João Oliveira Cardoso

Licenciado em Ciências da Engenharia de Materiais

Modeling the reverse segregation of reinforcements in functionally graded composites by the longitudinal casting method

Dissertação para obtenção do Grau de Mestre em

Engenharia de Materiais

Orientador: Professor Doutor Alexandre José da Costa Velhinho,

Professor Auxiliar, FCT/UNL

Júri:

Presidente:	Professor Doutor João Paulo Miranda Ribeiro Borges
Arguente(s):	Professor Doutor Rui Jorge Cordeiro Silva
Vogal(ais):	Professor Doutor Alexandre José da Costa Velhinho

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Setembro 2016

Acknowledgments

For all his support, patience and knowledge provided, I would like to thank my advisor, professor Alexandre Velhinho. I am very happy to have had the opportunity to work with him and look forward to continue in the future.

Secondly, I'd like to thank professor Pedro Medeiros, not only for his help with the computer provided for the simulations ran in this dissertation, but also for his help in my previous work.

I would like to thank all my teachers during my degree, for their knowledge transmitted, their kindness and everything they've done. It has truly been a blessing to have such great teachers as my guides.

I would like to thank my faculty and everyone in it, they housed me for all these years during my studies, and before that in Clubemath; in the same note, I would like to give a special thanks to my dear professor Nelson Chibeles, a true gem of a person and teacher.

I would like to thank all my friends; you guys give life that extra sweetness and make some of the worst days into the best of moments. If there's anything you need from me in the future, I'm here.

To my family, I'd like to thank all of them for their support, and for giving me the opportunity to follow my studies, without them, I literally wouldn't be here. A special mention to my parents, for helping me in this and all my journeys.

Lastly I would like to thank my girlfriend, Mariana, dearest to my hearth. May we one day be together and not an airplane away.

Resumo

Nesta dissertação foi simulada a produção de um compósito de matriz metálica com gradiente de funcionalidade de Al/SiC, fabricado usando o método de fundição centrífuga longitudinal. Este método permite a produção de geometrias que outras variantes da fundição centrífuga não permitem, tornando-o num método bastante vantajoso. Contudo, a fundição centrífuga longitudinal carece de atenção por parte da literatura, nomeadamente relativo aos fenómenos de segregação: inversa, dimensional e transversa.

Desenvolveu-se um código em MATLAB para o estudo da segregação inversa utilizando o método das diferenças finitas com base numa malha estática. Deste estudo foi possível observar a relação entre a temperatura de vazamento e a distribuição final de partículas. Uma simulação adicional foi feita considerando uma distribuição de partículas variado visando estudar brevemente o efeito da segregação dimensional.

Palavras-chave: Materiais com gradiente de funcionalidade, fundição centrifuga longitudinal, método das diferenças finitas

Abstract

In this dissertation, we simulate the production of an Al/SiC functionally graded metal matrix composite, made using the process of longitudinal centrifugal casting. This specific methodology allows the production of geometries that aren't possible with other variants of the centrifugal casting method, making it highly desirable; however, the longitudinal variant has not attracted a wide attention in the literature, namely in regards to the phenomena of reverse, dimensional and transverse segregation.

A MATLAB code was developed to study the effects of reverse segregation; a fixed grid finite difference method was used for this which resulted in the study of the relation between starting casting temperatures and final particle distributions. An additional simulation was made considering a varied particle distribution to briefly study dimensional segregation.

Keywords: Functionally graded materials, Longitudinal centrifugal casting, finite difference method

Contents

List of Figures

List of Tables

List of Abbreviations, Acronyms and Symbols

Abbreviations and Acronyms:

Al – Aluminum

CC – Centrifugal casting

FDM – Finite difference method

FEM – Finite element method

FGM – Functionally graded materials

FGMMC – Functionally graded metal matrix composites

FVM – Finite volume method

J – Coefficient of the cell in study

J_- – Coefficient of the previous cell in study, regarding the x-coordinates

J_+ – Coefficient of the next cell in study, regarding the x-coordinates

K_- – Coefficient of the previous cell in study, regarding the y-coordinates

K_+ – Coefficient of the next cell in study, regarding the y-coordinates

L_- – Coefficient of the previous cell in study, regarding the z-coordinates

L_+ – Coefficient of the next cell in study, regarding the z-coordinates

LCC – Longitudinal centrifugal casting

M – Heat Equations' coefficient matrix

RCC – Radial centrifugal casting

rpm – Rotations per minute

S – Position of interest in the M matrix

SiC – Silicon Carbide

TOL – Tolerance

TRM – Temperature recovery method

Symbols:

$\langle a \rangle$ – Average value of a

$\Delta(x,y,z)$ – Cell size in the x, y or z direction in m

Δt – Time step

μ_l – Aluminum melt viscosity in kg/m s

C_p – Specific heat in J/kg K

d – Particle size

D_v – Mean particle size in µm

H – Enthalpy in J/kg

$\boldsymbol{j}_T$ – Fourier's Law for diffusive heat flux

K – Thermal conductivity in W/m K

L – Latent heat of fusion in J/kg

R – Distance from the rotational center and the particle

T – Temperature in K

T_e – Mold temperature in K

T_m – Melting point of Al in K

t_{max} – Spin-up-time in s

v – Velocity in m/s

ε – Phase fraction in V/V

ρ – Density in Kg/m^3

ω – Angular velocity in rad/s^{-1}

Subscripts:

$\rightarrow V$ – Entering the current control volume

j, k, l – x, y, z components

L – Liquid phase

P – Particle phase

S – Solid phase

$V \rightarrow$ – Exiting the current control volume

Superscripts:

t – Current time step

$t+1$ – Next time step

1 Introduction

Centrifugal casting (CC) is a technique that can be used in the production of functionally graded metal matrix composites (FGMMC), a type of functionally graded materials (FGM). However, CC may be obtained with two distinct methodologies: longitudinal centrifugal casting (LCC) and radial centrifugal casting (RCC). LCC allows for more varied geometries to be produced but is not as much used as RCC because RCC is much easier to study and implement, despite its restrictions. We expect this work to be a further contribution to the production of LCC FGMMC.

1.1 FGM and FGMMC

FGM are an advanced class of materials whose properties vary along at least one axis. Although very ancient, this concept has attracted materials scientists' attention after Japan's space plane project in the mid 1980's, it was afterwards popularized in the rest of the world. The idea behind FGMs is to have a property gradient along an axis. In the case of a composite, by mixing two (or more) materials in a single matrix/reinforcement relation we can have the functionality gradient while avoiding interface problems [1]. Within the category of FGM, we have FGMMC, composites (typically) made from metal and ceramic constituents (Figure 1).

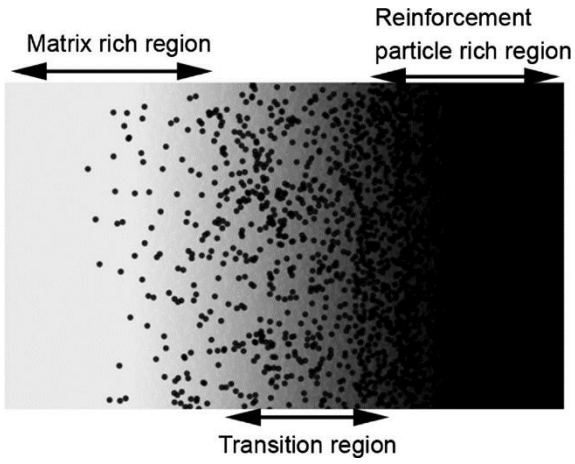


Figure 1 – Schematic illustration of a FGMMC with continuously graded microstructure. Adapted from [2]

There are several methods to produce FGMMC: infiltration, sedimentation/settling, centrifugal casting, sequential casting, spray casting, powder metallurgy, slurry casting, and several deposition techniques [2]–[6]. Within all casting techniques two of them, RCC and LCC (Figure 2), allow the production of materials with better properties than the rest, namely: a denser structure, reduced porosity, grains radially oriented giving superior strength, fracture toughness and

corrosion resistance; these properties make centrifugal casting the most sought process for the production of functionally graded components [2].

RCC has already been extensively studied and is the major method used to produce FGMMC [2], but is limited to the fabrication of radial geometries parts: tubes, poles, beams, pipes, etc... LCC, on the other hand, allows for more complex shapes but isn't as studied as its radial counterpart. Designing parts using either technique is a complex problem to solve since the equations associated with advection (particle displacement), temperature equilibrium and solidification are non-linear and inter-dependent, making numerical methods not only appropriate but also a necessity. Luckily in our modern age, computers have become powerful and common, making such calculations possible in viable amount of time.

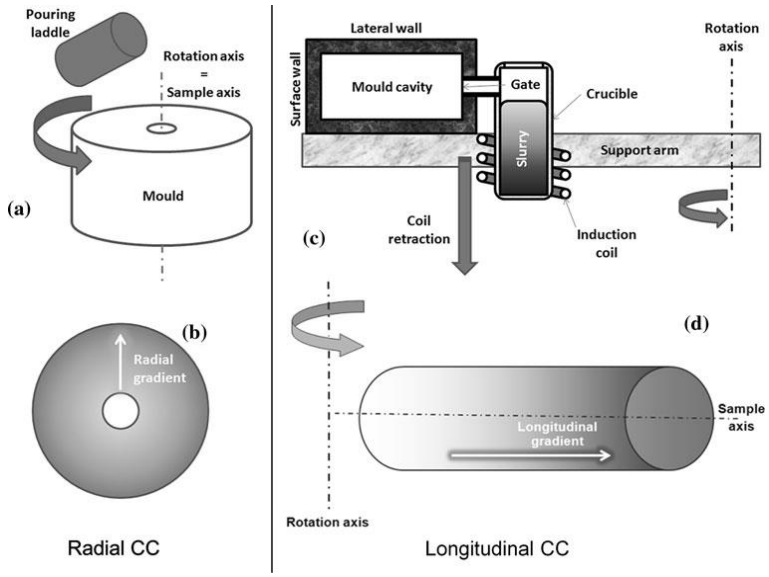


Figure 2 – Illustration of centrifugal casting. a) RCC furnace; b) gradient orientation in a FGMMC processed by RCC; c) LCC furnace; d) gradient orientation in a FGMMC processed by LCC. Adapted from [9]

1.2 Modeling

Numerical methods have been around for a long time; with the invention of the computer their use was increased. Recently, with the ever-increasing power of day to day computers, numerical methods have become popularized and of quick and simple access. Three methods are nowadays available: Finite element method (FEM), finite volume method (FVM) and finite difference method (FDM). The first being one of the most popular and the last one of the simplest. The FDM is a fixed grid technique with a defined mesh (amount of nodes in a grid and their arrangement), it qualifies as the best method for numerically solving partial differential equations [7], [8] since it is easy to apply and fast to run, its only limitations being that it is restricted to simple geometries.

In the FDM, the coefficients used can be calculated using an: explicit, implicit or mixed (Also known as Crank-Nicolson) method/scheme. This means that if the approximations are made using forward, backward or central approach, they vary in accuracy and amount of calculation required from smaller to greater, respectively. Table 1 shows the FDM discretization of Eq. (1) Figure 3 shows the FDM node stencil for the unidimensional normalized heat equation.

