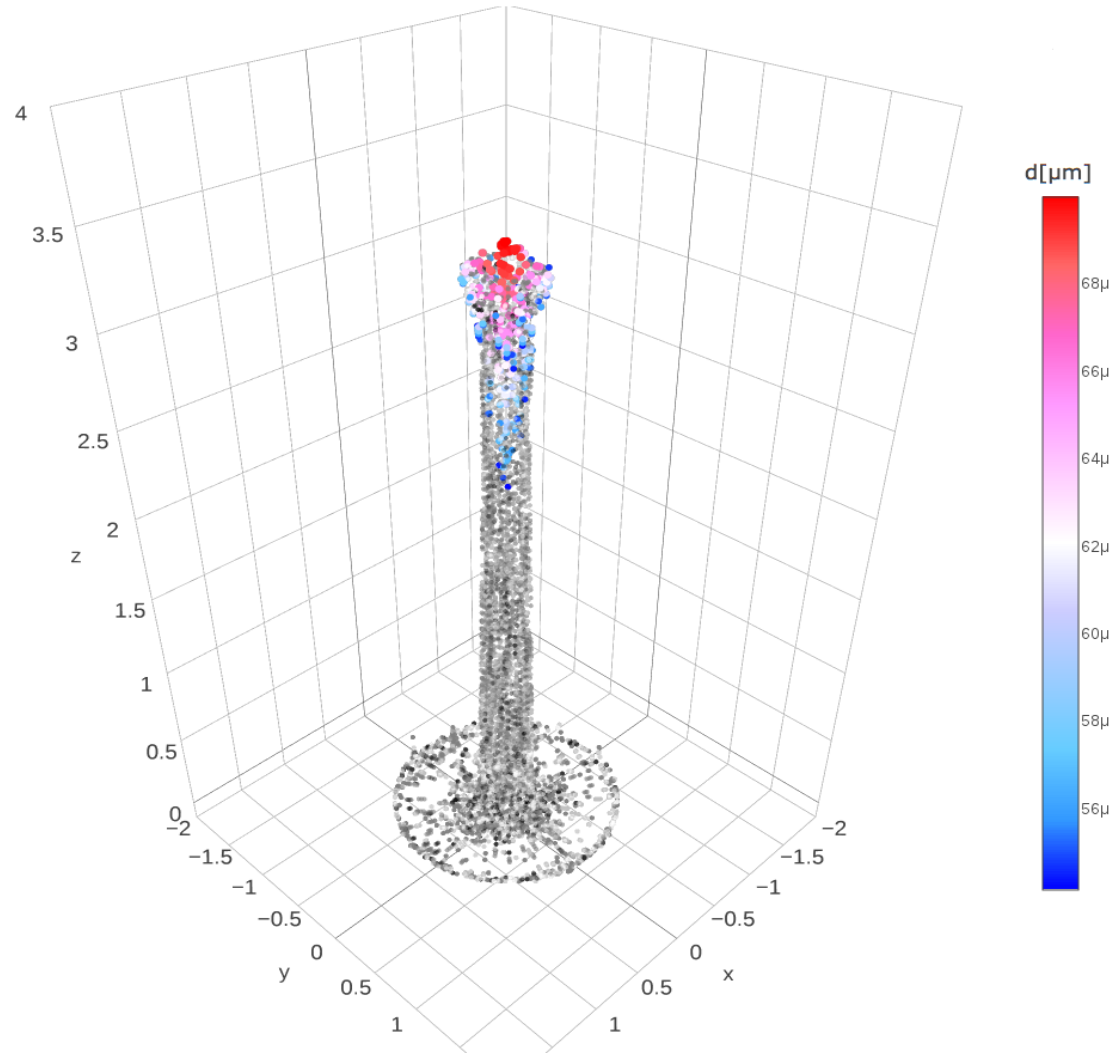


Figure 8.44: *Droplets cloud for MA_BuoyantKinematicParcelFoam. The black dots represent the droplets that reached the critical moisture content and can be considered solid particles.*

Conclusions and Future Developments

During this thesis a two stage drying model for droplets containing insoluble solids has been implemented. The specific case of silica slurry droplets with diameter of 2 $[mm]$ is treated throughout the first part of the work because of the possibility of an experimental data comparison.

The first part of the results concerns the validation of three different approaches for the first drying stage of a motionless drop with an increasing approximation level, showing a good agreement with temperature experimental data for all models.

