Modelling of Resonant Acoustic Mixing of Dry Spherical Particles

R.A. van der Walt

Modelling of Resonant Acoustic Mixing of Dry Spherical Particles

by

R.A. van der Walt

A thesis submitted in partial fulfilment
of the requirements for the degree

Masters in Mechanical Engineering

in the

Mechanical and Aeronautical Engineering
Faculty of Engineering, the Built Environment and Information Technology

University of Pretoria
Pretoria

February, 2022

ABSTRACT

MODELLING OF RESONANT ACOUSTIC MIXING OF DRY SPHERICAL PARTICLES

by

R.A. van der Walt

Supervisor(s):	Prof. D.N. Wilke
	Prof. W. Focke
Department:	Mechanical and Aeronautical Engineering
University:	University of Pretoria

This study documents the findings and methods of mapping the resonant acoustic mixing parameters of spherical particles in a cylindrical tube or container. The document systematically explores the approach followed in conducting this study by investigating the validity of certain mixing measurements, developing new measurement techniques, parameter reduction and parameter sensitivity studies to design an experimental approach that is finally used to validate a numerical discrete element model.

The study focused on separating key variables that influence the mixing outcome of each use case as well as establishing a better understanding of the mapped domain. All assumptions focused the attention on the key elements of the study to simplify each parameter to the most generic case. The study thus looks at the particle mixing behaviour of similar particles mixed in a cylindrical domain.

The study is supported by numerically analysing results that were captured using the Lagrangian discrete element method solver in the Star-CCM+ package. The simulations were launched embarrassingly parallel on a High-Performance Computer to map the domain in a reasonable time-frame. An experimental setup was constructed to aid the findings in analysing validating the numerical findings. This experimental setup has variable speed and amplitude settings which allows for a mixing intensity of up to $105Gs$ $[1030m/s^2]$. The setup is currently installed and in working order at the University of Pretoria.

All numerical parameters were set up such that they are in a way that it is physically implementable and realistic when varying the particle properties from soft to hard particles for various particle sizes, shaken using different container sizes. The result yielded some insight into which parameters played a larger role in the mixing efficiency; the key parameters included the container to particle volume ratio, mixing intensity and the container dimensions.

Each of these parameters had its role to play in determining the mixing state as is thoroughly explored in this study.

Contents

Introduction **1**
Background . . . 1
Scope . . . 2

1 Literature Review **3**
1.1 Resonant Acoustic Mixers . . . 3
1.2 Resonant condition . . . 4
1.3 Molecular Physics . . . 7
1.3.1 Discrete Element Dynamics . . . 7
1.3.2 Contact Models . . . 9
1.4 Mixing Quantification . . . 15
1.4.1 Sampling methods . . . 16
1.4.2 Displacement dependant methods . . . 20
1.4.3 Cell dependant methods . . . 23
1.4.4 Neighbourhood Approach . . . 25
1.4.5 Summary . . . 27

2 Mixing Index Review **29**
2.1 Sampling methods . . . 31
2.1.1 Nearest neighbours . . . 32
2.1.2 Coordination Number . . . 33
2.1.3 Particle Scale Index . . . 34
2.2 Displacement dependant methods . . . 37
2.2.1 Average Height Method . . . 37
2.2.2 Neighbour distance and Siiria . . . 38
2.3 Cell dependant methods . . . 38
2.3.1 Lacey . . . 39
2.3.2 Mixing Entropy . . . 41
2.4 Neighbourhood methods . . . 42
2.4.1 Chain Length . . . 43
2.4.2 Growth Point . . . 45
2.5 Domain rotation . . . 49
2.6 Mixing indexes to be used . . . 50

3 Numerical Setup and supporting material **51**
3.1 Simulation Initialization . . . 52
3.1.1 Licensing . . . 52

3.1.2 Create internal volume . . . 53
3.1.3 Create Physics Environment . . . 53
3.1.4 Hertz-Mindlin contact model setup . . . 54
3.1.5 Field functions . . . 55
3.1.6 Injector grid . . . 56
3.2 Setup HPC workflow . . . 57
3.2.1 Batch environment . . . 59
3.2.2 Java Macros and Automation . . . 61
3.3 Batch processing of results . . . 62

4 Dimensional Analysis and Numerical quantification 64
4.1 Dimensional study . . . 65
4.1.1 Buckingham Π theorem variable grouping . . . 66
4.2 Dimensionless Π-groups validity check . . . 69
4.2.1 Blending time and frequency . . . 70
4.2.2 Primarily cup height and diameter variation subject to constant Π-group constraints 72
4.2.3 Amplitude variance . . . 72
4.2.4 Varying the particle count and diameter independently . . . 73
4.2.5 Particle mechanical properties - density, elasticity and Poisson variance . . . 74
4.2.6 Normal and Tangential restitution coefficients . . . 74
4.2.7 Friction coefficients . . . 75
4.2.8 Summary . . . 76
4.3 Π_k variable sensitivity study . . . 76
4.3.1 Time, Frequency, and Mixing intensity (Π_0 & Π_2) . . . 77
4.3.2 Volumetric variables (Π_1 & Π_3) . . . 79
4.3.3 Particle Material properties (Π_4 & Π_5) . . . 79
4.3.4 Particle and Wall restitution groups (Π_6 & Π_7) . . . 80
4.4 Reflection and future . . . 81

5 Experimental analysis 82
5.1 Experimental setup . . . 82
5.1.1 Particle Shaker . . . 82
5.1.2 Interpretation and preparation of results . . . 83
5.2 Experimental results . . . 84
5.2.1 Particle Fill level - Π_3 experimental results . . . 86
5.2.2 Cup Volume or Π_1 experimental results . . . 87
5.2.3 Acceleration or Π_2 experimental results . . . 87
5.3 Summary . . . 88

Conclusion 89

Bibliography d

A Variable Independence data A- 1
A.1 Variable independence adjustment . . . A- 1
A.2 Variable independence data . . . A- 5
A.3 Sensitivity study adjustment . . . A- 9
A.4 Sensitivity study data . . . A- 11

B Experimental Data **B- 1**
B.1 Acceleration experiment . . . . . B- 4
B.1.1 Acceleration - $\Pi_2 = 0.3$ . . . . . B- 4
B.1.2 Acceleration - $\Pi_2 = 1.4$ . . . . . B- 6
B.1.3 Acceleration - $\Pi_2 = 1.6$ . . . . . B- 7
B.1.4 Acceleration - $\Pi_2 = 2.2$ . . . . . B- 9
B.2 Particle Fill level Sweep . . . . . B- 10
B.2.1 Particle Fill level - $\Pi_3 = 0.2$ . . . . . B- 10
B.2.2 Particle Fill level - $\Pi_3 = 0.3$ . . . . . B- 11
B.2.3 Particle Fill level - $\Pi_3 = 0.4$ . . . . . B- 12
B.3 Volume Sweep . . . . . B- 13
B.3.1 Volume ratio - $\Pi_1 = 0.5$ . . . . . B- 13
B.3.2 Volume ratio - $\Pi_1 = 1$ . . . . . B- 14
B.3.3 Volume ratio - $\Pi_1 = 2$ . . . . . B- 15

C Supporting Code **C- 1**

List of Figures

1.1 Resonant Acoustic Mixer types . . . 3
1.2 RAM - γ mapping . . . 5
1.3 RAM - β mapping . . . 6
1.4 RAM - G mapping . . . 6
1.5 Example of a DEM iteration on a particle (Wills, 2016) . . . 7
1.6 Contact forces diagram . . . 8
1.7 Contact diagram (Wassgren, 2008) . . . 10
1.8 Particle contact resolution example . . . 11
1.9 Hertz contact area . . . 12
1.10 Cundall and Strack linear spring contact model . . . 14
1.11 Mixed states . . . 15
1.12 Nearest neighbours - $n_C = 6$ . . . 17
1.13 Coordination number (ex. $n_C = 4$) . . . 18
1.14 Boundary case - Coordination number versus. Nearest neighbour . . . 18
1.15 PSI Mixing Variance . . . 20
1.16 Average Height Method . . . 21
1.17 Neighbour distance . . . 22
1.18 Domain cells . . . 24
1.19 Chain length method . . . 26
1.20 Growth point method . . . 27
1.21 Mixed and segregated example . . . 28

2.1 Hypothetical Mixing state . . . 29
2.2 Hypothetical mixed cases . . . 30
2.3 Teeth samples - $n_C = 6$ . . . 32
2.4 NN surface area comparison . . . 33
2.5 CN surface area comparison . . . 34
2.6 PSI surface area comparison . . . 35
2.7 PSI Segregated case . . . 36
2.8 PSI Mixed case . . . 36
2.9 PSI particle fraction relationships . . . 37
2.10 Lacey surface area comparison . . . 40
2.11 On and off boundary cells . . . 40
2.12 Lacey - cell saturation level . . . 41
2.13 ME surface area comparison . . . 42
2.14 Hypothetical mixed cases - heat mapping . . . 43
2.15 Hypothetical mixed cases - heat mapping (continued) . . . 43

2.16 CL surface area comparison . . . 44
2.17 GP surface area comparison . . . 46
2.18 Computational time for a segregated case vs. a mixed case . . . 47
2.19 Spherical packing . . . 47
2.20 Growth Point mixing evolution . . . 48
2.21 Domain rotations . . . 49
2.22 Domain Rotation performance . . . 49

3.1 DEM-flow field comparison . . . 52
3.2 Internal Volume . . . 53
3.3 Physics environment setup . . . 54
3.4 RAM movement profiles . . . 56
3.5 Presentation Grid . . . 57
3.6 Particle count time scaling . . . 58
3.7 Processing processes . . . 59
3.8 Numerical setup process flow . . . 62

4.1 Variable reduction vs. Non-dimensional analysis . . . 64
4.2 Dimensional Analysis - Independent Variables of a non-cohesive two-particle type RAM setup 66
4.3 Variable adjustment; Blending time independence . . . 71
4.4 Blending time . . . 72
4.5 Variable adjustment - Π_0 . . . 77
4.6 Variable adjustment - Π_2 . . . 78
4.7 Π sensitivity analysis - Π_0 and Π_2 . . . 78
4.8 Π_4 realistic parameters . . . 80

5.1 Experimental setup . . . 83
5.2 Experimental procedure . . . 85
5.3 Experimental Example . . . 86
5.4 Experimental Π_3 results . . . 86
5.5 Experimental Π_1 results . . . 87
5.6 Experimental Π_2 sweep . . . 88

List of Tables

1.1 Cation-anion radius ratio rules (Toofan, 1994) . . . 18
1.2 Mixed state Mixing Performance (Fig. 1.20) . . . 28

2.1 Nearest Neighbours performance . . . 32
2.2 Coordination Number performance . . . 34
2.3 Particle Scale Index performance . . . 35
2.4 Average Height performance . . . 38
2.5 Neighbour distance and Siiria performance . . . 38
2.6 Lacey performance . . . 39
2.7 Mixing Entropy performance . . . 42
2.8 Chain Length performance . . . 44
2.9 Growth Point performance . . . 45
2.10 Growth Point neighbourhoods . . . 46

3.1 Balancing scaling . . . 59
3.2 Ports and Host definition of Listing 3.1 . . . 60
3.3 Variable and File definition of Listing 3.2 . . . 60

4.1 Dependant and Independent variables in RAM shaking . . . 68
4.2 Variable independence non-dimensional group values . . . 69
4.3 Π_k benchmark values - Variable sweep . . . 70
4.4 Π_k benchmark values - Non-dimensional sweep . . . 77

5.1 Experimental properties parameters . . . 84

B.1 Experimental parameters . . . B- 1
B.1 Experimental parameters (continued) . . . B- 2
B.2 Experimental Π parameters . . . B- 2
B.2 Experimental Π parameters (continued) . . . B- 3
B.3 Experimental Mixing Information . . . B- 3
B.3 Experimental Mixing Information (continued) . . . B- 4

Listings

3.1 SSH Tunnels from License to server nodes 59
3.2 Star-CCM+ batch launch command 60
C.1 Mixing Index calculation code . . . . . C- 1

Introduction

Background

Mixing two dry materials efficiently and effectively has been of concern for many process engineers. Traditional mixing methods such as shear-mixing or tumbler mixing often causes segregation zones in the domain (Clark, 2009) or excessive inter-particle friction that may result in hotspots in the domain. Traditional mixing methods usually take longer to mix, and requires larger energy input. The localised heat zones may cause some of these materials to deteriorate and even detonate in the cases of mixing dry explosives. Thus low friction solid particle mixing methods forms a large part of many processes in the industry such as explosives, food and pharmaceutical productions.