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

Table 1 –FDM coefficient calculation example for the unidimensional normalized heat equation

Forward (Explicit)	$\frac{T_x^{t+1} - T_x^t}{\Delta t} = \frac{T_{x+1}^t - 2T_x^t + T_{x-1}^t}{\Delta x^2}$
Backward (Implicit)	$\frac{T_x^{t+1} - T_x^t}{\Delta t} = \frac{T_{x+1}^{t+1} - 2T_x^{t+1} + T_{x-1}^{t+1}}{\Delta x^2}$
Crank-Nicolson (Central Method)	$\frac{T_x^{t+1} - T_x^t}{\Delta t} = \frac{1}{2}\left(\frac{T_{x+1}^{t+1} - 2T_x^{t+1} + T_{x-1}^{t+1}}{\Delta x^2} + \frac{T_{x+1}^{t+1} - 2T_x^{t+1} + T_{x-1}^{t+1}}{\Delta x^2}\right)$

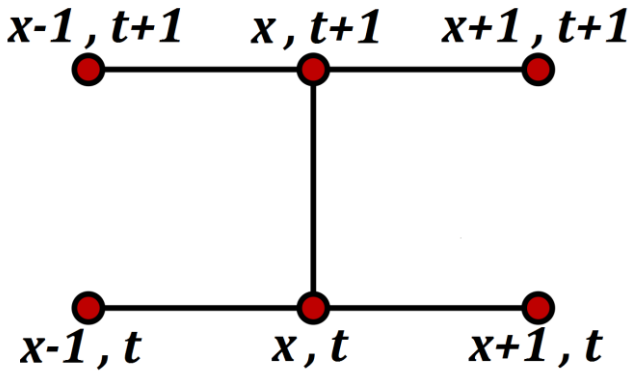


Figure 3 – FDM stencil for the heat equation using the Crank-Nicolson method. Adapted from [22]

The problems associated with this type of modeling have been studied for the RCC, it is the LCC that requires attention and the main difference is symmetry, for RCC we can simplify our problem by considering a unidimensional or bi-dimensional problem. Only in the latter case can we start to see transverse segregation but most articles consider only the first case. With LCC we must consider a tri-dimensional problem, not only increasing in complexity but also in sheer data volume. Velhinho *et al.* [9], [10] stated the problems related to this modeling and Gonçalo *et al.* [11] studied transverse segregation. From these studies resulted that there are three types of segregation to study: reverse, dimensional and transverse segregation.

1.3 Reverse Segregation:

Reverse segregation is a phenomenon in which the reinforcement particle content Of a FGM (measured by the particle area fraction in Figure 4), does not decrease monotonically from the part's surface to its interior (Figure 4 a), instead showing a maximum at some inner position (Figure 4 b), a reverse gradient, unlike what should be expected for particle motion according to Stokes' equation. This happens due to the solidification front, it stops particle motion and can even push particles back, both contributing factors for this gradient formation. Particle engulfment is negligible for radiuses at the micro scale [9], [12].

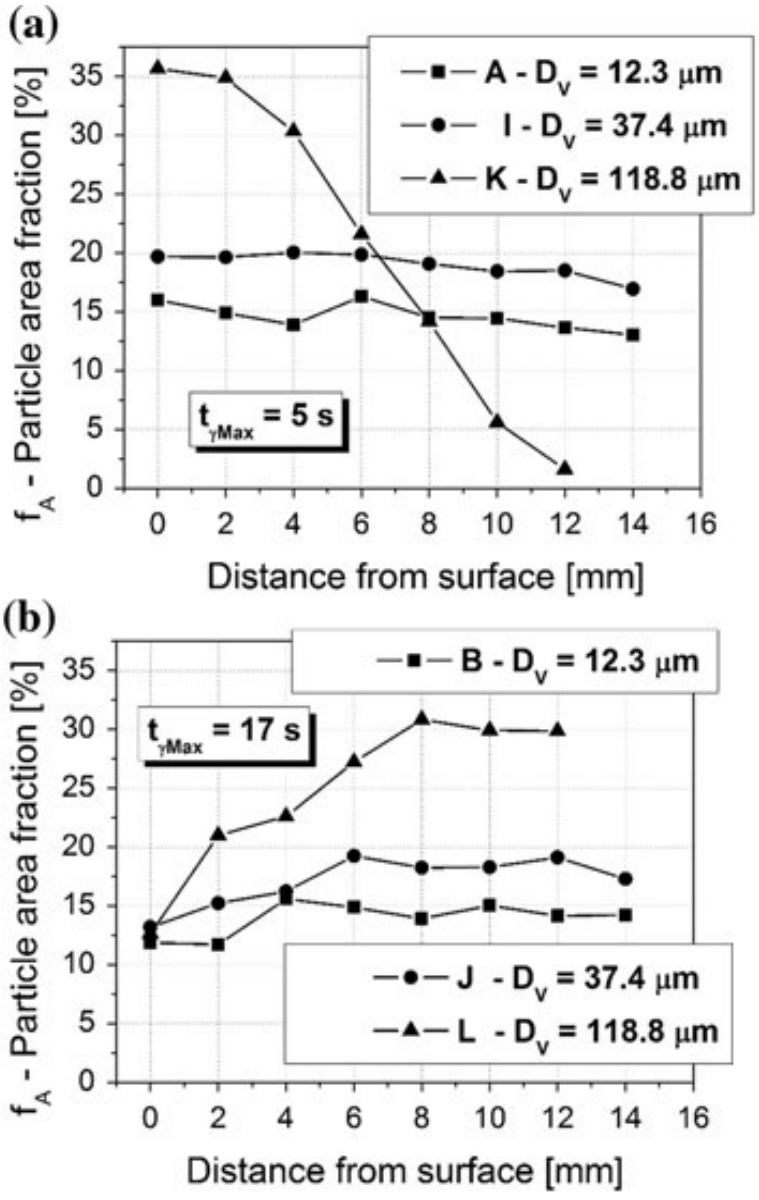


Figure 4 – Particle area fraction profile for FGMMC cast by LCC of Al/SiC. a) Spin-up time of 5s; b) Spin-up time of 17s. D_v is the mean particle size; A, I, K, B, J, L were experiments made; 0 is the furthest point from the centrifugal axis. Adapted from [9]

1.4 Dimensional Segregation:

As with reverse segregation, we observe that instead of larger particles occupying positions near the surface, smaller particles that take this position (Figure 5 b)). Centrifugal acceleration is higher for larger particles; it was expected that the opposite would happen. The effect happens due to the solidification front; the dendritic growth of Al crystals during solidification blocks larger particles more easily than smaller ones, and in some cases, allows smaller particles to travel further towards the surface.

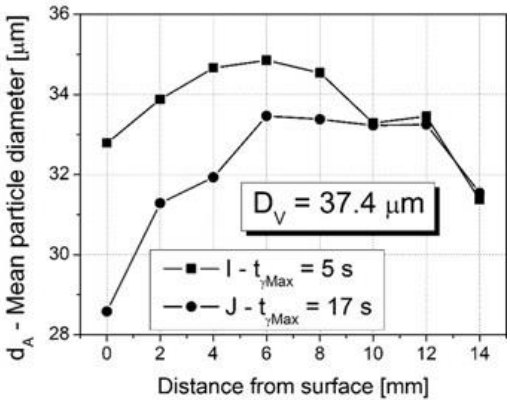


Figure 5 – Mean particle diameter longitudinal profile. Adapted from [9]

1.5 Transverse segregation

The final problem not considered in RCC is transverse segregation. Transverse segregation is the effect of particle variation along the axis perpendicular to its movement. This occurs because the solidification front is faster around the surface then in the parts core, causing particles to stop earlier than the ones closer to the center, therefore causing a transverse gradient; particle pushing also helps in the formation of such gradient but not as strongly. [9]–[11]

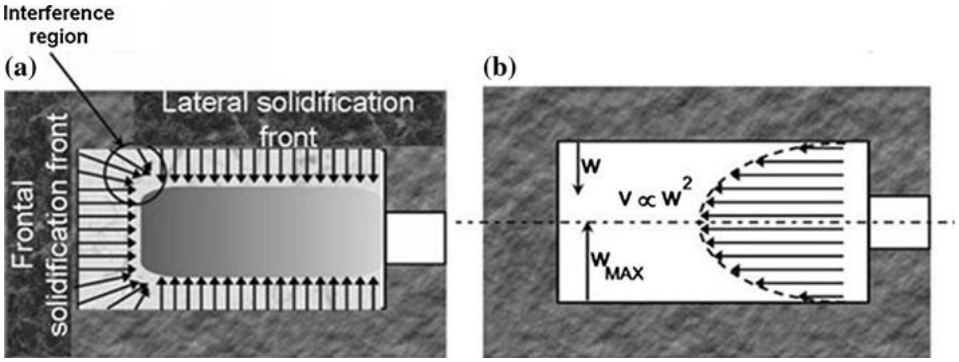


Figure 6 – Transverse segregation. Adapted from [9]

1.6 Synthesis

We now have all the necessary information to simulate the production of an Al/SiC FGMMC produced by LCC using a three-dimensional mesh. This type of mesh isn't commonly used in calculations, as problems tend to be simplified towards bi-dimensional or unidimensional solutions. In the next chapters, we will explain the nuances related to the application of FDM, namely: equation solving order, parameters used and assumptions made. Latter we explain in detail how each equation is solved, before presenting and analyzing some simulation results for particular cases of interest.

2 Numerical Model

In this section, how the computer model works, all the equations used and iterative method are explained.

2.1 Initialization

At the beginning of each time step we take the values of particle fraction (FP), solid fraction (FS) and temperature from the previous time step (initial conditions for the first time step) and use them for this step. On each subsequent iteration, we use the FS obtained in the previous iteration and feed it until convergence is achieved. Liquid fraction (FL) is obtained at each iteration using both the FP and FS.

2.2 Particle Movement

At each time step, we calculate the apparent particle velocity based on the equations of Gao et al. [15], this equation was derived from Stoke's equation for particle motion:

$$\boldsymbol{v_p} = \frac{(\rho_P - \rho_L)\varepsilon_P \omega^2 R d^2 (1-\varepsilon_P)^{4,65}}{18\mu_l(1-\varepsilon_L)(\varepsilon_P+\varepsilon_L)^2} \quad \textbf{(2)}$$

Where angular velocity ω is:

$$\omega = \frac{2\pi * rpm}{60} \quad \textbf{(3)}$$

At this point, maximum particle clustering and the solidification front are not considered.