The main assumptions for this stage is that the mixing inside the drop is considered to be ideal without recirculation, and the critical moisture content is evaluated through a simplified approach based on a critical averaged solid-liquid ratio.

In the second part the second drying stage has been implemented, and the solid crust formation with the receding wet core interface has been modelled. We assume a Stefan flow type for the evaluation of the vapour mass transfer rate through the solid crust porosity, and the wet core interface is still considered to be in saturation condition. According to this model it is possible to note a difference between the central wet core and the surface temperature, and this can lead to considerable thermal stresses inside the droplet. When the final moisture content is reached the droplet is considered to be completely dried.

Indeed more detailed models in literature are able to predict the particle breakage due to thermal stresses, to model the vapour diffusion through the crust capillaries and also to evaluate the possible pressure rising inside the wet core at elevated temperatures of the drying agent.

A simplified model of the second drying stage has been also implemented; when the critical moisture content is reached, the drop is considered to be completely solid and the wet core is neglected. The droplet temperature showed good agreement with experimental data, but the very important information about the droplet moisture

content and the rate at which the interface recedes are finally lost.

These preliminary studies have been the basis for the implementation of the two stage evaporation model in OpenFOAM®, resulting in the new application *MA_BuoyantKinematicParcelFoam*. The new implemented application uses the lumped approximation both for the first and second drying stage. The computational domain is a cylinder that respects the proportion of the tall-form spray drier described by Stefano Pastorino in his master thesis; 2000 droplets per second with an initial diameter and temperature of respectively 70 μm and 292 $[K]$ have been injected for 2 seconds when a steady state condition of the surrounding flow was reached; this has led to the need to evolve the flow up to 170 $[s]$ and then a one way coupling Lagrangian particle tracking is performed.

The comparison of the first drying stage, carried out with pure liquid droplets, between the new application and the OpenFOAM® built in solver *ReactingParcelFoam* showed a good agreement in terms of temperature droplets history and diameter shrinkage rate, and tests have been also made with different air relative humidity showing its importance on the equilibrium evaporation temperature.

After this comparison, the simpler second stage model has been added to the first one, resulting in the evaluation of droplets temperature raise until the one of the drying agent. Regardless the model applied for this stage, the innovation respect to the built-in OpenFOAM® solver is that particles are kept inside the computational domain and tracked until the end of the simulation, rather than being considered completely evaporated under a certain mass value.

The implementation of the first drying stage have to be improved by adding the diffusion process of the solid within the droplet, in order to predict with accuracy when the critical moisture content is reached on the droplet surface. Also the OpenFOAM® implementation needs further developments; simulations must be carried out with much more particles injected, and the new solver has to be tested in the geometry proposed by Stefano. The second stage model included in this application is simplified; we think that the way of lumped approximation could be right also for the second stage if no particle breakage model are used, but the diffusion of vapour through the crust must be included. In this way the particle moisture content is always evaluated, in order to know, when the final value is reached, and where the particles with that moisture content are in the domain.

At the present stage collisions between the particles are neglected, but for more dense fluid they can affect particles trajectories. Moreover since the particle Stokes number is less than 1 ($St \sim 0.12$ near the inlet), the effect of turbulent fluctuations

on particles trajectories is in general not negligible. The solution of RANS-URANS equations does not provide information about the instantaneous velocity field, but this approach has been adopted in the present study because of its simplicity and low computational cost. The most straightforward way to overcame this limitation is to describe the fluid using LES, where most of the large scales of the fluid are described and a model is employed for the smaller scales only: the resulting instantaneous velocity is much more similar to the actual one. Another possible way is to adopt a dispersion model, which takes into account the effects of the turbulent fluctuations. It can be done, for instance, by adding a correction to velocity field seen by the particles. Both these approaches can be easily adopted in OpenFOAM, once the evaporation has been implemented, but for time reason they have not been studied in the present thesis.

It would also be appropriate to modify the wall boundary conditions in order to include rebound coefficients depending on moisture content that can predict better the particle wall deposition.

Bibliography

[1] Web page. `https://cfd.direct/`.

[2] C. Anandharamakrishnan. *Experimental and Computational Fluid Dynamics Studies on spray-Freeze-Drying and Spray-Drying of Proteins.* PhD thesis, Loughborough University, 2008.

[3] Martina Naumann Andreas Buck, Mirko Peglow and Evangelos Tsotsas. Population balance model for drying of droplets containing aggregating nanoparticles. *AIChE Journal*, 58(11):3318–3328, November 2012.