More common mixers such as auger or ribbon mixers offer the ability to mix large amounts of materials at a fairly cheap rate (Directory, 2020). However Resonant Acoustic Mixing (RAM) only recently became a more popular approach in mixing due to its ability to mix materials that were usually deemed too difficult to mix. The idea of shaking a holder with two or more segregated substances is a long-established method of mixing something. Even at a young age, one can grasp the concept by helping someone in the kitchen or sorting through a jar with various particles inside. A simple shaking can manoeuvre the content to bring forth the desired object; sometimes it seems even more effective than stirring the mixture. For example, you have a bowl of mixed nuts and cranberries in front of you and you particularly only enjoy the nuts. To separate the two and make it easier to pick out, you shake the bowl in different ways and watch how the mixture carefully separates. Even in this case, one can understand that shaking particles can cause segregation regions in the domain. This is highlighted in the Brazil nut example in Granular Convection theory (Rosato, Strandburg, Prinz, et al., 1987). This study investigates the effect that different spherical particles, mixing intensities, volume size, material properties and friction components have on the effectiveness of mixing using a simulation-driven approach.

The mixing of particle matter is inherently non-linear, due to the particles constantly being in and out of contact with each other. The micro-mechanics driving the dynamic behaviour of the mixing may influence the mixing behaviour. However in this study, we will be looking at the problem holistically in trying to understand the overall mixing behaviour.

By mapping the process holistically one can then determine certain particle properties which influence the mixing outcome, then predict what the outcome of the mixing could be and this in return will help in understanding and conceptualising designs, using RAM.

To obtain the influential parameters one needs to start by identifying the parameters that may influence the particle movements in the domain by testing and analysing the results - numerically and experimentally.

Scope

The focus of this study is centred around conducting a sensitivity study of Resonant Acoustic mixing by looking into the non-dimensional groups that can be extracted from the input variables and investigating the determining drive force of this mixing method.

The first step is to construct a shaker that can withstand the immense forces and speeds that are required for this study. The design parameters are set at an amplitude of 20 mm and a frequency of 60 Hz. The shaker is designed and constructed so that it can handle this type of speed and load for a package weight of $200g$. The crank-piston type of setup is chosen as the most cost-effective solution. This also ensures that spare parts could easily be manufactured and sourced where needed without breaking the bank.

For the numerical studies to simulate the shaking of these particles using the discrete element method (DEM), the commercial software package Star-CCM+ from CDApapco (now Siemens) is applied. Discrete Element Method is a discontinuum approach computational method that uses a Lagrangian model to track and determine particle physics in a continuum field (Board, 1989). The start of Discrete Element Methods came around in the early 1970s with work done by Cundall and Strack, 1979 and a generalized theorem was developed by the mid-1980s. Although the youngest in the three major mechanical continuum mechanics fields (Computational Fluid Dynamics *CFD*, Finite Element Methods *FEM* and Discrete Element Methods *DEM*), the idea had been developed and basic particle physics codes were being implemented in most of the major commercial algorithms. The software offered a wide range of capabilities with a user-friendly front-end and a back-end that can plug into some of the world's most powerful computers. This is especially needed in parallel computing. One of the concerning topics that arose was the solving time of a simulation required which can range from 10 minutes to 12 hours+, depending on the input parameters. This study will require that quite a few simulations have to be run ($\pm$650 simulations) and an average total simulation time of 6700 hours (270 days) is required. To solve this, parallel processing steps need to be put into place to break down the solution time to digestible pieces. To solve this, the computing nodes at the Centre for High-Performance Computing needed to be used to run at least 48 simulations simultaneously.

As part of this study is to measure the performance between the numerical and experimental domains there was time allocated to investigate the core principles of mixing indices. By looking into the mixing indices it is possible to identify in what manner the index quantifies mixing and how they can be improved to identify segregation and anomalies in the particle patterns. In addition to the mixing indices the study looked into identifying Π groups and identifying sensitive variables in the study, by the method of the Buckingham Π (Buckingham, 1914) methodology. This methodology is focused on decreasing the dimensionality of a problem to a few core variables, ensuring experimental and numerical analysis time can be reduced significantly.

As the study stretched over two domains, numerical and experimental, choosing the right materials and sources for the investigation had proven as a key factor in the study this ensured accuracy and correlation in both domains. The numerical parameters were chosen in such a manner as to mimic real-world materials and realistic speeds as well as input energies.

As a proxy roughly two to three months was spent on researching discrete element methods, understanding the topic and identifying variables in the problem. Additional three months were devoted to unpacking and understanding mixing indices and the mechanics thereof. One month was then spent learning modelling software and the integration, automation and deployment on the HPC. Four-five months was spent on analysing and simulating the numerical landscape, identifying sensitive parameters and isolating Π groups. One month was used to build the experimental setup and an additional month was used to test using the shaker.

Chapter 1

Literature Review

1.1 Resonant Acoustic Mixers

Currently, in the industrial market, there are two main types of acoustic mixers being used - Resodyn's acoustic mixers and a wide range of dental triturators. Examples of both these devices are shown in Figure 1.1. The manner in which these devices operate is by actuating a payload or mixing vessel at high speed for a set time - this motion activates the contents flows within the vessel and mixes. This type of mixing is different from some other mixers where a physical instrument is inserted into the volume to agitate the mixture.

In a typical scenario, the mixing is controlled by two input variables; blending time and blending intensity. In some dental triturators the only input variable to be controlled is the blending time (*How RAM Mixes - Resodyn Acoustic Mixers*).

(a) Resodyn Acoustic Mixer (*LabRAM I - Resodyn Acoustic Mixers*)

(b) Dental Triturator (Ebert, 2014)

Figure 1.1: Resonant Acoustic Mixer types

By controlling, whichever of these variables, the user can find the optimum mixing criteria for their sub-

stances. These devices have promised great results when mixing some of the more viscous substances in a matter of seconds; this is specifically appealing to the dental industry when they require to mix small volumes of viscous putty type materials before they set.

One of the key differences between the Resodyn and the dental type mixers is that the actuation direction, the Resodyn mixers actuate in, is in the vertical direction whilst the dental triturators mix is in a horizontal direction. For this study, the key focus area will be on the vertical type mixers.

The blending time is the total time the mixture is being agitated by the mixing mechanism. The blending intensity is defined by kinetic motion exerted on the vessel; this can be defined by the amount of G-force applied to the vessel to mix. This is then translated into the vessel by adjusting the two variables controlling the applied force; agitation amplitude and agitation frequency. The agitation frequency is either specified in Hertz or revolutions/cycles per minute. The dental triturators typically function at frequencies around $3000cpm$ or $50Hz$ platt16, whilst the Resodyn machines operate at frequencies around $60Hz$ (*How RAM Mixes - Resodyn Acoustic Mixers*). At these frequencies, these devices are set apart from platform shakers which operate typically at frequencies between $0.6Hz$ and $6.6Hz$ (*Laboratory Platform Shaker — Labcompare.com*).

The aim of the study is to understand the influence of all the parameters on mixing. As in most study fields, the best way to start investigating is to look at the origin of the study and the foundational principles that may have an influence on the mixing outcome. In this case, it is the dynamics of particle mixing, mixing quantification, particle properties and volume constraints. When investigating particle physics and the models associated with them. The researcher will dive deeper into particle collisions, specifically looking at the Hertz-Mindlin model. The Hertz-Mindlin model is one of the most commonly used particle interaction models in commercial software.

Particle movement and mixing occur when two or more particles move past each other in the domain it is classified in. In the case of Resonant Acoustic Mixing (RAM), movement is produced by the forces which are generated by the acceleration provided by the motion to overcome the gravitational pull on each particle. This is in contrast to paddle or ribbon mixers where the forces are produced by a mechanism exerting forces onto the matter in the domain causing the flow of particles. Additional forces will also be induced by particle-particle or particle-wall collisions as the matter is excited in the domain.

1.2 Resonant condition

The Resodyn acoustic mixers work on the principle where it optimizes the amplitude variable to find what is called the *resonant condition* (*How RAM Mixes - Resodyn Acoustic Mixers*) at which the most effective mixing occurs. The Resodyn machines are constructed using a set of springs and dampers with additional weights to increase the mass of the entire system. The mixing platform is then agitated by a setup very similar to a conventional audio speaker by a method of a coil and a strong magnet. The coil is then turned on and off to create the oscillating effect. The resonant condition is achieved by changing the input frequency marginally as well as the input force to ensure the agitation is within the design boundaries of the machine. The system can be simplified to a single degree of freedom spring-mass-damper (ibid.) with a harmonic force subjected to the base. The following condition can be used to describe the system:

$$F_0 sin(\omega t) = M\frac{d^2x}{dt^2} + C\frac{dx}{dt} + kx \tag{1.1}$$

Where $x = Xsin(\omega t + \phi)$, F_0 is the input force and ω the frequency. M is the combined mass of the payload and the mechanisms. C is the combined damping force of the internal payload damping and the

mechanical dampers. The response amplitude can be written as (Rao and Nguyen, 2011):

$$X = \frac{F_0}{\sqrt{(k - \omega^2 M)^2 + \omega^2 C^2}} \tag{1.2}$$

The Resodyn mixers make use of the amplitude variable to optimize the input force F_0. By adjusting the input forces, the response X will adjust accordingly. As stated on Resodyns' website, the mixer mixes optimally at $60Hz$ and $100G$, where the maximum payload weight is $500g$.

The mass and internal damping of the Resodyn mixers are classified. However, it is possible to describe these variables in the form of ratios; $\gamma = m_{pl}/m_m$ and $\beta = c_{pl}/c_m$ - for example, if the mechanism weight is $m_m = 5kg$, the payload vs. system ratio is $\gamma = 0.1$. Similarly, if the mechanical damping is assumed to be $10N/m.s^{-1}$ the β indicates the ratio between the payload damping and the mechanism damping.

By adjusting γ, one can see the input force in Figure 1.2 increase exponentially as γ increases. Also, by increasing γ, the natural frequency decreases incrementally indicated by the peak of each plot. It is easy to see that by recognizing the added mass to the system in conjunction with the known damping on the system, the onboard computers in the Resodyn machine can adjust the input forces to ensure the mixing intensity of $100G$ is correctly applied to the system.

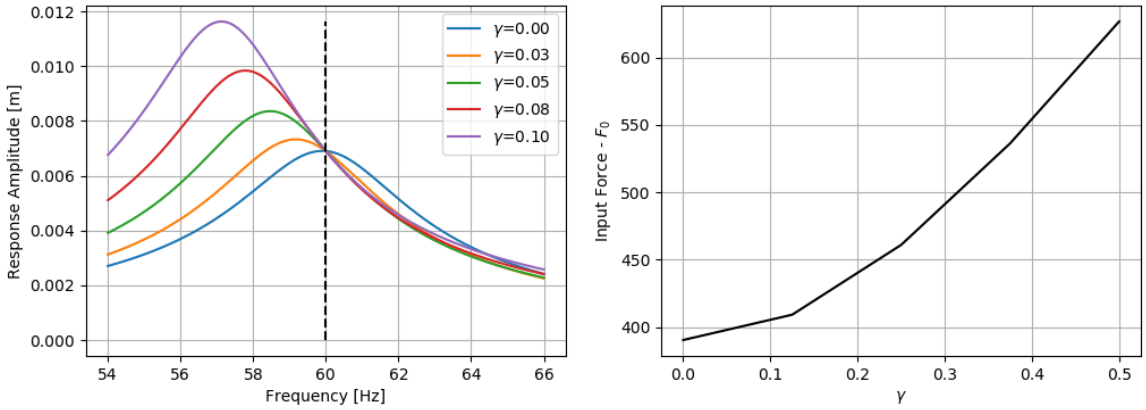


Figure 1.2: RAM - γ mapping

Similarly by changing β, by increasing the particle damping, the input forces increase. As shown in Figure 1.3, the increase in β does not affect the natural frequency of the system; more so, it affects the response X. As the β increases, the response decreases.

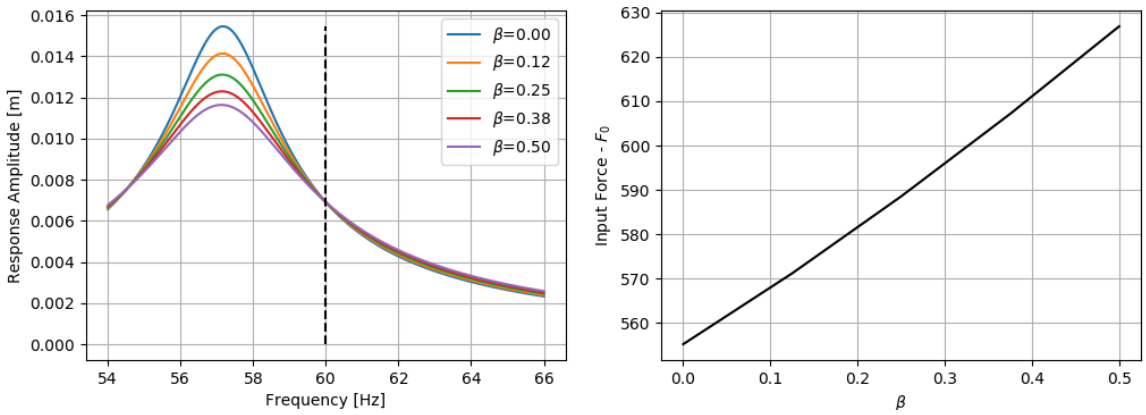


Figure 1.3: RAM - β mapping

Additionally, G is the mixing intensity as mentioned before and is defined by $G = (X\omega^2)/9.81$. In the case in which G is varied (see Fig. 1.4), the natural frequency will stay the same; however, the response will decrease as G increases because it decreases the response required from the springs and dampers.