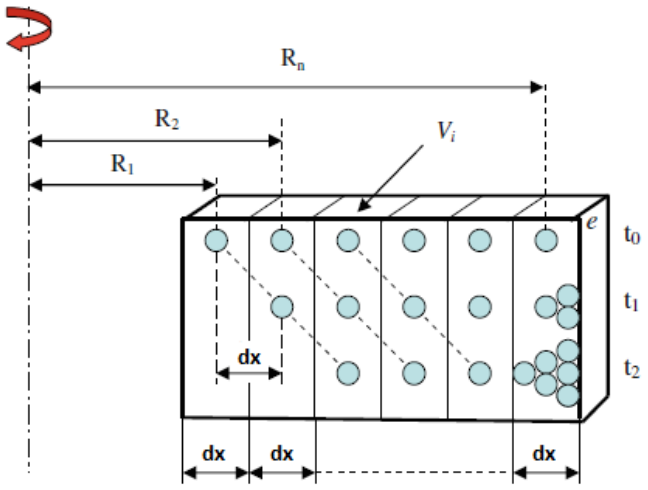


Figure 7 – Particle Movement. Adapted from [16]

We use particle velocity v_p to calculate particle displacement based on the article of B.Balout *et al.* [16] (Figure 7). The method functions by if the particle fraction in any time step

is given by the number of particles in that time step, minus the particles leaving plus the particle entering:

$$\varepsilon_{jkl}^{t+1} = \varepsilon_{jkl}^{t} - \varepsilon_{V\rightarrow}^{t} + \varepsilon_{\rightarrow V}^{t} \quad \textbf{(4)}$$

The amount of particle leaving a control volume is the same entering the control volume next to it. The particle displacement can be determined by multiplying the velocity by the time step and dividing it by the cell size in the x component. If we multiply that by the number of particles in that volume, we get the particle fraction that exits that volume:

$$\varepsilon_{V\rightarrow}^{t} = \frac{\varepsilon_P v_P \Delta t}{dx} \quad \textbf{(5)}$$

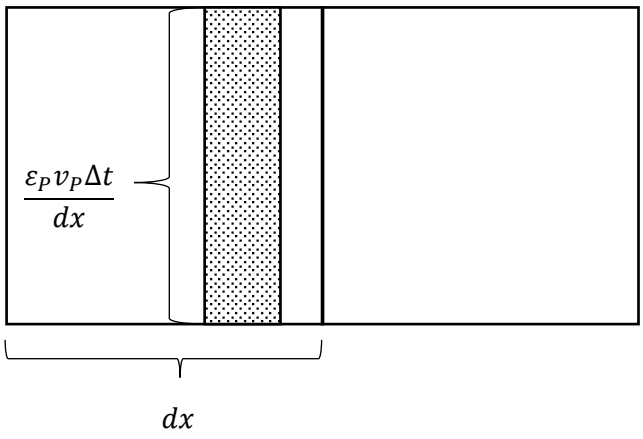


Figure 8 – Particle movement

The particle clustering is limited to 52%, value given by Watanabe *et al.* [17], value obtained for the Al/SiC system. We also assume the particles stop upon reaching the solidification front. A consequence of this method though is that particles can only move from one cell to the next, following this rule:

$$v_P \Delta t < dx \quad \textbf{(6)}$$

This means that:

$$\Delta t < \frac{dx}{v_P} \quad \textbf{(7)}$$

Liquid fraction are now calculated using the constitutive equation:

$$\varepsilon_S + \varepsilon_L + \varepsilon_P = 1 \quad \textbf{(8)}$$

For the next time step we will need all the previous calculations to the best of detail. This means that the particle velocity equation used before isn't accurate enough, but we know the actual particle movement so we can simply calculate the actual particle movement by comparing the previous particle fraction vector with the current one

2.3 Temperature Field

We now have all the basis used to update the temperature field. For that we use the heat equation given by *Rappaz* et. al. [18]:

$$\frac{\partial}{\partial t}(\rho H) + div(\rho H v) + div(j_T) = \dot{Q}_T \tag{9}$$

In which the first term $\frac{\partial}{\partial t}(\rho H)$ is associated with the energy stored in the material, the second term $div(\rho H v)$ with the energy present in moving phases (particle and liquid), the third term $div(j_T)$ is the *Fourier's Law* for a diffusive heat flux and the last is the energy transferred into or out of the system.

The last term $\dot{Q}_T$, related to heat supplied or taken from the system by inside agents is considered zero for the purposes of finding the thermal equilibrium.

The latent heat of fusion *L* is only considered when calculating solidification.

The rest of the equation results in the following (note that particle movement is only considered along the x-axis):

$$\begin{aligned}\frac{\partial}{\partial t}\left[\left(\rho_s \varepsilon_s C_{Ps} + \rho_P \varepsilon_p C_{PP} + \rho_L \varepsilon_L C_{PL}\right)T\right] + \frac{\partial}{\partial x}\left[\left(\rho_P \varepsilon_P C_{PP} \boldsymbol{v}_P + \rho_L \varepsilon_L C_{PL} \boldsymbol{v}_L\right)T\right] \\ = \frac{\partial}{\partial x}\left[\left(\varepsilon_S K_S + \varepsilon_P K_P + \varepsilon_L K_L\right)\frac{\partial T}{\partial x}\right] + \frac{\partial}{\partial y}\left[\left(\varepsilon_S K_S + \varepsilon_P K_P + \varepsilon_L K_L\right)\frac{\partial T}{\partial y}\right] \\ + \frac{\partial}{\partial z}\left[\left(\varepsilon_S K_S + \varepsilon_P K_P + \varepsilon_L K_L\right)\frac{\partial T}{\partial z}\right]\end{aligned} \tag{10}$$

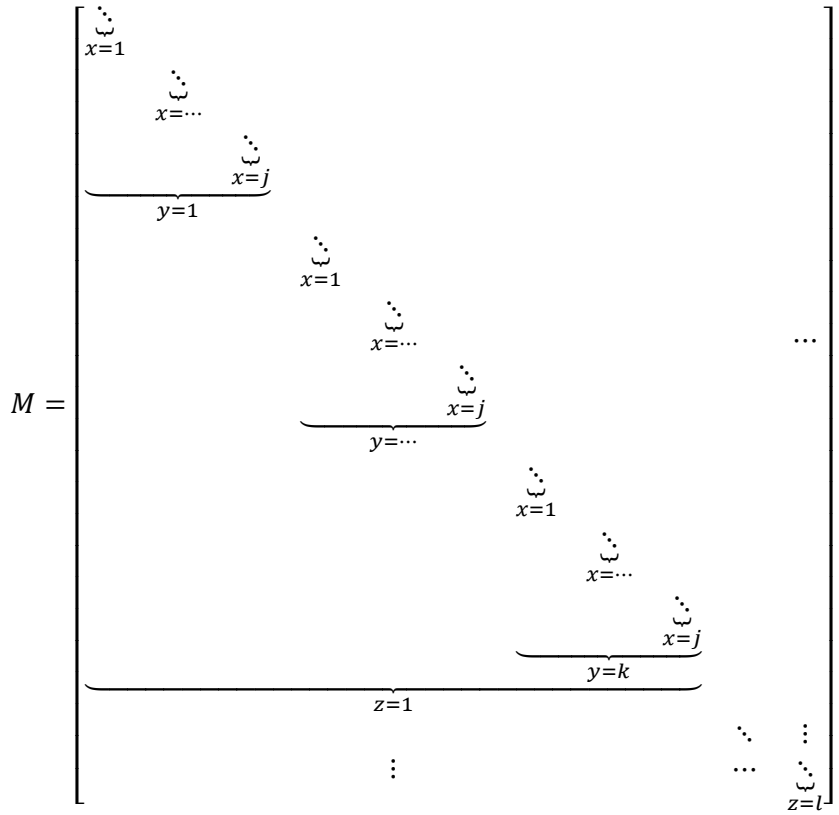
We solve the equation in order of T using a central difference scheme for the FDM coefficients and arrive at an equation of the following form:

$$T^t = T^{t+1}J + T_{j+1}^{t+1}J_+ + T_{j-1}^{t+1}J_- + T_{k+1}^{t+1}K_+ + T_{k-1}^{t+1}K_- + T_{l+1}^{t+1}L_+ + T_{l-1}^{t+1}L_- \tag{11}$$

For simplicity and time saving we assume a Dirichlet Boundary Condition of $T = T_e$, T_e being the mold temperature. We arrange the coefficients in the matrix and multiply it by the current temperature field:

$$T^{t+1} = M \backslash T^t \tag{12}$$

In which T is a vector of size $x * y * z$ and M is a sparse square matrix of dimension $(x * y * z)$ per $(x * y * z)$, in which x, y and z are the mesh sizes in each direction. The matrix organization is the following:



For a given jkl coordinate, we determine the corresponding row:

$$S = j + (k-1)x + (l-1)xy \quad \textbf{(13)}$$

In this row, we organize the constants' information per the following table:

Table 2 – Coefficients' position.

Coefficient	Row
J	S
J_+	$S+1$
J_-	$S-1$
K_+	$S+x$
K_-	$S-x$
L_+	$S+xy$
L_-	$S-xy$

For a $3 * 3 * 3$ system, the result would be a 27 by 27 sparse matrix where the only position not in contact with any surface would be position $2 * 2 * 2$ located in the 14^{th} row:

$$\begin{bmatrix} \underbrace{\overbrace{\cdots}^{y=1} \quad \overbrace{\cdots \; L_- \; \cdots}^{y=2} \quad \overbrace{\cdots}^{y=2}}_{z=1} & \underbrace{\overbrace{\cdots \; K_- \; \cdots}^{y=1} \quad \overbrace{J_- \quad J \quad J_+}^{y=2} \quad \overbrace{\cdots \; K_+ \; \cdots}^{y=2}}_{z=2} & \underbrace{\overbrace{\cdots}^{y=1} \quad \overbrace{\cdots \; L_+ \; \cdots}^{y=2} \quad \overbrace{\cdots}^{y=2}}_{z=3} \end{bmatrix}$$

2.4 Enthalpy Field

After estimating the temperature update, we determine the correspondent enthalpy field using the following relation:

$$H^{t+1} = H^t - \int_{T^t}^{T^{t+1}} \langle \rho C_P \rangle \, dT \tag{14}$$

In which $\langle \rho C_P \rangle$ can be seen has average properties in each node $\rho_s \varepsilon_s C_{Ps} + \rho_P \varepsilon_p C_{PP} + \rho_L \varepsilon_L C_{PL}$, simplified gives us:

$$H^{t+1} = H^t - \langle \rho C_P \rangle (T^{t+1} - T^t) \tag{15}$$

This gives us an approximated enthalpy field used to determine the solid update.

2.5 Solidification

There are many methods to simulate a solidification front. They range from front-tracking, source-term and enthalpy formulations. In this work a variation of the latter example is used, the temperature recovery method (TRM); first devised by Tszeng [19].

In the TRM the solidification process used in FDM schemes and FEM, it's formulated based on the temperature and enthalpy fields and is very easy to apply. The first step is to calculate the thermal equilibrium for a given time step using the previous equations, using the relation between temperature and enthalpy [20]:

$$H(T) = \int_{T_r}^{T} \rho C_p dT + \rho L \tag{16}$$

Assuming the reference temperature (T_r) is the alloy's melting point (T_m), with each time step we obtain a ΔT, feeding this into equation (15) we can determine the ΔH that occurred, doing this we can obtain the enthalpy field, furthermore, if we divide the enthalpy variation by the latent heat we obtain the liquid fraction that solidified in this time step (Figure 9).

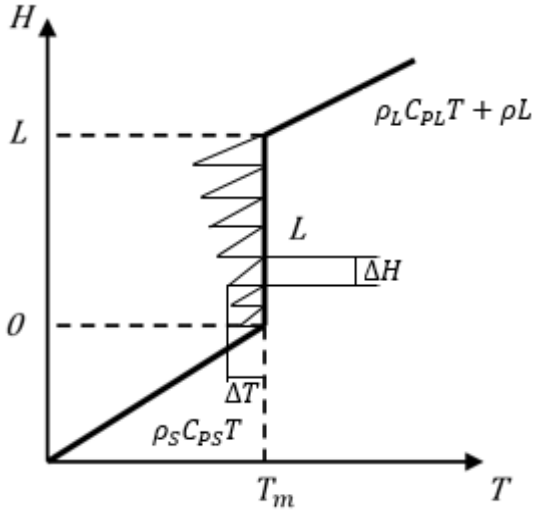


Figure 9 – Enthalpy in function of temperature. Adapted from [23]

2.6 Convergence

Properties vary with difference fractions, as such convergence is necessary. At this step, we save the current solid fraction and iterate the time step again with the updated solid fraction, this step is repeated until the difference between this iterations' enthalpy field minus the previous iterations' is under a certain tolerance, normally $TOL < 10^{-5}$

2.7 Iteration

After convergence in the enthalpy field is achieved, a new time step is calculated using the values calculated in this time step. This process is repeated until particle movement stops.