[4] Y. Cengel. *Heat transfer, A pratical approach.* Mcgraw-Hill, 2004.

[5] `https://cfd.direct/openfoam/user-guide/` Christopher J. Greenshields. *OpenFOAM User Guide.*

[6] Clayton T. Crowe. *Multiphase flow handbook.* CRC Press, Taylor & Francis Group, 2006.

[7] T. A. G. Langrish D. J. E. Harvie and D. F. Fletcher. A computational fluid dynamics study of a tall-form spray dryer. *IChemE*, 80, September 2002.

[8] Irene Borde David Levi-Hevroni, Avi Levy. Mathematical modelling of drying of liquid solid slurries in steady state one-dimensional flow. *Drying Technology*, 13:1187–1201, 1995.

[9] D.J.E. Harvie T.A.G. Langrish J.J. Nijdam J.Williams D.F. Fletcher, B. Guo. What is important in the simulation of spray dryer performance and how do current cfd models perform? *Applied Mathematical Modelling*, 30:1281–1292, 2006.

[10] Alberto F. Scarpettini Edgardo A. Moyano. Numerical stability study and error estimation for two implicit schemes in a moving boundary problem. *Numerical Methods for Partial Differential Equations*, 16:42–61, 2000.

[11] S. E. Elghobashi. On predicting particle-laden turbulent flows. *Applied Scientific Research*, 52:309–329, 1994.

[12] M. Darwish F. Moukalled, L. Mangani. *The Finite Volume Method in Computational Fluid Dynamics, An Advanced Introduction with* OpenFOAM® *and* Matlab®. Springer, 2016.

[13] P.J.A.M. Kerkhof F.G. Kieviet. Air flow, temperature and humidity patterns in a co-current spray dryer: modelling and measurements. *Drying Technology*, 15:1763–1773, 1997.

[14] Dr. Franjo Juretic. *cfMesh User Guide*. Creative Fields, Ltd, 1.1 edition, May 2015.

[15] F.G. Kieviet. *Modelling quality in spray drying*. PhD thesis, Techniche Universiteit Eindhoven, 1997.

[16] A.S. Mujumdar L. Huang, K. Kumar. A parametric study of the gas flow patterns and drying performance of co-current spray dryer: results of a computational fluid dynamics study. *Drying Technology*, 21:957–978, 2003.

[17] Jesse Liberty. *C++*. Sams, 1999.

[18] E. Loth. Numerical approaches for motion of dispersed particles, droplets and bubbles. *Progress in Energy and Combustion Science*, 26(3):161–223, 2000.

[19] P.J. Heggs M.Ghadiri V.F. Gracia A. Bayly D. Djurdjevic H. Ahamadian L. Martin de Juan M. Ali, T. Mahmud. Cfd modelling of a counter-current spray drying tower. *8 International Conference on Multiphase Flow*, pages 26–31, May 2013.

[20] I. Borde M. Mezhericher, A. Levy. Theoretical drying model of single droplets containing insoluble or disssolved solids. *Drying Technology*, 25(6):1035–1042, 2007.

[21] I. Borde M. Mezhericher, A. Levy. Heat and mass transfer of single droplet/wet particle drying. *Chemical Engineering and Processing*, 63:12–23, 2008.

[22] I. Borde M. Mezhericher, A. Levy. Modelling of particle breackage during drying. *Chemical Engineering and Processing*, 47:1404–1411, 2008.

[23] F.R. Menter. Two-equation eddy-viscosity turbulence models for engineering applications. *AIAA Journal*, 32(8):1598–1605, August 1994.

[24] M. Mezhericher. *Drying of Slurries in Spray Dryers*. PhD thesis, Ben-Gurion University of the Negev, September 2008.

[25] M. Necati Ozisik. *Heat conduction*. John Wiley Sons, 1993.

[26] J. Vodnik S. Nesic. Kinetics of droplet evaporation. *Chemical Engineering Science*, 46(2):527–537, 1991.

[27] D.F. Fletcher T.A.G. Langrish. Spray drying of food ingredients and applications of cfd in spray drying. *Chemical Engineering and Processing*, 40:345–354, 2001.

[28] H. Tennekes and J. L. Lumley. *A first course in turbulence*. MIT Press, 1972.

[29] `http://cfd.direct/openfoam/user-guide/thermophysical/`. *OpenFOAM User Guide: 7.1: Thermophysical models*.

[30] David C. Wilcox. *Turbulence modelling for CFD*. DCW Industries, 2006.