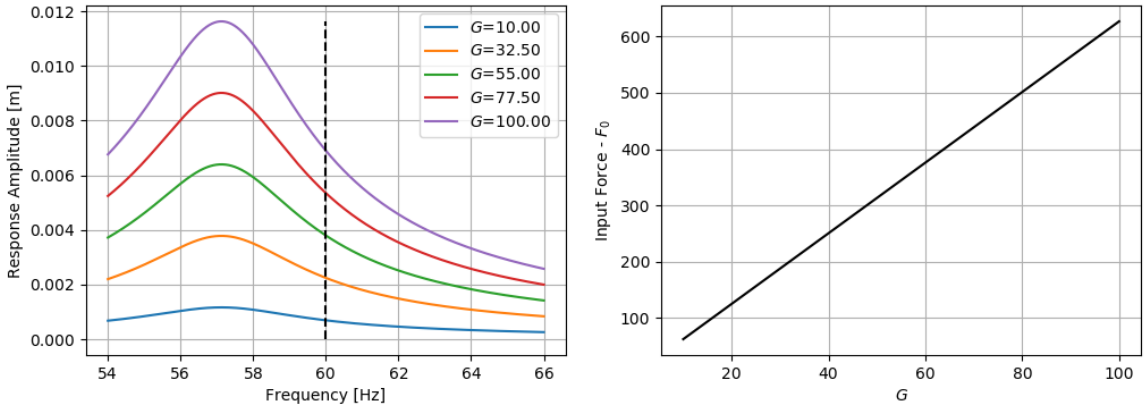
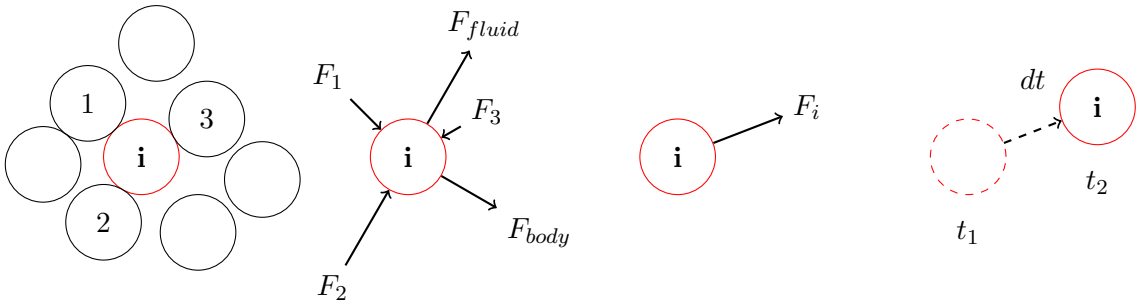


Figure 1.4: RAM - G mapping

1.3 Molecular Physics

1.3.1 Discrete Element Dynamics



(a) Contacts Identification (b) Forces application (c) Forces summation (d) Particle displacement computation

Figure 1.5: Example of a DEM iteration on a particle (Wills, 2016)

The Discrete Element Method (DEM) proposed by Cundall and Strack, 1979, allows for the simulation of numerous individual particles by resolving Newton's second law for each particle. Forces acting on the particles are typically body forces and interaction forces due to particle-particle or particle-wall collisions. In Figure 1.5, the DEM process is simply put forward. As seen in 1.5c, assuming the particle mass remains unchanged, then for each particle the sum of all external forces is equal to the mass times acceleration (Luding, 2008):

$$m_i \frac{d^2(x(t)_i)}{dt^2} = \mathbf{F}_i,\ i = 1, \ldots, N \tag{1.3}$$

By integrating over the sum of the forces in equation 1.3 on particle i, the velocity and positional components can be extracted. $\mathbf{F}_i$ is the sum of all forces imposed onto particle i by the contacting particles and external forces acting on particle j:

$$\mathbf{F}_i = \sum_{j \neq i} \mathbf{F}_{ij} + \mathbf{F}_{ext,i} \tag{1.4}$$

As contact laws usually express contact forces as tangential and normal components (unit vectors given by $\hat{e}_n$ and $\hat{e}_t$), the contact force between particle i and j can also be expressed as the sum of the normal and tangential contributions:

$$\mathbf{F}_{ij} = F_n \vec{e}_n + F_t \vec{e}_t \tag{1.5}$$

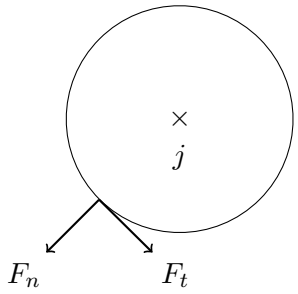


Figure 1.6: Contact forces diagram

The forces depicted in Figure 1.6 can then be decomposed as in the Hertz-Mindlin model on the contact of elastic solids as proposed by Hertz, 1881. The contact forces cause the particles to move in two manners. The normal force (F_n) causes the particle to translate in the direction of the normal force in the absence of other contact forces. As to where the tangential force (F_t) exerts a torque on the particle which causes the particle to translate, in the direction of the tangential force, it rotates in the direction of the torque around the centre of mass in the absence of other contact forces. Juergen Jaeger adds that: "*The global behaviour of impacting objects can be described by the equations of motion, while the contact force depends on the local contact law. It is, therefore, necessary to study contact mechanics, in order to describe the impact process correctly*" (Jaeger, 1994)

There are a few different contact models to resolve the local contact forces of which a few examples are (DEM Solutions Ltd., 2016):

- **The Hertz-Mindlin (no-slip) model** is a model with accurate and efficient force calculations. The normal force component is based on the Hertzian contact law (ibid.)(Hertz, 1881), and is the tangential component of work done by Mindlin, 1989 by using the Coulomb law of friction models to simulate the tangential friction force (DEM Solutions Ltd., 2016). This model also emits the effects of Van der Waals forces on the particles. The assumptions made by Mindlin at the time were simplified due to the absence of some modern techniques (Etsion, 2010) - this led to a few adaptations to cater for particles that cannot be modelled using these conditions.
- **Hertz-Mindlin (no-slip) with RVD Rolling Friction** is a model that is very similar to the no-slip model mentioned above. However, here the Coulomb friction laws are replaced by a Relative Velocity Dependent rolling friction model (DEM Solutions Ltd., 2016). The model is based on the relative velocity of two elements in contact (Zhou, Wright, Yang, et al., 1999).
- **Hertz-Mindlin with Archard Wear** is a model used to simulate the wear imposed on particles as they come in contact with each other (DEM Solutions Ltd., 2016).
- **Hertz-Mindlin with Bonding** is a model used to model particles bonded to each other. The particle bond is of finite thickness and can withstand a certain amount of deformity under a maximum normal and tangential shear stress. After the bond is broken, the particles interact with each other as per the normal Hertz-Mindlin model as solid hard spheres (ibid.) (Potyondy and Cundall, 2004). This model is typically used to simulate concrete or rock formations under stress.
- **Hertz-Mindlin with Heat Conduction** is used for densely packed particle volumes. The heat conduction between particles can be modelled with this contact model. As an added feature, this

model uses the Hertz-Mindlin contact model to calculate the conductive heat transfer between particles based on the contact area (DEM Solutions Ltd., 2016) (Chaudhuri, Muzzio, and Tomassone, 2006).

- **Hertz-Mindlin with JKR Cohesion** - The Johnson-Kendall-Roberts cohesion model calculates and models the influence of Van der Waals forces as particles approach each other. This model is useful for strong adhesive particles such as dry powders and wet materials (DEM Solutions Ltd., 2016) (Kenneth Langstreth Johnson and Roberts, 1971).
- **Linear Cohesion** - this model was developed by DEM Solutions Ltd., 2016, whereas the Hertz-Mindlin model is adapted to add a normal cohesion force to particles to simulate cohesion between particles (ibid.).
- **Hysteric Spring** - the hysteric spring contact model is used for particles subjected to plastic deformation and particles are subjected to elastic deformation under a certain stress threshold. If the stress threshold is exceeded, plastic deformation takes place and large particle penetration is possible without the need for excessive forces (ibid.) (Walton and Braun, 1986).
- **Linear Spring** - The linear spring model is a model developed by Cundall and Strack, 1979 simulating the normal and tangential force components with a spring and linear damper configuration (ibid.). The spring and damper properties are calculated with the use of material properties like particle elasticity and density.

As espoused, there are numerous contact models available. For the purpose of this study, only particle contact models that neglect cohesive and Van der Waals forces are considered. Furthermore, in this study, no heat transfer will be discussed.

For illustration purposes, the Hertz-Mindlin and Cundall and Strack Linear Spring models will be discussed. As for implementation, the Hertz-Mindlin model was chosen as it is one of the contact models implemented by StarCCM+ and a relevant model for this study.

1.3.2 Contact Models

Consider the contact between spherical particles i and j in Figure 1.7. Each of these particles has a position $\vec{x}$ and linear velocity $\vec{v}$, and additionally, each particle has a rotational velocity of ω and radii of r.

When particles are in contact the particle perimeters intersect as described by:

$$|\vec{x}_j - \vec{x}_i| - (r_i + r_j) < 0 \tag{1.6}$$

For this model the following normal and tangential unit vectors are considered (ibid.);

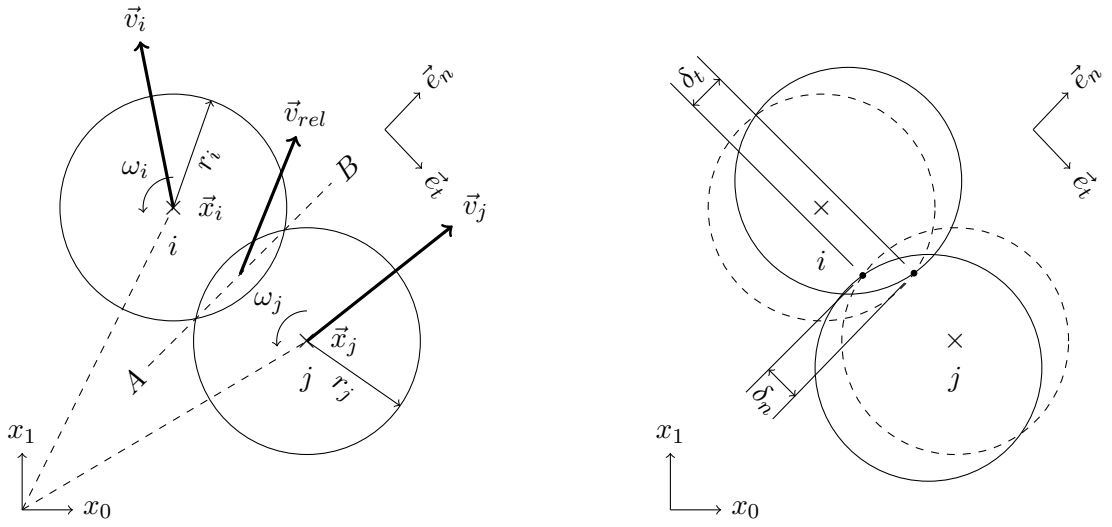
$$\vec{e}_n = \frac{\vec{x}_j - \vec{x}_i}{|\vec{x}_j - \vec{x}_i|} \tag{1.7}$$

Where the tangential unit vector is perpendicular to $\vec{e}_n$ in a clockwise rotation:

$$\vec{e}_t = (e_{n,y}, -e_{n,x}) \tag{1.8}$$

Thereafter the relative velocities of both particles can be defined by the summation of the linear velocities $\vec{v}_i$ and $\vec{v}_j$ and the tangential particle velocities due to the rotational velocities ω_i and ω_j:

$$\vec{v}_{rel} = (\vec{v}_i - \vec{v}_j) - (\omega_i r_i + \omega_j r_j)\vec{e}_t \tag{1.9}$$



(a) Contact Kinematics (b) Particle penetration distance

Figure 1.7: Contact diagram (Wassgren, 2008)

From the relative velocity, Cundall and Strack, 1979 propose that the penetration velocity $\dot{\delta}$ in either the normal or tangential directions are extracted with the use of the respective unit vectors as:

$$\dot{\delta}_n = \vec{v}_{rel}\vec{e}_n = (\vec{v}_i - \vec{v}_j)\vec{e}_n - (\omega_i r_i + \omega_j r_j)\vec{e}_t\vec{e}_n = (\vec{v}_i - \vec{v}_j)\vec{e}_n \tag{1.10}$$

In the normal direction the penetration velocity $\dot{\delta}_n$ is dependent on the linear velocities perpendicular to the surface on the contact plane AB as denoted in Figure 1.7. Similarly, the tangential penetration is dependent on a combination of the relative velocity parallel to the contact plane AB. This is done as follows:

$$\dot{\delta}_t = \vec{v}_{rel}\vec{e}_t = (\vec{v}_i - \vec{v}_j)\vec{e}_t - (\omega_i r_i + \omega_j r_j)\vec{e}_t\vec{e}_t = (\vec{v}_i - \vec{v}_j)\vec{e}_t - (\omega_i r_i + \omega_j r_j) \tag{1.11}$$