3 Method

The simulations were made using MATLAB version R2016b in a 64-bit Windows 10 Education operative system with 12,0 GB of RAM and an Intel® Xeon® CPU E5506 @ 2.13GHz 2.13 GHz processor.

In the procedure developed in the present work, the equations were solved iteratively using a fixed-grid, single-domain FDM. The advection equations were discretized using an explicit scheme while an implicit one was used for the heat equation, the coefficients used are calculated using a central scheme. For the solidification, an enthalpy method was used. The equations were solved iteratively in the following manner:

1. Import values from the previous time step
2. Determine particle velocity from Eq. (2)
3. Calculate particle displacement from Eq. (5)
4. Obtain new liquid phase from Eq. (8)
5. Compute the new temperature field T from Eq. (9)
6. Determine the enthalpy field from Eq. (15)
7. Guess solidification using Eq. (16)
8. Iterate until convergence in the enthalpy field

This general procedure was used during each time step. The exception was the first time where the starting values were used. All the simulations were using dimensions of $8 * 4 * 4$ cm, 1000 rpm and a TOL of 10^{-5}.

Despite many attempts, an implicit method for the advection equation wasn't possible to implement, and an explicit method was used instead. The decision to maintain an implicit scheme for the heat equation instead of an explicit one comes from the coefficient restriction of the explicit method. Unlike the implicit method, which is always numerically stable and convergent, the numeric stability and convergence of the explicit method require that the conditions in Table 3 be attained [13]:

Table 3 – Explicit method convergence

Explicit Method	Advection Equation	Heat Equation
$\frac{\Delta t}{\Delta x^2} < \frac{1}{2}$	$\frac{\Delta t}{\Delta x} < \frac{1}{2}$	$\frac{\Delta t}{\Delta x^2} < \frac{1}{2}$

Therefore, maintaining an implicit scheme for the heat equation assures we don't lose time in its implementation.

Table 4 – Different parameters used

Parameters	Experiment A		Experiment B	
Initial Temperature	850ºC		1000ºC	
Particle Size	$10\ \mu m$			
Mesh Size	(80,40,40)			
Time Step	$10^{-4}\ s$			
	Experiment C			
Initial Temperature	850ºC			
Particle Sizes	A 75 μm	B 112,5 μm	C 137,5 μm	D 175 μm
Mesh Size	(40,20,20)			
Time Step	$7 * 10^{-3}\ s$			

The constants used were:

Table 5 – Properties of Al/SiC. Adapted from [14]

Quantity	Symbol	Value	Units
Liquid Density (Al)	ρ_L	2390	kg/m^3
Solid Density (Al)	ρ_S	2550	kg/m^3
Particle Density (SiC)	ρ_P	3200	kg/m^3
Liquid Specific Heat (Al)	C_{PL}	1079.5	$J/kg\ K$
Solid Specific Heat (Al)	C_{PS}	1176.5	$J/kg\ K$
Particle Specific Heat (SiC)	C_{PP}	840	$J/kg\ K$
Liquid Thermal Conductivity (Al)	K_L	95	W/mK
Solid Thermal Conductivity (Al)	K_S	210	W/mK
Particle Thermal Conductivity (SiC)	K_P	16	W/mK
Latent Heat of Fusion (pure Al)	L	$3.97 * 10^5$	J/kg
Melting Temperature (pure Al)	T_m	933.6	K
Mold Temperature	T_e	300	K
Liquid Viscocity (Al)	μ_L	$1.26 * 10^{-3}$	kg/ms

3.1 Assumptions

The equations were formulated and solved per the following assumptions:

- Heat transfer occurs strictly due to conduction
- No energy loss due to radiation or convection
- Thermal equilibrium inside each control volume is reached instantly
- Gravity is neglected
- Solidification front is considered planar
- Angular velocity is constant and is achieved instantly
- Particles reach terminal velocity instantly
- There is no particle pushing by the solidification front
- Particles stop the instant they reach the solidification front
- Particles are non-deformable, spherical and chemically inert
- Initial particle positions are considered uniform across the melt
- There is no change in volume
- Thermophysical properties are considered constant
- There is no porosity

4 Results and Discussion

4.1 Results

Three experiments (Table 4) were made using the above methods. The first two were made using a fine mesh and small time step, resulting in a high-quality surface used to compare two different initial temperatures. The final experiment was ran using four particle sizes at the same time to simulate granulometry, this was made in a lower resolution.3.2.1 Main Experiments

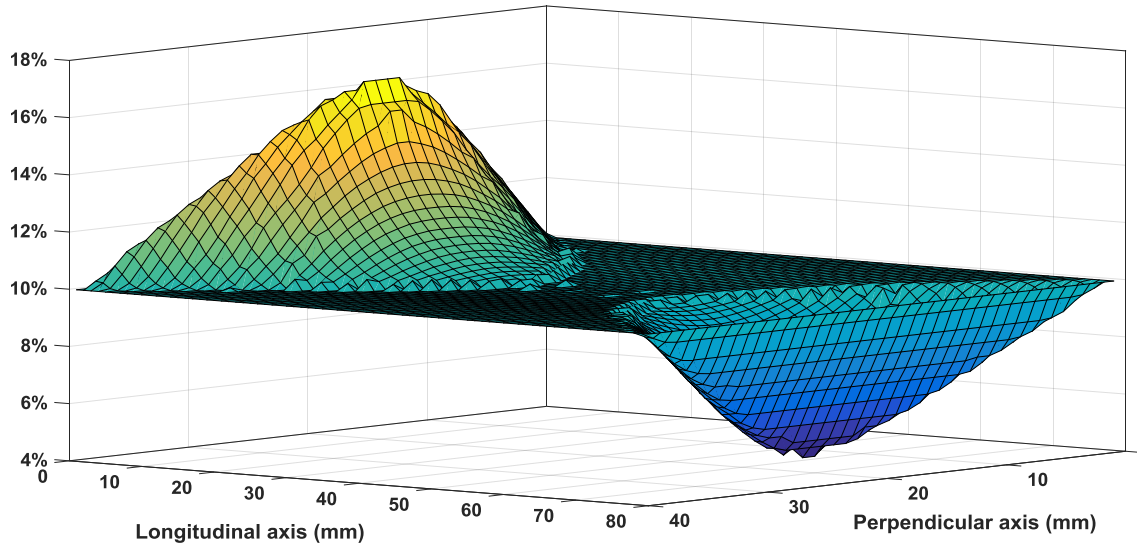


Figure 10 – Percentage surface for particles along the 20mm height. This is a result of experiment A

In the first result clearly observe the effect of transverse segregation.

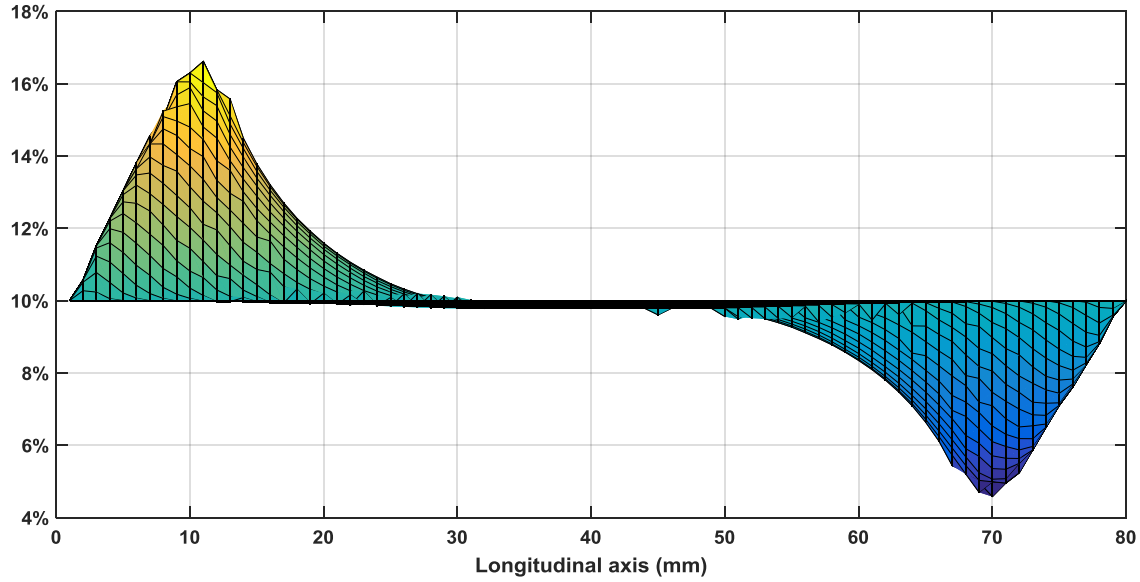


Figure 12 – Side-view of percentage surface for particles along the 20mm height. This is a result of experiment A

In the side-view we can see the phenomenon of severse segregation, even without considering particle pushing. This is a good indicator that particle pushing contributes only slightly to this effect.

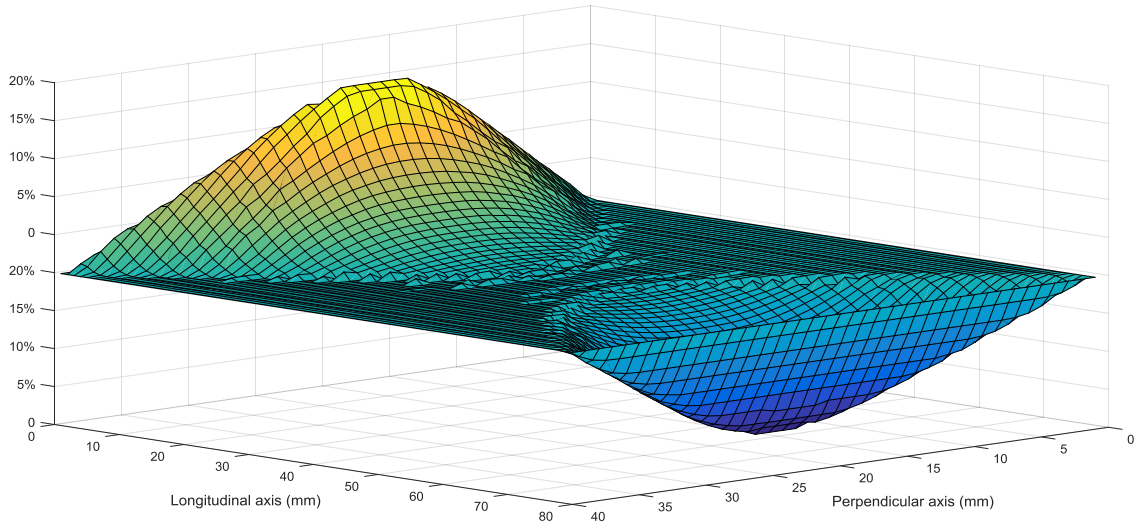


Figure 11 – Percentage surface for particles along the 20mm height. This is a result of experiment B

Similarly to experiment A we can observe the effect of transverse segregation, we can't compare these two figures but both show prominent results.