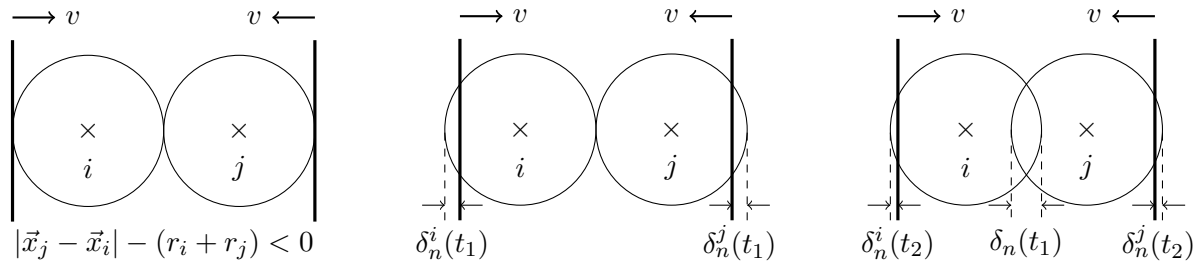
For the purposes of contact models, the normal and tangential penetration is needed to simulate and model the resultant normal and tangential forces F_n and F_t shown in Figure 1.6. Integrating $\dot{\delta}_n$ and $\dot{\delta}_t$ over time t will present δ_n and δ_t:

$$\delta_n = \int_{t_0}^{t} \dot{\delta}_n dt \tag{1.12}$$

$$\delta_t = \int_{t_0}^{t} \dot{\delta}_t dt \tag{1.13}$$

The following steps are performed in the distinct element method alternating between Newton's second law and contact laws to calculate the contact forces as illustrated by (ibid.), a time-step $\Delta t = t_1 - t_0$ is used for the numerical integration to calculate δ_n and δ_t. The steps are best described with a simple example, assuming two weightless particles are being compressed between two moving plates in the normal direction. The plates are moving towards each other at a constant velocity v. Initially, at $t = t_0$ the particles are in contact but no force is generated.

1. Two particles in Figure 1.8a come in contact at t_0 where the condition in equation 1.7 is satisfied.



(a) $t = t_0$ (b) $t = t_1 = t_0 + \Delta t$ (c) $t = t_2 = t_0 + 2\Delta t$

Figure 1.8: Particle contact resolution example
(Cundall and Strack, 1979)

2. At time $t1$ after time Δt, the plates have moved a distance of $v\Delta t$. Therefore, according to penetration models, the relative velocity in the normal direction is equal to the velocity of the plates $\vec{v}\vec{e}_n = v$. Integrating over time $\int_{t_0}^{t_1} v dt = vt_1 - vt_0 = v\Delta t$ produces the penetration distance, thus the normal penetration distance of particle i is $\delta_n^i(t_1) = v\Delta t$, as is shown in Figure 1.8b. A similar calculation is done for particle j. The total penetration distance is $\delta_n(t_1) = \delta_n^i(t_1) + \delta_n^j(t_1)$.

3. After calculating the penetration distance $\delta_n(t_1)$, the normal force component F_n can be calculated; this is done by using a contact law, such as the Hertz-Mindlin contact model.

4. Using Newton's second law, the particle acceleration for each particle can be calculated by, $\vec{a}_i(t_1)\vec{e}_n = F_n/m_i$ and $\vec{a}_j(t_1)\vec{e}_n = -F_n/m_j$; the acceleration $\vec{a}_i$ and $\vec{a}_j$ is assumed to be constant over Δt.

5. Calculating the particle velocities at t_2 of particles i and j with $\vec{v}_i(t_2)\vec{e}_n = (\vec{a}_i(t_1)\Delta t)\vec{e}_n$ and $\vec{v}_j(t_2)\vec{e}_n = (\vec{a}_j(t_1)\Delta t)\vec{e}_n$. Thus, updating the penetration velocity at t_2 according to equation 1.10 and integrating over Δt, the penetration distance $\delta_n(t_2)$ is given by:

$$\delta_n(t_2) = \int_{t_1}^{t_2} (\vec{v}_j(t_2) - \vec{v}_i(t_2))\vec{e}_n dt = \left(\frac{F_n(t_1)}{m_i}\Delta t - \frac{F_n(t_1)}{m_j}\Delta t \right) \Delta t \tag{1.14}$$

6. Repeat the steps from 3 to 5 until the contact is lost between particles.

From here, any contact model can be applied to estimate the contact forces for the specific application to the problem. For illustrative purposes, the two models to be considered are the Linear Spring model proposed by Cundall and Strack, 1979, as well as the Hertz-Mindlin model. These models are well established DEM models that show reputable results across a wide variety of problems. In the specific RAM problem in this study no additional complexities such as binding of particles or particles in a fluid. Thus a model where the Van der Waals forces between particles can be ignored can be chosen.

1.3.2.1 Hertz-Mindlin contact model

The Hertz contact model for normal forces was set out in 1881 with work done by Hertz, 1881. Initially, only the normal force components were set out. Later Mindlin, 1989 added the tangential force components which complete the contact force model as it is being used today. Figure 1.9 displays two particles i and j in contact. The penetration area is characterized by radius a.

For spherical particles in contact the Hertz contact model defines a as:

$$a = \sqrt[3]{\frac{3F_n E^*}{4r^*}} \tag{1.15}$$

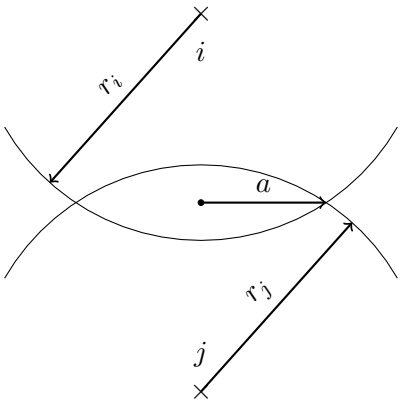


Figure 1.9: Hertz contact area

Thereafter the particle penetration distance δ_n is given by:

$$\delta_n = \frac{a^2}{r^*} = \sqrt[3]{\frac{9F_n^2}{16r^*E^{*2}}} \tag{1.16}$$

Rewriting this equation the principle normal force component F_n is defined as (DEM Solutions Ltd., 2016):

$$F_n^p = \frac{4}{3}E^*\sqrt{r^*}\delta_n^{\frac{3}{2}} \tag{1.17}$$

The Hertz model states that the normal force component is dependent on the equivalent particle Young's modulus E^* and equivalent radius r^* as defined by each particle i and j subscripts. The equivalent radius is defined by:

$$\frac{1}{r^*} = \frac{1}{r_i} + \frac{1}{r_j} \tag{1.18}$$

The equivalent Young's modulus for isotropic materials is defined by the particle Poisson's ratio ν and Young's modulus E:

$$\frac{1}{E^*} = \frac{(1-\nu_i^2)}{E_i} + \frac{(1-\nu_j^2)}{E_j} \tag{1.19}$$

The normal component damping force is then calculated as:

$$F_n^d = -2\sqrt{\frac{5S_nm^*}{6}}\beta\dot{\delta}_n \tag{1.20}$$

The equivalent mass m^* of the particles is calculated as:

$$\frac{1}{m^*} = \frac{1}{m_i} + \frac{1}{m_j} \tag{1.21}$$

Where β and the normal stiffness S_n, where R_n is the coefficient of restitution, is calculated by:

$$\beta = \frac{\ln R_n}{\sqrt{\ln^2 R_n + \pi^2}} \tag{1.22}$$

$$S_n = 2E^* \sqrt{r^* \delta_n} \tag{1.23}$$

The total normal force F_n is defined by:

$$F_n = F_n^p + F_n^d \tag{1.24}$$

The tangential components are then defined by two components, the tangential principle force F_t^p and the tangential damping force F_t^d. The tangential stiffness component S_t is dependent on the equivalent shear modulus G^*:

$$\frac{1}{G^*} = \frac{2-\nu_i}{G_i} + \frac{2-\nu_j}{G_j} \tag{1.25}$$

This for isotropic materials is defined by:

$$\frac{1}{G^*} = \frac{2(2-\nu_i^2)}{E_i} + \frac{2(2-\nu_j^2)}{E_j} \tag{1.26}$$

The principle tangential force is calculated by the tangential stiffness S_t and equivalent shear modulus:

$$S_t = 8G^* \sqrt{r^* \delta_n} \tag{1.27}$$

The principle tangential force is defined by:

$$F_t^p = -S_t \delta_t \tag{1.28}$$

The tangential damping component is defined by:

$$F_t^d = -2\sqrt{\frac{5S_t m^*}{6}} \beta \dot{\delta}_t \tag{1.29}$$

The tangential force component is limited by the Coulomb friction law between particles where μ is the static friction coefficient of the contact. Therefore the tangential force is defined by:

$$F_t = \begin{cases} F_t^p + F_t^d, & F_t \geq \mu F_n \text{ (slipping)} \\ \mu F_n, & F_t < \mu F_n \text{ (no} - \text{slip)} \end{cases} \tag{1.30}$$

1.3.2.2 Linear spring contact model

Figure 1.10 represents the Cundall Strack contact model; both the normal and tangential force components have an elastic spring (k) and damping component (ζ). The tangential component has an added Coulomb friction (μ) component (Cundall and Strack, 1979).

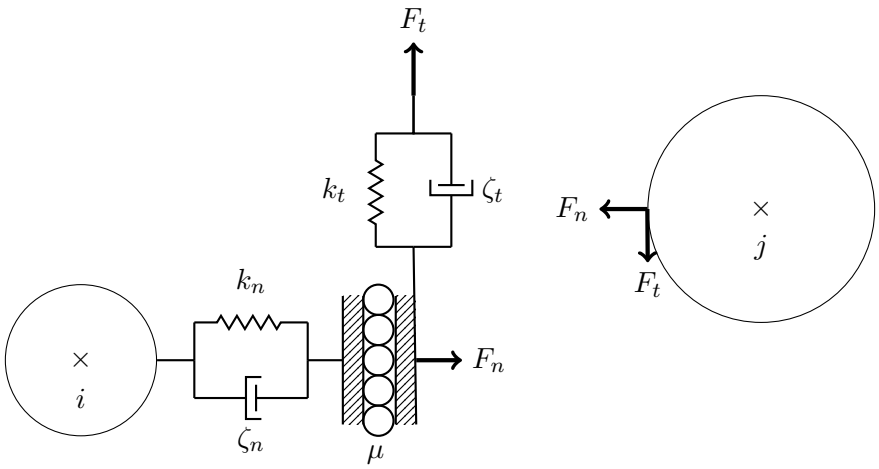


Figure 1.10: Cundall and Strack linear spring contact model

By calibrating the spring stiffness constant k one can simulate the contact forces between particles. One of the popular methods of calibrating the spring stiffness is by calculating the maximum strain energy in pure Hertzian contact of the particles in the system. This is done by using the following properties, the equivalent radius r^*, mass m^* and Young's modulus E^* components as defined in equations 1.18, 1.21 and 1.19. Additionally, the typical impact velocity V is used (DEM Solutions Ltd., 2016).

$$k = \frac{16E^*\sqrt{r^*}}{15}\left(\frac{15m^*V^2}{16E^*\sqrt{r^*}}\right)^{1/5} \tag{1.31}$$

The typical impact velocity of a simulation is selected in line with what the particles are subjected to. For instance, in the case of RAM, the typical impact velocity V will be selected to be the same as the maximum linear velocity of the mixing vessel. The damping component ζ is related to the spring stiffness k and particle restitution R:

$$\zeta = \sqrt{\frac{4m^*k}{1+\left(\frac{\pi}{\ln R}\right)^2}} \tag{1.32}$$

The normal force F_n is then defined by:

$$F_n = k\delta_n + \zeta\dot{\delta}_n \tag{1.33}$$

Similarly, the tangential force component is defined by F_t, where, if the particles overcome the friction component μ, then the tangential force is defined by the slipping force μF_n:

$$F_t = \begin{cases} k\delta_t + \zeta\dot{\delta}_t, & F_t \geq \mu F_n(\text{slipping}) \\ \mu F_n, & F_t < \mu F_n(\text{no}-\text{slip}) \end{cases} \tag{1.34}$$

At this point, one can look back and see that there are a couple of particle properties and external factors that affect the contact between particles, assuming no cohesive and or Van der Waals forces are present and also neglecting any heat transfer between particles. These parameters in turn influence how the particles move and flow within this system. Firstly, the particle properties are:

- Particle size in as expressed in its radius, r.
- Elasticity, E.
- Density, ρ.
- Restitution, R.
- Poisson's ratio, ν.

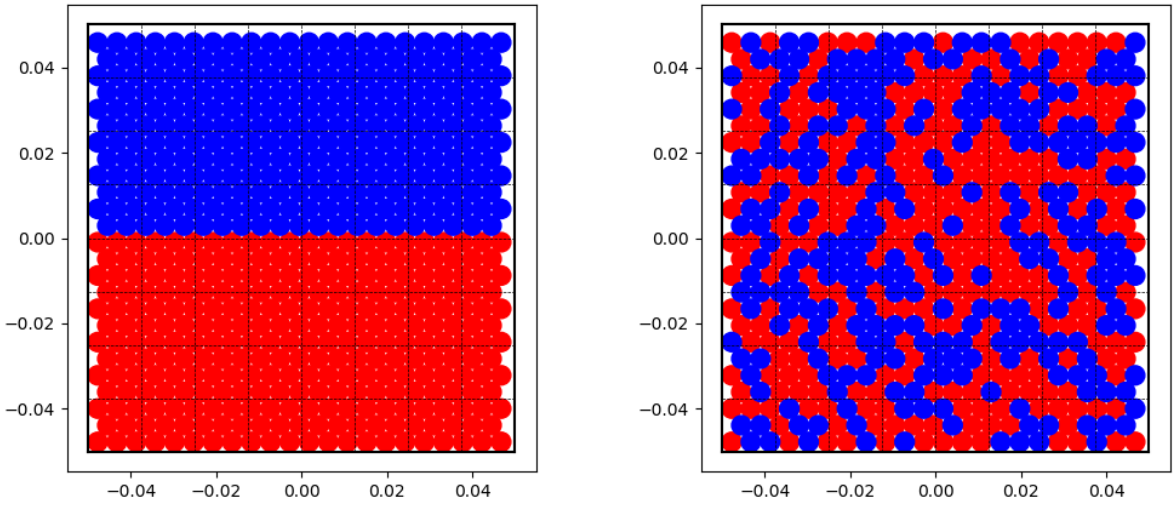
Secondly, the external factors are;

- Spatial constraints or boundaries.
- External forces applied to the system, $\mathbf{F}_{ext}$.

1.4 Mixing Quantification

Following a basic understanding of the simulation models used to simulate particle kinematics, attention is given to ways that mixing performance is quantified. Numerous mixing models have been proposed thus it is, therefore, critical to properly understand the benefits and limitations of the various mixing indexes to make an informed decision on how to quantify mixing in this study.

The mixing index (M) of fluid and granular materials is typically defined by a value between 0 and 1, denoting the degree or quality of mixing in the substance (Wiley and Weihs, 2016). Where $M = 0$ is segregated, as shown in Figure 1.11a, and $M = 1$ shown in Figure 1.11b, it indicates a fully mixed state. Binary mixing is the result of a mixture between two components - in this study, only binary mixing will be discussed and used for comparison sake.



(a) Unmixed binary state

(b) Mixed binary state

Figure 1.11: Mixed states

The different mixing indexes classify the mixing state in several ways:

- A Particle domain is randomly transformed from its initial particle locations.
- A Smaller sample, such as a cell in the volume, contains equal amounts of each particle type (given that there are equal amounts of particles in the greater domain).
- Neighbouring particles are of equal amounts to a sample particle (Akao, Shindo, Yagi, et al., 1976) (given that there are equal amounts of particles in the greater domain).
- The distance a particle has moved throughout the domain.

The focus of this section will solely be on the theoretical setup of each of the methods and the logic behind the method.

This chapter will outline the various methods of determining the mixed state of a sample, and highlight theoretical insights into each. The mixing indexes under consideration are classified as follows:

- Sampling methods:
  - Nearest neighbours (Godlieb, Gorter, Deen, et al., 2010)
  - Coordination number (Chou, Song, and Hsiau, 2017)
  - Particle Scale Index (Chandratilleke, Yu, Bridgwater, et al., 2011)
- Displacement based methods:
  - Average Height method (Godlieb, Gorter, Deen, et al., 2010)
  - Neighbour distance method (ibid.)
  - Siiria method (Siiriä and Yliruusi, 2009)
- Cell dependant methods:
  - Lacey (Lacey, 2007)
  - Mixing entropy (Arntz, Otter, Briels, et al., 2008)
- Neighbourhood method:
  - Chain Length

1.4.1 Sampling methods

Sampling methods determine the mixed state by taking a random sample out of the domain and tracking it as the particle moves through the domain. The sample size can range from one particle to all the particles in the system. The accuracy and computing time of the index is dependent on the number of samples taken. The more sample particles are taken the longer the computing time will be. The fewer particles are taken in a sample the quicker the computational time.

These methods generally only take into regard the particle type, providing some insights into the segregation patterns in the domain and not necessarily the mixing performance.

1.4.1.1 Nearest neighbours (NN)

The nearest neighbours (NN) method looks at the closest neighbouring particles and counts the number of each particle type (see type A and B in Figure 1.12, where dashed particles are considered outside of the neighbourhood) to determine the fraction of each particle type present. Neighbouring particles are

the closest number of particles to the sample particle; this number is assumed to be the coordination number. The sample particle is considered perfectly mixed if the fraction is $n_{diff}/n_C = 0.5$, showing that there are equal amounts of each particle type in the neighbourhood. n_{diff} is the number of particle types which is different to the number of particles in a sample. The coordination number (n_C) is the number of particles in contact with a certain particle in any given packing order. Using equally sized particles in a 3D face-centred cubic packing structure the coordination number is 12 (Godlieb, Gorter, Deen, et al., 2010).

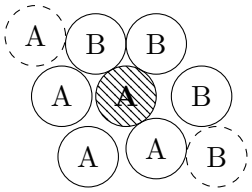


Figure 1.12: Nearest neighbours - $n_C = 6$

The Nearest neighbours mixing index (M_{NN}) is calculated as the average nearest neighbour mixing and is given by the following, where N' is the number of particles in the sample set:

$$M_{NN} = \frac{2}{N'(n_C)} \sum_{i=1}^{N'} n_{(i)diff} \tag{1.35}$$

The method produces an accurate mixing index at a local level and is generally accurate for most DEM applications. This method is useful for:

- Quick spot-checking with small sample sizes;
- A robust method to implement in most applications

With slight drawbacks such as:

- Larger sample sizes are needed to accurately predict the mixing quality of the domain.
- The particles do not always touch the sample particles, and thus this is not a method to be used when particles are suspended in the domain nor when sampling close to the boundary, as seen in Figure 1.14 - this will cause an inaccurate computation.
- It is difficult to use in experimental observations as this method remains a computational tool.

1.4.1.2 Coordination number

The Coordination number (CN) method is very similar to that of the NN method and differs mainly in the way the closest particles to the sample particle is defined (Chou, Song, and Hsiau, 2017). The method considers any particle within close range (usually 10% of the sample particle's radius) to be considered particles for mixing (see Figure 1.13). In this case, the dashed lined particles are not considered to be close range and are excluded from the sample subset.

Likewise, in *equation* 1.36, it is observed that similarities between the Coordination number method and that of the Nearest neighbours, with the change being a dynamic coordination number n'_C which adjusts based on each sample particle and the number of particles in close range to it.

$$M_{CN} = \frac{2}{N'} \sum_{i=1}^{N'} \frac{n_{(i)diff}}{n'_{(i)C}} \tag{1.36}$$

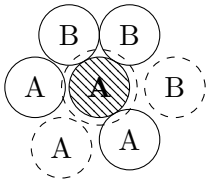


Figure 1.13: Coordination number (ex. $n_C = 4$)

The Nearest neighbour's method uses a blanket approach when it comes to the coordinate number, assuming a constant value for all samples, whereas the Coordination number method dynamically selects a new coordination number for each sample particle. The method is, in particular, more powerful than the nearest neighbours approach in:

- Determining the mixing index in mixtures of different particle sizes. Considering that particles follow the Cation-anion radius ratio rule as proposed by Hüttig in 1920. The rule states that the relationship between the radii of the particles affects the coordination number. The ratio r_c/r_a uses the annotation of anion and cation to denote the different particle sizes. Table 1.1 shows the relationship between the radius ratio and the coordination number.

Table 1.1: Cation-anion radius ratio rules (Toofan, 1994)

Cation-anion radius ratio $[r_c/r_a]$	**Coordination number** $[n_C]$
< 0.155	1
$0.155 - 0.255$	3
$0.255 - 0.414$	4
$0.414 - 0.732$	6
$0.732 - 1.000$	8

- Determining the mixture quality on the boundary of the volume where the particles may not have the same coordination number as in the mixture.

  Consider the scenario in Figure 1.14, the Coordination number (CN) mixing with respect to the sample particle highlighted would be $M_{CN} = 2(0/3) = 0$, however the Nearest neighbour method will produce a mixing index of $M_{NN} = 2(3/6) = 1$. In this case, the CN mixing index more accurately describes the boundary mixing behaviour.

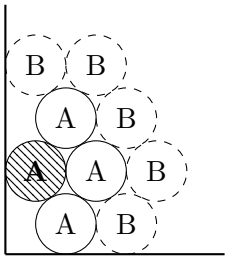


Figure 1.14: Boundary case - Coordination number versus. Nearest neighbour

Unfortunately the method still does not account for suspended mixtures as this method specifically relies on the particles to be in contact.

1.4.1.3 Particle Scale Index

The Particle Scale Index (PSI) method is used to express binary mixing and is based on the Nearest neighbours method and the Lacey method, another popular mixing index that will be covered in section 1.4.3.1. This is achieved by analysing the variance in mixing quality for a sample particle (Chandratilleke, Yu, Bridgwater, et al., 2011).

Firstly determine the particle ratio p_i and the average particle ratio $\bar{p_i}$ for all sample particles i in subset N;

$$\begin{aligned} p_i &= \frac{n_{(i)diff}}{n_{(i)C}+1} \\ \bar{p_i} &= \frac{1}{N'}\sum_{i=1}^{N'} p_i \end{aligned} \tag{1.37}$$

The user can decide on a method of determining the coordination numbers for each sample. For accuracy and simplicity sake the contact model used in this case is the Nearest neighbour method (Section 1.4.1.2) assuming a constant n_C throughout the calculation. Secondly, calculate the particle ratio variance based on the values calculated in equation 1.4.1.3:

$$\sigma^2 = \frac{1}{N'}\sum_{i=1}^{N'}(p_i - \bar{p_i})^2 \tag{1.38}$$

For the method to function as it should the particle ratio variance over the domain should be compared to an initial variance state - σ_0^2, this can be done by assuming the initial state was perfectly segregated and the particle ratio of one particle type is given by p_0. The particle ratio p_0 can also be expressed as the fraction of the number of particles of one particle type (N_α) divided by the total number of particles (N) in the domain $p_0 = N_\alpha/N$. Since the PSI method is only suitable for binary mixing applications the initial variance state can be simplified and expressed by:

$$\sigma_0^2 = p_0(1-p_0) \tag{1.39}$$

Adapting this domain-based variance to a particle type variance it is necessary to incorporate the contact number (plus one to account for the sample particle) into the variance, by introducing σ_R^2 as follows:

$$\sigma_S^2 = \frac{\sigma_0^2}{n_C+1} \tag{1.40}$$

Now it is possible to compare the various variances; the idea is to compare the adjusted sample variance $\sigma^2 - \sigma_0^2$ to the overall adjusted variance $\sigma_S^2 - \sigma_0^2$.

$$M_{PSI} = \frac{\sigma^2 - \sigma_0^2}{\sigma_S^2 - \sigma_0^2} \tag{1.41}$$

Additionally consider the sample size for a moment - if a pseudo mixing scheme was set up for a large number of particles in a mixed state (where the contact number is $n_C[A] = 12 - \beta(n_C, 0.5)$ where β is sampled from a Binomial distribution with a mean of 0.5). Figure 1.15 then shows the PSI mixing variance across a sample set of 500 samples. Not all the particles have to be sampled to get an accurate result of the mixed state. By using Figure 1.15 as a proxy, one can gauge the level of accuracy of the mixing index based on the sample size taken.

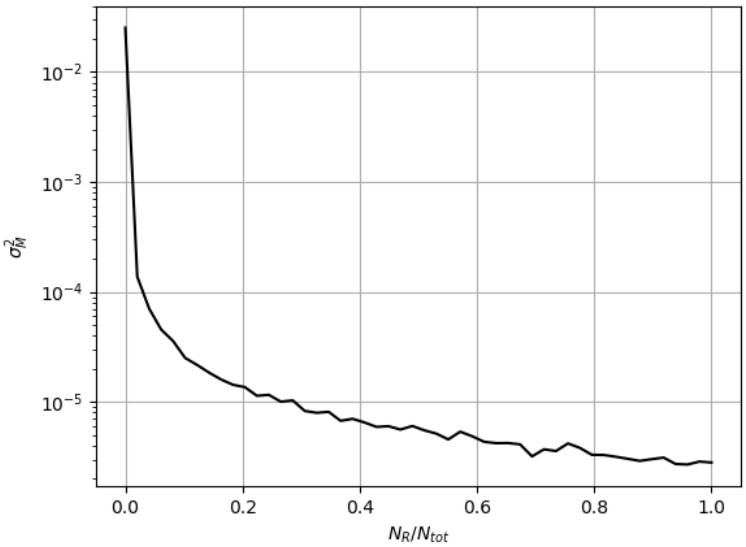


Figure 1.15: PSI Mixing Variance

This type of mixing index tells us a bit more about the segregation patterns and is thus by far one of the better-defined methods. This method can be used for:

- segregation analysis if the sample size is adequate;
- robust method for final state analysis.
- adjusting to account for particle amounts per particle type by using σ_0^2 - to ensure there is an even distribution of particles for each particle type throughout the domain.