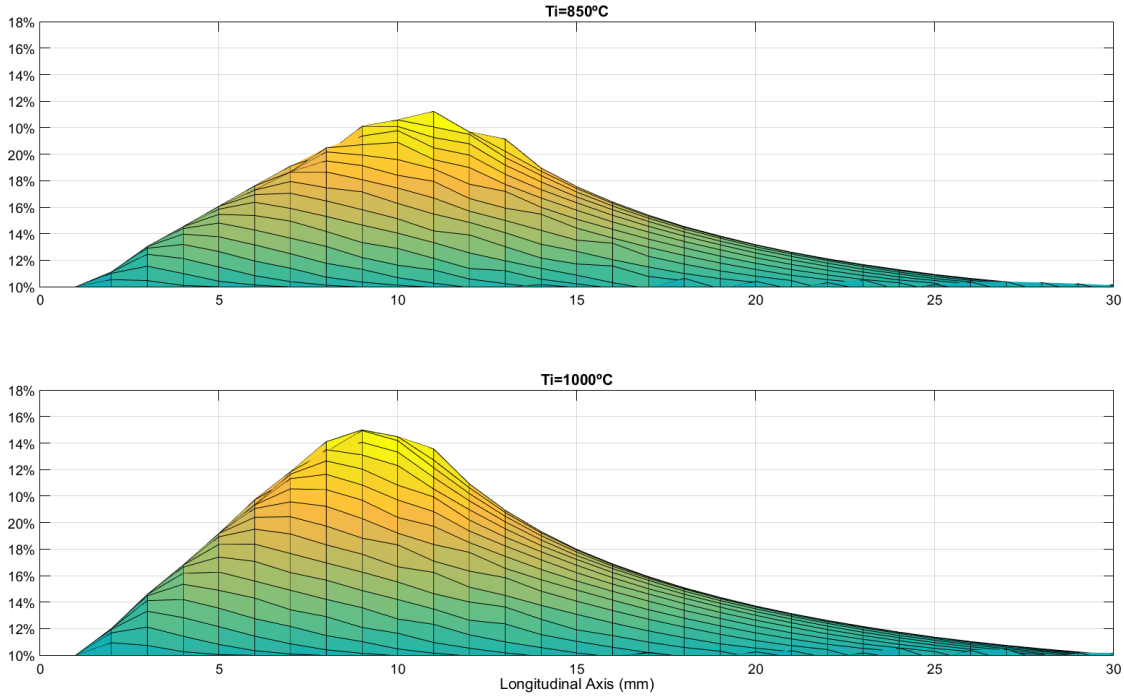


Figure 13 – Percentage surface comparison between experiment A and B.

Comparing both results we can see the effects of different starting conditions, namely, we can observe that for higher starting temperatures the particle distribution peaks closer to the surface, another note is that the particles peak higher than for lower temperatures.

4.2 Effects of reinforcement particle size distribution

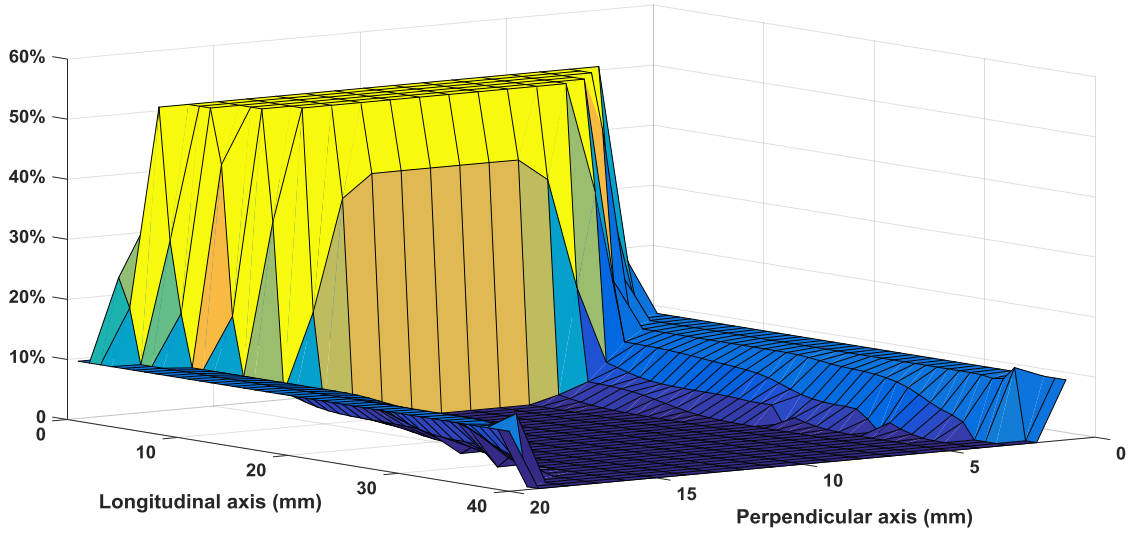


Figure 14 – Percentage surface for particles along the 20mm height. This is a result of experiment C.

We can observe the effect of transverse segregation, even if lessened by the bigger particle sizes. The plateau observed is stuck at 52% because that was the theoretical limit set [17].

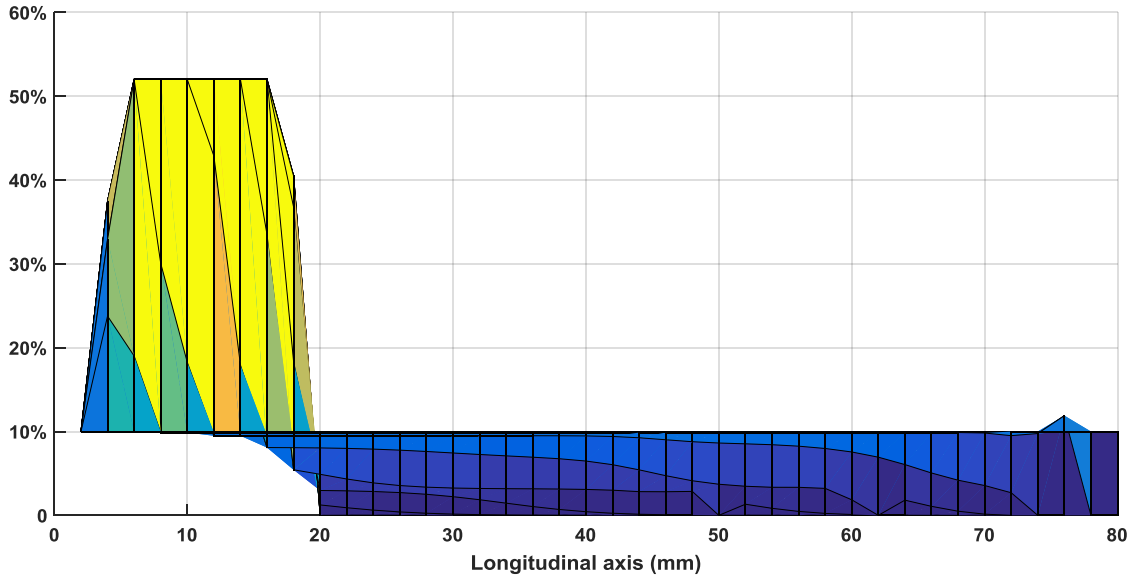


Figure 15 – Side-view of percentage surface for particles along the 20mm height. This result is of experiment C.

We can observe a minor case of reverse segregation, this effect is dulled due to the particle sizes used, the larger they are, smaller the result.

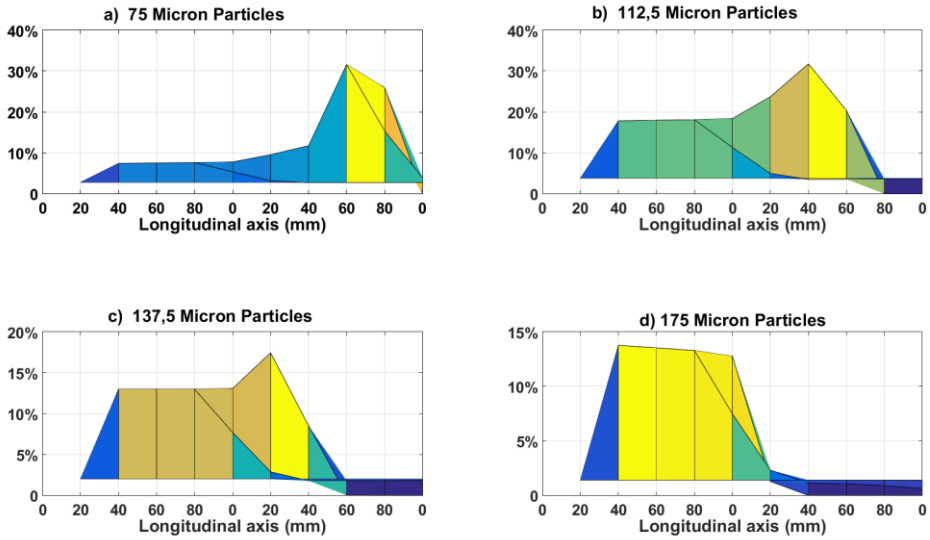


Figure 16 – Side-view of the different particle sizes of experiment C

We can see that bigger particles arrive near the surface faster, occupying space and not letting other particle in there; we can also see that from the smaller to the bigger particles the profiles change showing this effect.

5 Conclusions and Future Work

5.1 Conclusions

A MATLAB code has been developed, capable of simulating the production of any LCC FGMMC. The code is robust, precise and it can simulate a panoply of different starting conditions; however, its main downside concerns the execution time to perform detailed calculations. As such, a low-resolution simulation is advised as a way to predict any problems or the expect time of completion for a high-resolution simulation.

Despite not taking particle pushing into account, we can clearly observe the effect of reverse segregation, either when particle grain size is considered to be homogeneous or when variations of this parameter are taken into account. This shows us that particle impediment is an important contributor for this effect.

We can also see that for a higher starting temperature particles reach further and a higher concentration is obtained. With the model currently operational we can easily study this.

In the simulated results, we can observe that larger particles reach the surface faster than smaller ones, this is expected as larger particles have a higher velocity. What happens in experimental results is that smaller particles reach further than larger one. This means that the effects of particle pushing and non-planar solidification are responsible and critical for the effect of dimensional segregation.

The effect of transverse segregation is clearly visible, which goes according with previous works [10].

5.2 Future Work

For the future, there are two main areas of development: model improvement and speed.

For the model improvement:

- Migrating to a Finite Element Method might be advantageous for complex geometries and commercial applications.
- Using a Neumann boundary condition is more accurate but requires extra calculation time.
- Considering a non-planar solidification front might reveal more insight and flexibility to the process.

For speed:

- The usage of a source-term method might improve computational speed [7].

- The complete change to either implicit or explicit method, depending on the solidification time. The first method is better for transient and timely casts while the second one is better for fast changing processes.

Notes:

- For simplification particle shapes were considered spherical, a correcting factor might be added since in reality the existing sharp edges result in different interface shapes, making the resulting microstructures different than expected [21].

5 Bibliography

[1] R. M. Mahamood, E. T. A. Member, M. Shukla, and S. Pityana, "Functionally Graded Material : An Overview," *World Congr. Eng.*, vol. III, pp. 2–6, 2012.

[2] T. P. D. Rajan and B. C. Pai, "Developments in Processing of Functionally Gradient Metals and Metal–Ceramic Composites: A Review," *Acta Metall. Sin. (English Lett.*, vol. 27, no. 5, pp. 825–838, 2014.

[3] A. Gupta and M. Talha, "Recent development in modeling and analysis of functionally graded materials and structures," *Prog. Aerosp. Sci.*, vol. 79, pp. 1–14, 2015.