1.4.2 Displacement dependant methods

Displacement based methods have no relation to particle type and only considers the displacement change in the particle location. This is a volatile premise to determine particle segregation patterns.

Some of the displacement dependant methods go hand-in-hand with some of the sampling methods. The displacement-based methods are thus a better indication of the development of the mixing.

1.4.2.1 Average Height Method

Godlieb, Gorter, Deen, et al., 2010 proposed the Average Height method (AHM) which uses the particle packing height as a determination method of the mixing index. If the packing height (centroid height) of

two distinct materials have a large difference, the material is considered unmixed, but if the materials have centroids close to each other, the material is considered mixed (Godlieb, Gorter, Deen, et al., 2010).

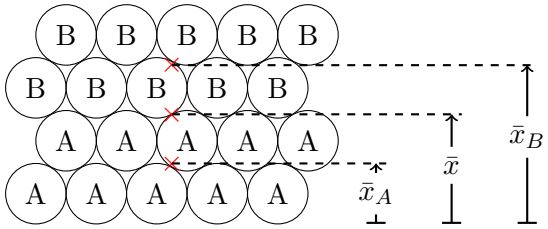


Figure 1.16: Average Height Method

Two distinct particle sets, A and B, is represented in Figure 1.16. In this case, the layer of particles A is on the bottom with a centroid $\bar{x}_A$ and particle set B which is layered on top. The overall particle centroid is given by $\bar{x}$. In the AHM, only one particle type centroid in conjunction with the overall centroid can be used to calculate the MI. Here particle type A will be used. The respective centroids are determined by:

$$\begin{aligned} \bar{x}_A &= \frac{1}{N_A}\sum_{i=1}^{N_A} x_{(i)A} \\ \bar{x} &= \frac{1}{N}\sum_{i=1}^{N} x_{(i)}. \end{aligned} \tag{1.42}$$

Followed by finding the overlap of these centroids, this is done by calculating X as follows:

$$X = \frac{\bar{x}_A}{\bar{x}} \tag{1.43}$$

If $X = 0.5$ the mixture is considered segregated as the centroids $\bar{x}_A$ and $\bar{x}$ are separated. When $X = 1$, the centroids line up and the mixture is considered fully mixed. To normalize the mixing index between 0 and 1, the following adjustment is made to calculate M_{AHM}.

$$M_{AHM} = 2(X - 0.5) \tag{1.44}$$

Currently this method is set up to calculate the mixing index in one direction only, the particle heights can however be calculated in 2 and 3 dimensions, depending on the information that is sought after. This phenomenon will be explored in section 2.2.1.

The method is commonly used in suspended particle applications where there is not always contact between particles. Some of the positive outcomes of this method are;

- It is a robust method to implement.
- It is a method that is suited for both numerical and experimental applications, as can be seen by the use of image recognition by ibid. and some high-speed photography (ibid.).

Although there are some drawbacks because the method;

- Is not always the most accurate measurement tool and can easily misrepresent what is going on in the data.

- As seen in the paper by Godlieb, Gorter, Deen, et al., 2010, once the particle heights move past each other it is possible for $M_{AMH} > 1$. This can be confusing if misrepresented as the other methods typically scale between 0 and 1.

1.4.2.2 Neighbour distance

The Neighbour distance (ND) method (ibid.) has been developed alongside the Nearest neighbours method to quantify not only particle type mixing but also to incorporate the thoroughness of the mixing. The method considers three particles, α, β and γ. Where particle α is the sample particle to base the mixing index from, particle β is the nearest neighbouring particle to α and particle γ is a randomly selected particle in the domain. Particle α is part of a random sample N. By tracing the particles throughout the domain, and comparing the distances between α and β and the distance between α and γ is what the mixing index is based on.

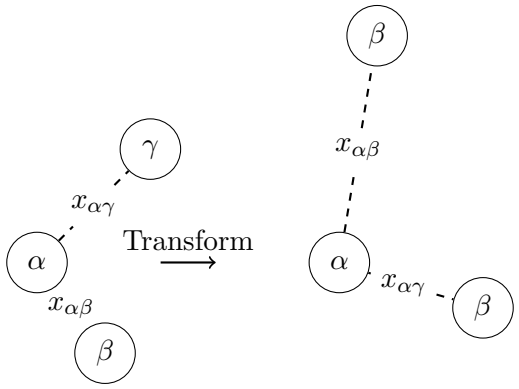


Figure 1.17: Neighbour distance

In Figure 1.17 the distance between the particles is shown by $x_{\alpha\beta}$ and $x_{\alpha\gamma}$. By subtracting the average particle diameter $\bar{d}$ from the particle distances $x_{\alpha\beta}$ and $x_{\alpha\gamma}$, comparison can be made between how far the nearest neighbour (β) to particle α has moved in relation to the other particles in the domain, represented by γ. As shown in *eq.* 1.45:

$$M_{ND} = \frac{\sum_{i=1}^{N}(x_{(i)\alpha\beta} - \bar{d})}{\sum_{i=1}^{N}(x_{(i)\alpha\gamma} - \bar{d})} \tag{1.45}$$

The nearest distance method can be used;

- To determine mixtures with suspended particles, as the relative distances are the determining factor to quantify the mixing index.
- The mixing index is basic to implement and contains information on particle movement.

Some of the issues that can be experienced with using this method are:

- If the particles are cohesive, the neighbouring particle β could potentially hang onto sample particle α and skew the results as to how mixed overall the domain is.
- The accuracy is determined by the sample size N of α particles.

1.4.2.3 Siiria

The following method was proposed by Siiriä and Yliruusi, 2009, where the algorithm compares the current state to a predefined random state. This is done once again by the change in distance between particles in a sample N as defined by x_{ij}, where i and j are iterators in the sample matric $[N \times N]$. Defining S as a current mixed state at time t:

$$S(t) = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} \Gamma^{g(x_{ij}(t_0))} \Big(1 - \Gamma^{g(x_{ij}(t_0) - x_{ij}(t))}\Big) \tag{1.46}$$

The algorithm has two adjustable constants; Γ a sensitivity parameter and g an exponent scaler. Γ scales the sensitivity of the change in particle distance ($0 < \Gamma \geq 1$).

Where g scales the distance travelled of each particle, this helps to scale the exponents so that the S values remain within a reasonable computational tolerance. To compare the time-dependent values of $S(t)$, a randomly ordered state of the initial state is used as a calibration to quantify the mixing of the Siiria method:

$$M_S(t) = \frac{S(t)}{S_0} \tag{1.47}$$

The Siiria method uses a random state as input for calculating the mixing index; this is the first method discussed in this study that does this. In that sense, how the random transform is performed can have an effect on the results. Thus the Siiria method can be applied accurately depending on the calibration of Γ and g as well as the confidence in the mixed state of the random state.

Additional arguments to consider are:

- The method does not regard particle types as defining the mixed state and may fail at identifying segregation patterns in the domain.
- Adding an extra sampling dimension j increases the computation time.
- Calculating the displacement of particles in the domain experimentally can be difficult which makes this method difficult to implement experimentally.

1.4.3 Cell dependant methods

Cell dependant methods divide the mixing domain up into cells of equal size and geometry. This is done by calculating particle fraction per cell. Depending on the method, the particle fraction is used to calculate a single mixing index over the domain.

One of the key advantages of the cell-based methods is the ability to map out regions or hotspots in the domain that contributes to most of the mixing.

1.4.3.1 Lacey

The Lacey method is one of the oldest methods of determining a binary mixing degree and is based on the use of a particle fraction in a cell array in the domain. This method of determining the mixing index uses the statistical probability of a certain particle type in each cell (Lacey, 2007) (Wen, Liu, Liu, et al., 2015) (Tasirin, Kamarudin, and Hweage, 2009).

A typical cell array can be visually described as in Figure 1.18. In each cell c the particle fraction p_c is calculated by counting the number of a particle type $n_{c(A)}$ and dividing it by the total number of particles in that cell n_c.

$$p_c = \frac{n_{c(A)}}{n_c} \tag{1.48}$$

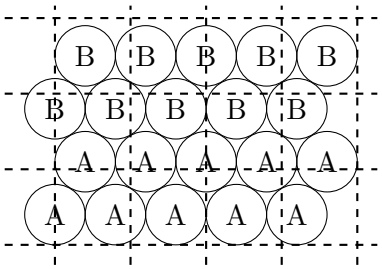


Figure 1.18: Domain cells

Furthermore the global particle fraction p_m can be calculated by the number of the particle type n_A and dividing it by the total number of particles in the volume n.

$$p_m = \frac{n_A}{n} \tag{1.49}$$

The global particle fraction can be seen as an average particle fraction across the domain. Similar to the PSI method discussed in section 1.4.1.3 the variance of particle fractions across the domain can be calculated, where N_c is the total amount of cells in the domain:

$$\sigma^2 = \frac{1}{N_c - 1} \sum_{c=1}^{N_c} (p_c - p_m)^2 \tag{1.50}$$

The spatial σ_0^2 and cell variances σ_c^2 are used to compare the particle cell variance σ^2, where $\bar{n_c}$ is the average number of particles in each cell:

$$\begin{aligned} \sigma_0^2 &= p_m(1 - p_m) \\ \sigma_c^2 &= \frac{\sigma_0^2}{\bar{n_c}} \end{aligned} \tag{1.51}$$

Following the mixing index:

$$M_L = \frac{\sigma^2 - \sigma_0^2}{\sigma_R^2 - \sigma_0^2} \tag{1.52}$$

The Lacey method has a couple of strong points:

- The method can identify segregation patterns depending on the cell resolution and position.

- Can identify regions where mixing is more active.
- As it is one of the older and more established methods it is a benchmark index and is used in many applications for this reason.

Although the Lacey method is strong in some areas, it has a pitfall.

- For larger particles in a small domain it becomes difficult to apply the method as there is too little information per cell to accurately calculate the variances needed.

1.4.3.2 Mixing Entropy

The Mixing entropy (ME) method is based on the Boltzmann gas mixing energy laws for binary mixing (Arntz, Otter, Briels, et al., 2008). The Mixing entropy method uses similar cells as the Lacey method. The local mixing energy per cell $e(i)$ is calculated as follows, where $p_{c(A)}$ and $p_{c(B)}$ is the particle fractions, as per cell, as determined by equation 1.4.3.1 for each particle type A and B.

$$e(c) = p_{c(A)} \ln p_{c(A)} + p_{c(B)} \ln p_{c(B)} \tag{1.53}$$

The total mixing energy over the domain is thus determined by the average of the mixing energy over the domain weighted by the number of particles per cell.

$$E = \frac{1}{N} \sum_{i=1}^{N} e(c) n_c \tag{1.54}$$

To normalize the energy calculated per cell the overall potential energy E_{mix} can be calculated by looking at the particle fraction in the domain, p_m is the global particle fraction as determined in equation 1.4.3.1.

$$E_{mix} = p_m \ln p_m + (1 - p_m) \ln (1 - p_m) \tag{1.55}$$

The mixing index is then calculated by using dividing the total mixing energy by the potential energy:

$$M_{ME} = \frac{E}{E_{mix}} \tag{1.56}$$

The Mixing entropy method has similar benefits and downfalls to the Lacey method; one added benefit is that the Mixing entropy method takes the particle count per cell in consideration. This ensures that the results are not skewed towards cells that have one or two particles per cell but rather accurately predict the mixing intensity.

1.4.4 Neighbourhood Approach

For some of the applications in particle physics, the methods mentioned thus far do not provide enough information on the segregation patterns that can take place in the domain. In this section, a new method of determining a mixing index is proposed. The neighbourhood approach identifies similar particles in contact with each other that form the neighbourhood. The neighbourhood is not limited to the number of particles as in the Nearest neighbours method. The larger the neighbourhood grows, the more segregated the sample, and the smaller the neighbourhoods are, the more mixed the sample gets.

1.4.4.1 Chain Length

The Chain length (CL) method is loosely based on the method of following the path of least resistance. The algorithm uses a random particle as an initiation point to then choose another neighbouring particle that has the highest amount of similar neighbouring particles. Creating a chain of particles that naturally terminates itself by either running into itself or running out of neighbouring particles.

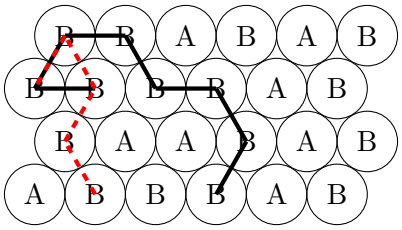


Figure 1.19: Chain length method

A sample set of N particles are used as initiation points, and if two chains of similar particle types intersect, the particle chain is concatenated to the previous one forming a larger neighbourhood as shown in Figure 1.19. This increases the chances of defining a neighbourhood properly.