[4] A. Bahrami, M. I. Pech-Canul, C. A. Gutierrez, and N. Soltani, "Effect of rice-husk ash on properties of laminated and functionally graded Al/SiC composites by one-step pressureless infiltration," *J. Alloys Compd.*, vol. 644, pp. 256–266, 2015.

[5] M. Pourmajidian and F. Akhlaghi, "Fabrication and Characterization of Functionally Graded Al/SiCp Composites Produced by Remelting and Sedimentation Process," *J. Mater. Eng. Perform.*, vol. 23, no. 2, pp. 444–450, 2014.

[6] T. P. D. Rajan, E. Jayakumar, and B. C. Pai, "Developments in solidification processing of functionally graded aluminium alloys and composites by centrifugal casting technique," *Trans. Indian Inst. Met.*, vol. 65, no. 6, pp. 531–537, 2012.

[7] H. Pointner, A. De Gracia, J. Vogel, N. H. S. Tay, M. Liu, M. Johnson, and L. F. Cabeza, "Computational efficiency in numerical modeling of high temperature latent heat storage : Comparison of selected software tools based on experimental data," *Appl. Energy*, vol. 161, pp. 337–348, 2016.

[8] C. Grossmann, H. Roos, and M. (Martin. Stynes, "Numerical Treatment of Partial Differential Equations," 2007.

[9] A. Velhinho and L. A. Rocha, "Longitudinal centrifugal casting of metal-matrix functionally graded composites : an assessment of modelling issues," *J. Mater. Sci.*, vol. 46, no. 11, pp. 3753–3765, 2011.

[10] A. Velhinho, G. Rodrigues, J. P. Mota, and R. Martins, "Modelling of transverse segregation on centrifugally-cast functionally graded composites," pp. 1–8, 2014.

[11] G. da C. Rodrigues, "Lisboa 08/2011," Universidade Nova de Lisboa, 2011.

[12] X.-H. Chen and H. Yan, "Solid–liquid interface dynamics during solidification of Al 7075–Al2O3np based metal matrix composites," *Mater. Des.*, vol. 94, pp. 148–158, 2016.

[13] J. Crank, "THE MATHEMATICS OF DIFFUSION," 1975.

[14] J. W. Gao and C. Y. Wang, "Modeling the solidification of functionally graded materials by centrifugal casting," *Mater. Sci. Eng. A*, vol. 292, no. 2, pp. 207–215, 2000.

[15] J. W. Gao and C. Y. Wang, "Transport Phenomena During Solidification Processing of Functionally Graded Composites by Sedimentation," *J. Heat Transfer*, vol. 123, no. 2, p. 368, 2001.

[16] B. Balout and J. Litwin, "Mathematical Modeling of Particle Segregation During Centrifugal Casting of Metal Matrix Composites," *J. Mater. Eng. Perform.*, vol. 21, no. 4, pp. 450–462, 2012.

[17] Y. Watanabe, N. Yamanaka, and Y. Fukui, “Control of composition gradient in a metal-ceramic functionally graded material manufactured by the centrifugal method,” *Compos. Part A Appl. Sci. Manuf.*, vol. 29, no. 5–6, pp. 595–601, 1998.

[18] M. Rappaz, M. Bellet, and M. Deville, *Numerical Modeling in Materials Science and Engineering*, vol. 32. 2003.

[19] T. C. Tszeng, Y. T. Im, and S. Kobayashi, “Thermal analysis of solidification by the temperature recovery method,” *Int. J. Mach. Tools Manuf.*, vol. 29, no. 1, pp. 107–120, 1989.

[20] V. R. Voller and C. R. Swaminathan, “FIXED GRID TECHNIQUES FOR PHASE CHANGE PROBLEMS : A REVIEW,” vol. 30, pp. 875–898, 1990.

[21] E. M. Agaliotis, M. R. Rosenberger, a. E. Ares, and C. E. Schvezov, “Influence of the Shape of the Particles in the Solidification of Composite Materials,” *Procedia Mater. Sci.*, vol. 1, pp. 58–63, 2012.

[22] G. Biswas, “Finite difference method,” vol. 6, no. 10. p. 9685, 2011.

[23] B. Mochnacki, S. Lara, and E. Pawlak, “APPLICATION OF THE ENTHALPY APPROACH,” vol. 1, no. 1, pp. 163–176, 2002.

Appendix 1: Heat Equation Deduction

Constituent Equation:

$$\varepsilon_S + \varepsilon_L + \varepsilon_P = 1 \tag{17}$$

Average thermal conductivity:

$$\langle K \rangle = \varepsilon_S K_S + \varepsilon_P K_P + \varepsilon_L K_L \tag{18}$$

Average volumetric specific heat:

$$\langle \rho C_p \rangle = \rho_s \varepsilon_s C_{Ps} + \rho_P \varepsilon_p C_{PP} + \rho_L \varepsilon_L C_{PL} \tag{19}$$

The heat equation used at each time step is given by *Rappaz* et. al. [1]:

$$\frac{\partial}{\partial t}(\rho H) + div(\rho H v) + div(j_T) = \dot{Q}_T \tag{9}$$

Extending all terms:

$$\begin{aligned} &\frac{\partial}{\partial t}\left[\left(\rho_s \varepsilon_s C_{Ps} + \rho_P \varepsilon_p C_{PP} + \rho_L \varepsilon_L C_{PL}\right)T\right] + \frac{\partial}{\partial x}\left[\left(\rho_P \varepsilon_P C_{PP} \boldsymbol{v}_P + \rho_L \varepsilon_L C_{PL} \boldsymbol{v}_L\right)T\right] \\ &= \frac{\partial}{\partial x}\left[\left(\varepsilon_S K_S + \varepsilon_P K_P + \varepsilon_L K_L\right)\frac{\partial T}{\partial x}\right] + \frac{\partial}{\partial y}\left[\left(\varepsilon_S K_S + \varepsilon_P K_P + \varepsilon_L K_L\right)\frac{\partial T}{\partial y}\right] \\ &\quad + \frac{\partial}{\partial z}\left[\left(\varepsilon_S K_S + \varepsilon_P K_P + \varepsilon_L K_L\right)\frac{\partial T}{\partial z}\right] \end{aligned} \tag{10}$$

Using the substitution [2]:

$$\boldsymbol{v}_L = -\frac{\varepsilon_P \boldsymbol{v}_P}{\varepsilon_L} \tag{20}$$

We arrive at:

$$\begin{aligned} &\underbrace{\left(\frac{\partial \varepsilon_s}{\partial t}\rho_s C_{Ps} + \frac{\partial \varepsilon_p}{\partial t}\rho_P C_{PP} + \frac{\partial \varepsilon_L}{\partial t}\rho_L C_{PL}\right)}_{A} T + \underbrace{\left(\rho_s \varepsilon_s C_{Ps} + \rho_P \varepsilon_p C_{PP} + \rho_L \varepsilon_L C_{PL}\right)}_{B}\frac{\partial T}{\partial t} \\ &\quad + \frac{\partial}{\partial x}\left[\underbrace{\left(\rho_P C_{PP} - \rho_L C_{PL}\right)}_{C} \varepsilon_P \boldsymbol{v}_P T\right] \\ &\quad + \underbrace{\left(\frac{\partial \varepsilon_S}{\partial x}K_S + \frac{\partial \varepsilon_P}{\partial x}K_P + \frac{\partial \varepsilon_L}{\partial x}K_L\right)}_{D_x}\frac{\partial T}{\partial x} + \underbrace{\left(\varepsilon_S K_S + \varepsilon_P K_P + \varepsilon_L K_L\right)}_{E}\frac{\partial^2 T}{\partial x^2} \\ &\quad + \underbrace{\left(\frac{\partial \varepsilon_S}{\partial y}K_S + \frac{\partial \varepsilon_P}{\partial y}K_P + \frac{\partial \varepsilon_L}{\partial y}K_L\right)}_{D_y}\frac{\partial T}{\partial y} + \underbrace{\left(\varepsilon_S K_S + \varepsilon_P K_P + \varepsilon_L K_L\right)}_{E}\frac{\partial^2 T}{\partial y^2} \end{aligned} \tag{21}$$

$$+\underbrace{\left(\frac{\partial \varepsilon_S}{\partial z}K_S+\frac{\partial \varepsilon_P}{\partial z}K_P+\frac{\partial \varepsilon_L}{\partial z}K_L\right)}_{D_z}\frac{\partial T}{\partial z}+\underbrace{(\varepsilon_S K_S+\varepsilon_P K_P+\varepsilon_L K_L)}_{E}\frac{\partial^2 T}{\partial z^2}$$

We use the letter substitutions to simplify writing and arrive at:

$$AT+B\frac{\partial T}{\partial t}+\underbrace{C\frac{\partial \varepsilon_P}{\partial x}v_P}_{F_1}T+\underbrace{C\frac{\partial v_P}{\partial x}\varepsilon_P}_{F_2}T+\frac{\partial T}{\partial x}\underbrace{\varepsilon_P v_P C}_{F_3}$$
$$=D_x\frac{\partial T}{\partial x}+D_y\frac{\partial T}{\partial y}+D_z\frac{\partial T}{\partial z}+E\left(\frac{\partial^2 T}{\partial x^2}+\frac{\partial^2 T}{\partial y^2}+\frac{\partial^2 T}{\partial z^2}\right) \quad \textbf{(22)}$$

$$(=)$$

$$T(A+F_1+F_2)+B\frac{\partial T}{\partial t}+F_3\frac{\partial T}{\partial x}$$
$$=D_x\frac{\partial T}{\partial x}+D_y\frac{\partial T}{\partial y}+D_z\frac{\partial T}{\partial z}+E\left(\frac{\partial^2 T}{\partial x^2}+\frac{\partial^2 T}{\partial y^2}+\frac{\partial^2 T}{\partial z^2}\right) \quad \textbf{(23)}$$

Unfolding $\frac{\partial T}{\partial t}$ using the FDM:

$$\frac{T^{t+1}-T^t}{\Delta t}B=\cdots \quad \textbf{(24)}$$

And evaluating all other terms at time $t+1$:

$$T^t=T^{t+1}\underbrace{\left[1+\frac{\Delta t}{B}(A+F_1+F_2)\right]}_{G}+\underbrace{\left(\frac{F_3\Delta t}{B}-\frac{D_x\Delta t}{B}\right)}_{H_x}\frac{\partial T}{\partial x}$$
$$\underbrace{-\frac{D_y\Delta t}{B}}_{H_y}\frac{\partial T}{\partial y}\underbrace{-\frac{D_z\Delta t}{B}}_{H_z}\frac{\partial T}{\partial z}-\underbrace{\frac{E\Delta t}{B}}_{I}\left(\frac{\partial^2 T}{\partial x^2}+\frac{\partial^2 T}{\partial y^2}+\frac{\partial^2 T}{\partial z^2}\right) \quad \textbf{(25)}$$

$$(=)$$

$$T^t=T^{t+1}G+H_x\frac{T_{j+1}^{t+1}-T_{j-1}^{t+1}}{2\Delta x}+H_y\frac{T_{k+1}^{t+1}-T_{k-1}^{t+1}}{2\Delta y}+H_z\frac{T_{l+1}^{t+1}-T_{l-1}^{t+1}}{2\Delta z}$$
$$-I\left(\frac{T_{j+1}^{t+1}-2T_j^{t+1}+T_{j-1}^{t+1}}{\Delta x^2}+\frac{T_{k+1}^{t+1}-2T_k^{t+1}+T_{k-1}^{t+1}}{\Delta y^2}+\frac{T_{l+1}^{t+1}-2T_l^{t+1}+T_{l-1}^{t+1}}{\Delta z^2}\right) \quad \textbf{(26)}$$

$$(=)$$

$$T^t=T^{t+1}\underbrace{\left[G-I\left(\frac{1}{\Delta x^2}+\frac{1}{\Delta y^2}+\frac{1}{\Delta z^2}\right)\right]}_{J}$$
$$+T_{j+1}^{t+1}\underbrace{\left(\frac{H_x}{2\Delta x}-\frac{I}{\Delta x^2}\right)}_{J_+}+T_{j-1}^{t+1}\underbrace{\left(-\frac{H_x}{2\Delta x}-\frac{I}{\Delta x^2}\right)}_{J_-}+T_{k+1}^{t+1}\underbrace{\left(\frac{H_y}{2\Delta y}-\frac{I}{\Delta y^2}\right)}_{K_+} \quad \textbf{(27)}$$

$$+T_{k-1}^{t+1}\underbrace{\left(-\frac{H_y}{2\Delta y}-\frac{I}{\Delta y^2}\right)}_{K_-}+T_{l+1}^{t+1}\underbrace{\left(\frac{H_z}{2\Delta z}-\frac{I}{\Delta z^2}\right)}_{L_+}+T_{l-1}^{t+1}\underbrace{\left(-\frac{H_z}{2\Delta z}-\frac{I}{\Delta z^2}\right)}_{L_-}$$

We now arrive at an equation of the form:

$$T^t = T^{t+1}J + T_{j+1}^{t+1}J_+ + T_{j-1}^{t+1}J_- + T_{k+1}^{t+1}K_+ + T_{k-1}^{t+1}K_- + T_{l+1}^{t+1}L_+ + T_{l-1}^{t+1}L_- \quad \textbf{(11)}$$

Of which:

$$J = 1 + \frac{\Delta t(\rho_P C_{PP} - \rho_L C_{PL})}{\langle \rho C_p \rangle}\left(\frac{\partial \varepsilon_P}{\partial x}v_P + \frac{\partial v_P}{\partial x}\varepsilon_P\right) + \frac{\partial \langle \rho C_p \rangle}{\partial t}\frac{\Delta t}{\langle \rho C_p \rangle} + \frac{2\Delta t\langle K \rangle}{\langle \rho C_p \rangle}\left(\frac{1}{\Delta x^2}+\frac{1}{\Delta y^2}+\frac{1}{\Delta z^2}\right) \quad \textbf{(28)}$$

$$J_+ = \frac{\Delta t \varepsilon_P \boldsymbol{v}_P(\rho_P C_{PP} - \rho_L C_{PL})}{2\Delta x\langle \rho C_p \rangle} - \frac{\partial \langle K \rangle}{\partial x}\frac{\Delta t}{2\Delta x\langle \rho C_p \rangle} - \frac{\Delta t\langle K \rangle}{\langle \rho C_p \rangle \Delta x^2} \quad \textbf{(29)}$$

$$J_- = -\frac{\Delta t \varepsilon_P \boldsymbol{v}_P(\rho_P C_{PP} - \rho_L C_{PL})}{2\Delta x\langle \rho C_p \rangle} + \frac{\partial \langle K \rangle}{\partial x}\frac{\Delta t}{2\Delta x\langle \rho C_p \rangle} - \frac{\Delta t\langle K \rangle}{\langle \rho C_p \rangle \Delta x^2} \quad \textbf{(30)}$$

$$K_+ = -\frac{\partial \langle K \rangle}{\partial y}\frac{\Delta t}{2\Delta y\langle \rho C_p \rangle} - \frac{\Delta t\langle K \rangle}{\langle \rho C_p \rangle \Delta y^2} \quad \textbf{(31)}$$

$$K_- = \frac{\partial \langle K \rangle}{\partial y}\frac{\Delta t}{2\Delta y\langle \rho C_p \rangle} - \frac{\Delta t\langle K \rangle}{\langle \rho C_p \rangle \Delta y^2} \quad \textbf{(32)}$$

$$L_+ = -\frac{\partial \langle K \rangle}{\partial z}\frac{\Delta t}{2\Delta z\langle \rho C_p \rangle} - \frac{\Delta t\langle K \rangle}{\langle \rho C_p \rangle \Delta z^2} \quad \textbf{(33)}$$

$$L_- = \frac{\partial \langle K \rangle}{\partial z}\frac{\Delta t}{2\Delta z\langle \rho C_p \rangle} - \frac{\Delta t\langle K \rangle}{\langle \rho C_p \rangle \Delta z^2} \quad \textbf{(34)}$$

Appendix 2: Main Function Code

```
    function       [       VP,FP,FL,FS,T2imp]       =       Simulate(
x,re,ri,y,Y,z,Z,t,passo,FRi,Ti,Te,rpmi,sut,TOL)
    % Test Function
    %                                          [VP,FP,FL,FS,T2imp]=Simu-
late(20,0.38,0.3,10,0.04,10,0.04,90,0.5,0.1,1120,300,1000,5,120*10e-
6);
    %
    %
    % Entry:
    % "x"                       % Matrix dimension in x
    % "re"                      % Exterior radius in m
    % "ri"                      % Interior radius in m
    % "y"                       % Matrix dimension in y
    % "Y"                       % Dimension Y in m
    % "z"                       % Matrix dimension in z
    % "Z"                       % Dimension Z in m
    % "t"                       % Maximum solidification time in s
    % "passo"                   % Time step
    % "FRi"                     % Initial particle fraction
    % "Ti"                      % Initial melt temperature in K
    % "rpm"                     % Rotations per minute
    % "sut"                     % Spin-Up Time
    %
    %================================================================%
    % Performance timer
    tic


    %Constants:
```

```
     rpm=2*pi*rpmi/60;          % Rotations per minute -> S.I. (rad.s^-
1)

     ROl=2390;                   % Liquid phase density kg/m^3

     ROs=2550;                   % Solid phase density kg/m^3

     ROp=3200;                   % Particle phase density kg/m^3

     NEUl=1.26*(10^-3);          % Liquid phase viscocity kg/m.s

     Cpl=1079.5;                 % Liquid phase specific heat J/kg.K

     Cps=1176.5;                 % Solid phase specific heat J/kg.K

     Cpp=840;                   % Particle phase specific heat J/kg.K

     Kl=95;                     % Thermal condutivity for liquid phase
W/m.K

     Ks=210;                     % Thermal condutivity for solid phase
W/m.K

     Kp=16;                       % Thermal condutivity for particle
phase W/m.K

     Deltah=3.97*10^5;           % Latent heat of fusion J/kg

     Tm=933.6;                   % Melting point K


     %=================================================================%

     % Particle size

     PS=10*10^-6;                % Particle diameter in m


     %=================================================================%

     % Experimental conditions (time and space)


     tempo=t/passo;              % Number of iterations


     dx=(re-ri)/x;

     dy=Y/y;

     dz=Z/z;


     x=x-1;
```

```
r=re:-dx:ri;
x=x+1;


%=============================================================%
% Initial conditions


E=zeros(x,y,z);             % Enthalpy field
FP=E;                       % Particle phase field
FL=E;                       % Liquid phase field
FS=E;                       % Solid phase field
VP=E;                       % Particle velocity field
VPR=E;                      % Real particle velocity field


Entrada=E;
Saida=E;


T2imp=zeros(x*y*z,1);
T2imp(:)=Ti;
for l=1:z
    for k=1:y
        for j=1:x
            if l==1 || l==z || k==1 || k==y || j==x || j==1
                S=j+(k-1)*x+(l-1)*x*y;
                T2imp(S)=Te;
                FS(j,k,l)=1-FRi;
            end
        end
    end
end


E(:)=Deltah+(FRi*Cpp+Cpl*(1-FRi))*(Ti-Tm);     % Initial Enthalpy
```

```
    FP(:,:,:)=FRi;                                  % Initial particle
fraction
    FL(:,:,:)=1-FRi-FS(:,:,:);                      % Initial liquid
phase


    %=================================================================%
    % Testing
    % If the time step is too short, an error will occur

    Vmax=(rpm^2*re*(ROp-ROl)*PS^2*(1-FRi)^4.65*FRi)/(18*NEUl*(FRi));
    a=dx/Vmax;
    if passo<a
    else
        disp('Time step bigger than');
        disp(a);
        return
    end

    Timp=sparse(x*y*z,x*y*z);

    FS1=FS;

    %=================================================================%
    % Simulation
    for i=2:tempo
        % Iteration counter
        count=0;
        % Time step starts
        FPO=FP;
        T2impO=T2imp;
        EO=E;
```

```
        % Meaningless entry condition
        E1=2*E;

    while (abs(sum(sum(sum(E-E1)))))>TOL

        if (i-1)*passo<sut
            rpm=0;
        else
            rpm=2*pi*rpmi/60;
        end

        count=count+1;
        fprintf('Count is %1i\n',count);

        % Original values saved from previous time step
        FP=FPO;
        FL=1-(FP+FS);
        T2imp=T2impO;
        E1=E;
        E=EO;

        % Velocity calculation
        for l=1:z
            for k=1:y
                for j=1:x

VP(j,k,l)=(FP(j,k,l)/((FP(j,k,l)+FL(j,k,l))^2*(1-FL(j,k,l))))*(((1-
FP(j,k,l))^4.65)*rpm^2*r(j)*(ROp-ROl)*PS^2)/(18*NEUl);
                end
            end
        end
```

```
% Particle movement and liquid update
for l=1:z
    for k=1:y
        for j=1:x
            if j==1
            elseif FP(j,k,l)==0
                Saida(j,k,l)=0;
                Entrada(j-1,k,l)=0;
            elseif FP(j-1,k,l)+FS(j-1,k,l)>0.52
                Saida(j,k,l)=0;
                Entrada(j-1,k,l)=0;
            elseif FS(j,k,l)>=0.52
                Saida(j,k,l)=0;
                Entrada(j-1,k,l)=0;
            else
                Saida(j,k,l)=(VP(j,k,l)*passo)/dx*FP(j,k,l);
                Entrada(j-1,k,l)=Saida(j,k,l);

                if Entrada(j-1,k,l)+FP(j-1,k,l)>=0.52
                    Saida(j,k,l)=0.52-FP(j-1,k,l);
                    Entrada(j-1,k,l)=Saida(j,k,l);
                end

                if FP(j,k,l)-Saida(j,k,l)<=0
                    Saida(j,k,l)=FP(j,k,l);
                    Entrada(j-1,k,l)=Saida(j-1,k,l);
                end
            end
        end
    end
```

```
        end


        FP1=FP;

        FL1=FL;


        FP(:)=FP(:)+Entrada(:)-Saida(:);

        FL(:)=1-(FS(:)+FP(:));


        for j=1:x-1

            if j==1

                VPR(j+1,:,:)=(FP(j,:,:)-
FP1(j,:,:)).*dx./(passo.*FP1(j+1,:,:));

            else

                VPR(j+1,k,l)=((FP(j,k,l)-
FP1(j,k,l))*dx/passo+FP1(j,k,l)*VPR(j,k,l))/FP1(j+1,k,l);

            end

        end


        % Temperature field

        for l=1:z

            for k=1:y

                for j=1:x

                    if l==1 || l==z || k==1 || k==y || j==x || j==1

                        S=j+(k-1)*x+(l-1)*x*y;

                        Timp(S,S)=1;

                    else


ROCP=(FP(j,k,l)*ROp*Cpp+FS(j,k,l)*ROs*Cps+FL(j,k,l)*ROl*Cpl);

                        dROCPdt=((FP(j,k,l)-
FP1(j,k,l))*ROp*Cpp+(FS(j,k,l)-FS1(j,k,l))*ROs*Cps+(FL(j,k,l)-
FL1(j,k,l))*ROl*Cpl)/passo;

                        dEPdx=(FP(j+1,k,l)-FP(j-1,k,l))/(2*dx);

                        dVPdx=(VPR(j+1,k,l)-VPR(j-1,k,l))/(2*dx);
```

```
                        K=(Kp*FP(j,k,l)+Ks*FS(j,k,l)+Kl*FL(j,k,l));
                        dKdx=(Kp*(FP(j+1,k,l)-FP(j-
1,k,l))+Ks*(FS(j+1,k,l)-FS(j-1,k,l))+Kl*(FL(j+1,k,l)-FL(j-
1,k,l)))/(2*dx);
                        dKdy=(Kp*(FP(j,k+1,l)-FP(j,k-
1,l))+Ks*(FS(j,k+1,l)-FS(j,k-1,l))+Kl*(FL(j,k+1,l)-FL(j,k-
1,l)))/(2*dy);
                        dKdz=(Kp*(FP(j,k,l+1)-FP(j,k,l-
1))+Ks*(FS(j,k,l+1)-FS(j,k,l-1))+Kl*(FL(j,k,l+1)-FL(j,k,l-
1)))/(2*dz);


                        dROCP=(ROp*Cpp-ROl*Cpl);



A=1+passo*dROCP*(dEPdx*VPR(j,k,l)+dVPdx*FP(j,k,l))/ROCP+passo*dROCPd
t/ROCP+2*passo*K*(1/dx^2+1/dy^2+1/dz^2)/ROCP;


B=passo*FP(j,k,l)*VP(j,k,l)*dROCP/(2*dx*ROCP)-
passo*dKdx/(2*dx*ROCP)-passo*K/(ROCP*dx^2);
                        C=-
passo*FP(j,k,l)*VP(j,k,l)*dROCP/(2*dx*ROCP)+passo*dKdx/(2*dx*ROCP)-
passo*K/(ROCP*dx^2);
                        D=-passo*dKdy/(2*dy*ROCP)-
passo*K/(ROCP*dy^2);
                        E=passo*dKdy/(2*dy*ROCP)-
passo*K/(ROCP*dy^2);
                        F=-passo*dKdz/(2*dz*ROCP)-
passo*K/(ROCP*dz^2);
                        G=passo*dKdz/(2*dz*ROCP)-
passo*K/(ROCP*dz^2);


                        S=j+(k-1)*x+(l-1)*x*y;
                        Timp(S,S)=A;
                        Timp(S,S+1)=B;
                        Timp(S,S-1)=C;
```

```
                    Timp(S,S+x)=D;
                    Timp(S,S-x)=E;
                    Timp(S,S+x*y)=F;
                    Timp(S,S-x*y)=G;
                end


            end
        end
    end

T1=align(x,y,z,T2imp);
T2imp=Timp\T2imp;

% Solidification
T2=alinha(x,y,z,T2imp);
Dif=(T1-T2).*(FP*Cpp+FS*Cps+FL*Cpl);
E=E-Dif;

FS1=FS;
for l=2:z-1
    for k=2:y-1
        for j=2:x-1
            if FL(j,k,l)==0
            else
                if T2(j,k,l)>=Tm
                else
                    if E(j,k,l)<=0
                        FS(j,k,l)=1-FP(j,k,l);
                        FL(j,k,l)=0;
                    else
                        T2imp(j+(k-1)*x+(l-1)*x*y)=Tm;
```

```
                                FS(j,k,l)=FL(j,k,l)*(Deltah-
E(j,k,l))/Deltah;
                                FL(j,k,l)=1-(FS(j,k,l)+FP(j,k,l));
                            end
                        end
                    end
                end
            end
        end
    end
        % "Video"
    %     hold on
    %     cla
    %     surf(T2(:,:,z/2))
    %     drawnow
    %     grid on
        % Stoping condition
        if max(FL)==0
            save('Last Save');
            break
        end
        fprintf('Iteration is %1i\n',i)
     end
     toc
    end
```