The chain length fraction η_l is defined by the number of particles in the chain n_l, and the number of particles of that type in the domain N_t.

$$\eta_l = \frac{N_t - n_l}{N_t} \tag{1.57}$$

In Figure 1.19 there are 16 B particles. One chain of B particles is created with seven linked particles and the other chain of B particles of five. By concatenating the chains, the total length is ten particles $\eta = 0.375$.

The average chain length fraction then gives the mixing index as, where N_l is the number of chains:

$$M_{CL} = \frac{1}{N_l} \sum_{l=1}^{N_l} \eta_l \tag{1.58}$$

When implementing the chain length method, it is useful to consider the following.

- The method accurately represents the mixing index, granted the chain does not terminate prematurely. Thorough sampling is needed to ensure the mixing index is accurately determined.
- To run an algorithm such as the chain length method can be numerically taxing, especially in segregated samples where longer chains exist.

1.4.4.2 Growth Point

Similarly to the Chain length method, the Growth point (GP) method uses a random sample as an initiation point and includes all neighbouring particles from the same type as part of the neighbourhood. The total amount of particles in each neighbourhood is used to determine the particle fraction of each neighbourhood by equation 1.4.4.1. Similarly, the mixing index is determined by the average chain length fraction in 1.4.4.1.

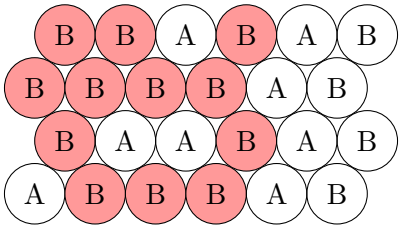


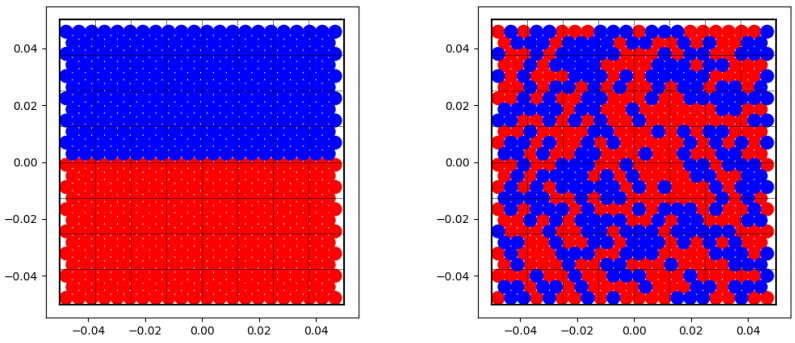
Figure 1.20: Growth point method

In Figure 1.20, the same example is used as in Figure 1.19. In this case, the neighbourhood has 12 B particles in and a total of 16 B particles in the domain, producing $\eta = 0.25$.

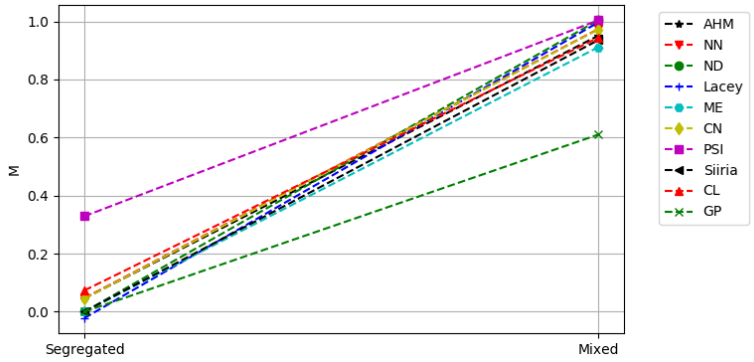
1.4.5 Summary

Illustrated in Figure 1.21, a sample is given in both its unmixed and mixed states. In Figure 1.21c the results are shown as per Table 1.2. The typical application of each mixing index method is explained in this chapter. We have seen similar results as expected. However, there are some key points to consider which will be discussed in the rest of the study.

- In this example, the mixing indexes show the expected behaviour where the unmixed state produces a mixing index of $M \approx 0$, and the mixed state a mixing index of $M \approx 1$. However, in this example, no information is given about the transient mixing state. When a volume is mixed a lot can be learnt from the transient behaviour, such as how quickly does the mixture reach a mixed state?
- The Particle Scale method shows a fairly high mixing index for the segregated state. This phenomenon will be unpacked in Section 2.1.3.
- The Growth Point method shows a similar behaviour and this might be due to the lack of a normalizing calculation used in some mixing indexes such as - Lacey, Mixing Entropy and PSI.
- The PSI method shows a mixed state of $M > 1$. This is discussed in Section 2.1.3.



(a) Segregated (b) Mixed



(c) Results

Figure 1.21: Mixed and segregated example

Table 1.2: Mixed state Mixing Performance (Fig. 1.20)

Method	**Segregated state**	**Mixed state**
Average Height	0.046	0.951
Nearest Neighbours	0.047	0.973
Neighbour distance	0.000	1.004
Lacey	-0.022	0.996
Mixing Entropy	0.000	0.911
Coordination Number	0.045	0.972
Particle Scale Index	0.329	1.004
Siiria	0.000	0.935
Chain Length	0.025	0.944
Growth Point	0.000	0.610
Average	**0.052**	**0.966**

Chapter 2

Mixing Index Review

In this chapter a study is conducted to identify three suitable mixing indexes that will be used in the remainder of the study. Adopted is a similar approach to Wen, Liu, Liu, et al., 2015, where several carefully designed mixing scenarios to identify differences between the mixing indexes are being considered.

In this chapter, a reverse engineering approach will be used to compare methods with each other as seen in Figure 2.1. This is achieved by selecting a randomly mixed state as the initial state and the fully mixed state as a selected segregation pattern. By following this approach one can try and prove how well some of the mixing indexes perform to identify segregation patterns.

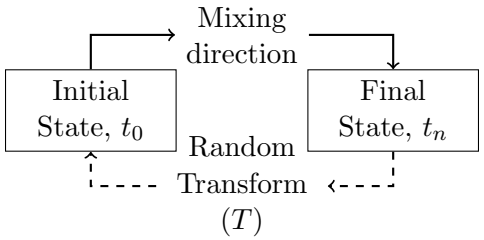
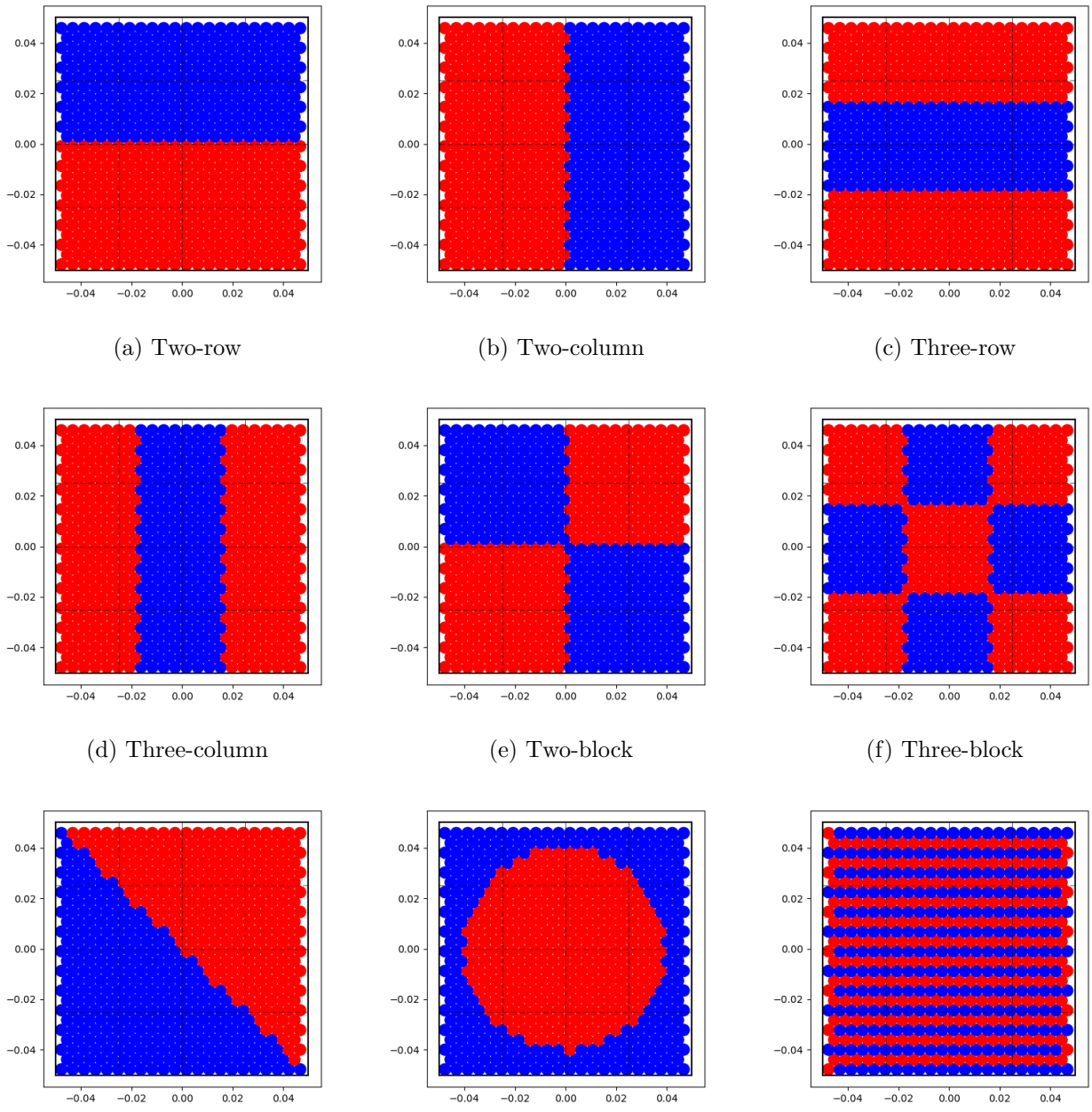


Figure 2.1: Hypothetical Mixing state

The initial state will be generated by a randomized transform T and will be kept constant throughout the chapter to compare each result accurately. There are several factors that influence the mixing index when identifying segregation patterns. These factors are:

- The number of segregation clusters.
- The number of contacting particles on a boundary layer between segregation layers.
- Configuration of segregation patterns, such as pattern rotation.

For the first test case let us consider a 2-dimensional particle-matrix with various segregation patterns. The volume in this case ($100 \times 100 \times 4.5mm$) is packed with 538 spherical particles of $4.5mm$ in diameter. The patterns are set up in such a manner that should an inspector visually inspect the result, he/she would declare the result as either mixed or unmixed - Figure 2.2 shows these cases. There are many more patterns that can be used to illustrate some behaviour and one can be quite creative in a 3-dimensional environment, but for the purposes of this chapter, these cases will show what is needed.



(a) Two-row (b) Two-column (c) Three-row

(d) Three-column (e) Two-block (f) Three-block

(g) Diagonal (h) Circle (i) Teeth

Figure 2.2: Hypothetical mixed cases

The motivation behind each hypothetical scenario is as follows:

- The two scenarios of two layers depicted in Figures 2.2a and 2.2b are considered to ensure that MIs are invariant under a 90° rotation. The diagonal domain depicted in Figure 2.2g is considered to determine whether it is regarded as a similar mixing state or slightly more mixed due to the higher surface area between the two classes.
- The two scenarios of three layers depicted in Figures 2.2c and 2.2d are considered to determine

whether the mixing indexes are invariant under a 90° rotation. The Circle in Figure 2.2h is expected to have a slightly higher mixed state as the surface area between the two classes is higher. Note the spatial concentration variation of the same particle type. For the three layers the spatial concentration of the larger group of particle types is at the edges, while for the circle it is in the middle resulting in a larger surface area between the two particles types. Lastly, all three scenarios should reflect a higher mixed state than the two-layer and diagonal scenarios.

For example, the contact surface area between the segregation layers of the two-row or two-column in these cases are $100mm$ each. Likewise, the contact surface area in the triangle case is $\sim 141mm$ and in the case of the circle, the contact surface area is $\sim 251mm$. Additionally, the contact surface area between different particle types in the three-row and three-column examples are $200mm$ respectively.

- The cases illustrated in Figures 2.2c and 2.2d can be seen as one of the first steps in mixing, where one layer of particles intersect the other. In this case, there should be some evidence of mixing in each of the methods, but far from being fully developed.
- The cases in Figures 2.2e and 2.2f follow a similar logic to the cases in the above-mentioned point but show an even further developed structure. Although it seems that these cases are more mixed, one can still identify segregated patterns in the domain.
- The case scenario depicted in Figure 2.2i is a pathological scenario designed to show a more extensive mixed state. However, there is one distinct neighbourhood of red particles running through the domain.

We anticipate some of the displacement dependant methods, such as the Neighbour distance and Siiria methods to not perform well at all. This is due to the reliance on progressive mixing happening throughout the cycle, where the initial state of a sample is assumed to be segregated and the final state to be mixed. By switching the functions around the result for these methods remains the same.