Appendix 3: Align function code

```
    function [ T ] = align( x,y,z,T2imp )
    % Function to return a "3D" matrix of the original 2D matrix
```

```
T=zeros(x,y,z);
for l=1:z
    for k=1:y
        S=1+(k-1)*x+(l-1)*x*y;
        T(:,k,l)=T2imp(S:S+x-1);
    end
end



end
```

Appendix 4: Particle distribution function code

For the particle distribution, the following sections differ from the main code:

```
    % Particle Size
    [size,PA,PB,PC,PD]=Granolometria(FRi,x,y,z);
    PS=max(size);
```

```
    % Velocity calculation
        for l=1:z
            for k=1:y
                for j=1:x

VPA(j,k,l)=(FP(j,k,l)/((FP(j,k,l)+FL(j,k,l))^2*(1-FL(j,k,l))))*(((1-
FP(j,k,l))^4.65)*rpm^2*r(j)*(ROp-ROl)*size(1)^2)/(18*NEUl);

VPB(j,k,l)=(FP(j,k,l)/((FP(j,k,l)+FL(j,k,l))^2*(1-FL(j,k,l))))*(((1-
FP(j,k,l))^4.65)*rpm^2*r(j)*(ROp-ROl)*size(2)^2)/(18*NEUl);

VPC(j,k,l)=(FP(j,k,l)/((FP(j,k,l)+FL(j,k,l))^2*(1-FL(j,k,l))))*(((1-
FP(j,k,l))^4.65)*rpm^2*r(j)*(ROp-ROl)*size(3)^2)/(18*NEUl);

VPD(j,k,l)=(FP(j,k,l)/((FP(j,k,l)+FL(j,k,l))^2*(1-FL(j,k,l))))*(((1-
FP(j,k,l))^4.65)*rpm^2*r(j)*(ROp-ROl)*size(4)^2)/(18*NEUl);
                end
            end
        end
```

```
    % Calcular FP e FL


        FP1=FP;
        FL1=FL;
```

```
        [PA,PB,PC,PD]=Movi-
mento(PA,PB,PC,PD,VPA,VPB,VPC,VPD,x,y,z,FS,passo,dx);


        FP=PA+PB+PC+PD;
        FL=1-(FS+FP);


        for j=1:x-1
            if j==1
                VPR(j+1,:,:)=(FP(j,:,:)-
FP1(j,:,:)).*dx./(passo.*FP1(j+1,:,:));
            else
                VPR(j+1,k,l)=((FP(j,k,l)-
FP1(j,k,l))*dx/passo+FP1(j,k,l)*VPR(j,k,l))/FP1(j+1,k,l);
            end
        end
```

Appendix 5: Granulometry function code

```
function [size,PA,PB,PC,PD] = Granolometry(Fri,x,y,z)
% Function to generate the starting particle fraction populations
A=28;
B=38;
C=20;
D=14;

dA=75e-6;
dB=112.5e-6;
dC=137.5e-6;
dD=175e-6;

size=[dA dB dC dD];
A=A*Fri/100;
B=B*Fri/100;
C=C*Fri/100;
D=D*Fri/100;

PA=zeros(x,y,z);
PB=PA;
PC=PA;
PD=PA;
PA(:)=A;
PB(:)=B;
PC(:)=C;
PD(:)=D;
end
```

Appendix 6: Movement function code

```
    function [ PA,PB,PC,PD ] = Move-
ment(PA,PB,PC,PD,VPA,VPB,VPC,VPD,x,y,z,FS,passo,dx)
    % Function to calculate particle movement
    FP=PA+PB+PC+PD;


    % Exit vectors
    SA=zeros(x,y,z);
    SB=SA;
    SC=SA;
    SD=SA;
    % Entry vectors
    EA=SA;
    EB=SA;
    EC=SA;
    ED=SA;


    for l=1:z
        for k=1:y
            for j=1:x
                if j==1
                elseif FP(j,k,l)<=1e-5
                    SA(j,k,l)=0;
                    SB(j,k,l)=0;
                    SC(j,k,l)=0;
                    SD(j,k,l)=0;
                    EA(j-1,k,l)=0;
                    EB(j-1,k,l)=0;
                    EC(j-1,k,l)=0;
                    ED(j-1,k,l)=0;
```

```
            elseif FP(j-1,k,l)>=0.52
                SA(j,k,l)=0;
                SB(j,k,l)=0;
                SC(j,k,l)=0;
                SD(j,k,l)=0;
                EA(j-1,k,l)=0;
                EB(j-1,k,l)=0;
                EC(j-1,k,l)=0;
                ED(j-1,k,l)=0;
            elseif FS(j-1,k,l)>0
                SA(j,k,l)=0;
                SB(j,k,l)=0;
                SC(j,k,l)=0;
                SD(j,k,l)=0;
                EA(j-1,k,l)=0;
                EB(j-1,k,l)=0;
                EC(j-1,k,l)=0;
                ED(j-1,k,l)=0;
            else
                SA(j,k,l)=PA(j,k,l)*(VPA(j,k,l)*passo)/dx;
                SB(j,k,l)=PB(j,k,l)*(VPB(j,k,l)*passo)/dx;
                SC(j,k,l)=PC(j,k,l)*(VPC(j,k,l)*passo)/dx;
                SD(j,k,l)=PD(j,k,l)*(VPD(j,k,l)*passo)/dx;
                EA(j-1,k,l)=SA(j,k,l);
                EB(j-1,k,l)=SB(j,k,l);
                EC(j-1,k,l)=SC(j,k,l);
                ED(j-1,k,l)=SD(j,k,l);
                M=SA(j,k,l)+SB(j,k,l)+SC(j,k,l)+SD(j,k,l);
                if M+FP(j-1,k,l)>=0.52
                    Espaco=0.52-FP(j-1,k,l);
                    SA(j,k,l)=SA(j,k,l)*Espaco/M;
                    SB(j,k,l)=SB(j,k,l)*Espaco/M;
                    SC(j,k,l)=SC(j,k,l)*Espaco/M;
```

```
                    SD(j,k,l)=SD(j,k,l)*Espaco/M;
                    EA(j-1,k,l)=SA(j,k,l);
                    EB(j-1,k,l)=SB(j,k,l);
                    EC(j-1,k,l)=SC(j,k,l);
                    ED(j-1,k,l)=SD(j,k,l);
                end
                if PA(j,k,l)-SA(j,k,l)<0
                    SA(j,k,l)=PA(j,k,l);
                    EA(j-1,k,l)=PA(j,k,l);
                end
                if PB(j,k,l)-SB(j,k,l)<0
                    SB(j,k,l)=PB(j,k,l);
                    EB(j-1,k,l)=PB(j,k,l);
                end
                if PC(j,k,l)-SC(j,k,l)<0
                    SC(j,k,l)=PC(j,k,l);
                    EC(j-1,k,l)=PC(j,k,l);
                end
                if PD(j,k,l)-SD(j,k,l)<0
                    SD(j,k,l)=PD(j,k,l);
                    ED(j-1,k,l)=PD(j,k,l);
                end
            end
        end
    end
    PA=PA-SA+EA;
    PB=PB-SB+EB;
    PC=PC-SC+EC;
    PD=PD-SD+ED;
end
```