When considering the AHM in the purest form the mixing index is calculated directionally (Section 1.4.2.1), and by rotating the domain 90°, the particle centroids in the y-direction will start to align, and it is anticipated that this will cause the mixing index to increase without any mixing occurring in the domain.

In sampling methods, a random number generator was used to randomly choose 50 particles as the sample ($\approx 10\%$ of the domain). According to Figure 1.15, the results in a mixing variance of $\sigma_M^2 \approx 0.004$. The list of 50 particles is then reused as the sample for every mixing index, ensuring a uniform comparison across sampling methods.

The cell-based methods have been determined using a 4×4 grid, with each cell containing $\approx 6.25\%$ of the total amount of particles in the volume. The grid is visible in each case in Figure 2.2.

As an additional measure to calculate accurate results the mixing index for each case in this study is measured as an average MI across 100 random samples, the variance of which is also included to serve as a confidence measure in the data provided. This is to ensure that adequate sampling takes place for MI which requires random samples to calculate the MI thus, the initial state t_0 and final state t_n remains constant.

2.1 Sampling methods

In Section 1.4.1, it was discussed that the sampling methods are methods that are designed to be segregation determining methods for the most part.

The methods discussed in this section is expected to show a more developed mixed state for cases that have a larger contact surface area. This will be evident when looking at the cases in Figures 2.2e, 2.2f, 2.2h and 2.2i. Similarly, in the cases depicted in Figures 2.2a, 2.2b and 2.2g the mixing state is expected to be lower.

Likewise, due to the approach that the sampling methods calculate mixing the case in Figure 2.2i can be expected to show near-perfect mixing results. This is due to the particle contact surfaces area being large.

2.1.1 Nearest neighbours

Considering that the NN method is highly dependent on the sample particles and their neighbours, it is important to understand where the sample particles are located. If the sample particles are all in one neighbourhood, the mixing index will return a low value, and when all the particles are located close to the border of the neighbourhood the NN will return a high value.

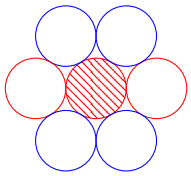
It is thus important to take various samples across the domain to ensure accurate measurements.

In Table 2.1 the NN method was applied to cases in Figure 2.2. As expected, the cases in Figures 2.2e, 2.2f, 2.2h and 2.2i returned a higher MI value than the cases in Figures 2.2a, 2.2b and 2.2g.

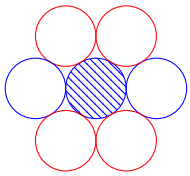
Table 2.1: Nearest Neighbours performance

Segregation pattern	**MI**	**Var.** $[\sigma^2]$
Two-row (Fig. 2.2a)	0.06	3.18E-04
Two-column (Fig. 2.2b)	0.07	4.97E-04
Three-row (Fig. 2.2c)	0.11	7.71E-04
Three-column (Fig. 2.2d)	0.13	7.10E-04
Two-block (Fig. 2.2e)	0.12	9.16E-04
Three-block (Fig. 2.2f)	0.23	1.49E-03
Diagonal (Fig. 2.2g)	0.09	2.10E-03
Circle (Fig. 2.2h)	0.15	9.76E-04
Teeth (Fig. 2.2i)	1.31	1.86E-04

The case in Figure 2.2i returned a high mixing value as expected and this is due to the number of different neighbouring particles for most samples being $n_{(i)diff} = 4$, as illustrated in Figure 2.3, resulting in a MI for both cases of $M_{NN} = 1.33$. The edge cases in the example bring the overall mixing index down to $M_{NN} = 1.31$ as seen in Table 2.1



(a) Red sample particle



(b) Blue sample particle

Figure 2.3: Teeth samples - $n_C = 6$

Some observations can be derived from the results of the NN method:

- The samples displayed in Figures 2.2a, 2.2b, 2.2g and 2.2h, are bi-segregated and produce a different mixed state for each. Thus as expected the method is dependent on the contact surface area between particle neighbourhoods. The surface contact area increases the probability of a sample particle on the perimeter of the segregation layers, causing a mixing concentration on the boundary layer. The more sample particles are taken on the layer, the higher the MI. Figure 2.4 illustrates this phenomenon.

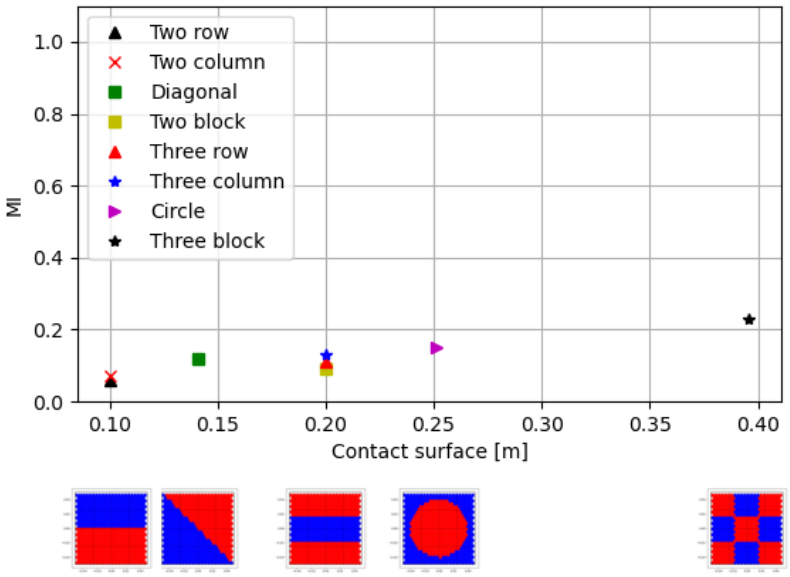


Figure 2.4: NN surface area comparison

- The method assumes having half the amount of neighbouring particles as different particle types as perfectly mixed ($M_{NN} = 1$); however, as can be seen in the teeth example in Figure 2.2i, this is not necessarily the case where more than half of the nearest neighbouring particles were from a different particle type, causing a hyper-mixed state where ($M_{NN} > 1$).

  The method thus also assumes that for perfect mixing ($M_{NN} = 1$) to be quantified there should be equal amounts of each particle type in the domain, and if there are unequal amounts of each particle type in the domain the final mixed result will most probably always be a hyper-mixed state.

  There are methods such as the PSI, Lacey or ME methods that mathematically account for the unequal amounts of particles in each domain.

2.1.2 Coordination Number

For the CN method the same hypothesis can be followed as for the NN method. This method is closely related to one another due to the dependencies on the nearest neighbouring particles and the number of different particles in that region.

For all the cases depicted in Figure 2.2, the particle domains are so closely packed that the NN and CN neighbours are the same except for particles on the perimeter of the domain, as shown in Figure 1.14.

Consider Table 2.2 where the values closely relate to the values in Table 2.1.

Table 2.2: Coordination Number performance

Segregation pattern	**MI**	**Var.** $[\sigma^2]$
Two-row (Fig. 2.2a)	0.06	3.73E-04
Two-column (Fig. 2.2b)	0.06	4.41E-04
Three-row (Fig. 2.2c)	0.14	9.67E-04
Three-column (Fig. 2.2d)	0.11	5.65E-04
Two-block (Fig. 2.2e)	0.12	8.16E-04
Three-block (Fig. 2.2f)	0.24	1.38E-03
Diagonal (Fig. 2.2g)	0.09	6.69E-04
Circle (Fig. 2.2h)	0.18	1.19E-03
Teeth (Fig. 2.2i)	1.22	3.62E-04

Again, in Figure 2.5 the dependency on the contact surfaces is illustrated.

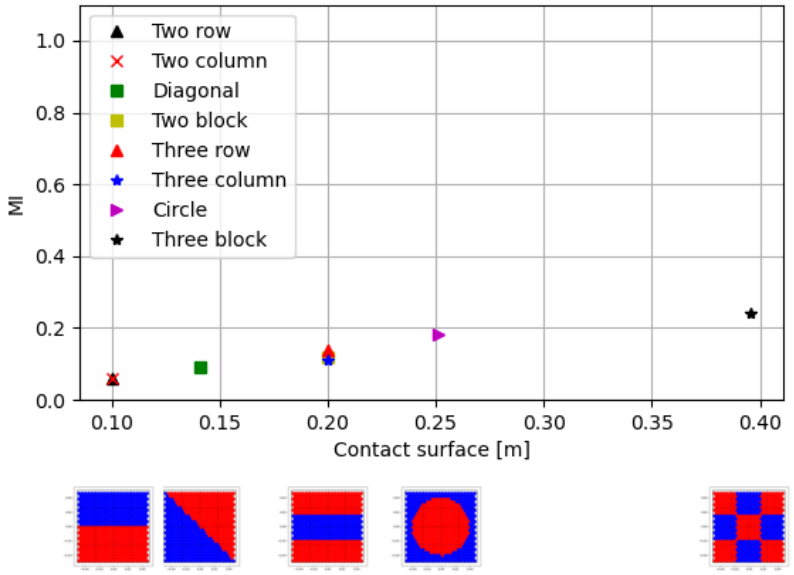


Figure 2.5: CN surface area comparison

An interesting behaviour to note is that of the variance σ^2 in both the NN and CN methods. As the contact surface areas increase, the variance in MI also increases - this is due to the number of particles that are sampled on and off the boundary of each neighbourhood. Although the confidence interval is still acceptable, the concern is that if the contact surface area increases, the variance will also increase, and possibly cause some inaccuracy in each reading.

2.1.3 Particle Scale Index

The PSI method combines some qualities from a method such as the NN or CN method and that of the Lacey method, which accounts for unequal amounts of particle types. Thus the PSI method can more

accurately represent results for cases such as the Figure 2.2c, 2.2d, 2.2f and 2.2h examples, all of which have more red particles than blue particles.

Looking at the results in Table 2.3, the mixing indexes provide similar results as in the NN and CN methods; however, one can expect more accurate results in the patterns which have unequal amounts of particles. Figure 2.6 shows the relationship between the surface area and the MI.

Table 2.3: Particle Scale Index performance

Segregation pattern	**MI**	**Var.** $[\sigma^2]$
Two-row (Fig. 2.2a)	0.35	5.10E-04
Two-column (Fig. 2.2b)	0.35	6.17E-04
Three-row (Fig. 2.2c)	0.41	2.50E-03
Three-column (Fig. 2.2d)	0.42	2.39E-03
Two-block (Fig. 2.2e)	0.41	9.58E-04
Three-block (Fig. 2.2f)	0.51	1.35E-03
Diagonal (Fig. 2.2g)	0.37	6.56E-04
Circle (Fig. 2.2h)	0.42	1.04E-03
Teeth (Fig. 2.2i)	1.13	1.11E-05

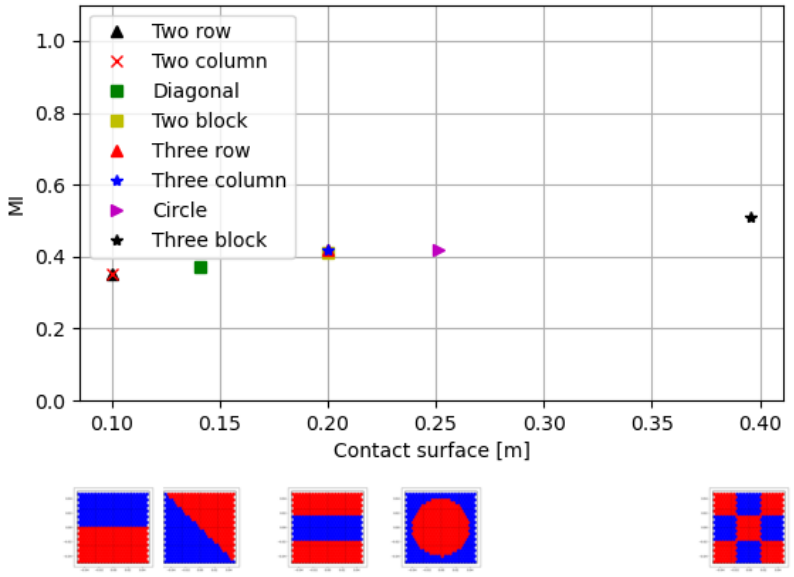


Figure 2.6: PSI surface area comparison

The PSI method's results are however difficult to compare with other MI's due to the fact that the lower boundary of the MI seems to be in the region of $M_{PSI} \approx 0.35$. This is due to the algorithm setup. Consider the case in Figure 2.7 as a segregated structure.

If in both of these cases the sample particles are inspected $p_i = [\frac{0}{7}, \frac{6}{7}]$, where $p_t = \frac{1}{2}(\frac{6}{7})$ for $N' = 2$, where the following additional parameters are $\sigma^2 = \frac{9}{49}$, $\sigma_0^2 = \frac{1}{4}$ where the particle fraction $p_0 = 0.5$ and $\sigma_S^2 = \frac{1}{28}$ the MI for the first segregated case can be calculated as:

(a) Extreme case 1 (b) Extreme case 2

Figure 2.7: PSI Segregated case

$$M_{PSI} = \frac{\sigma^2 - \sigma_0^2}{\sigma_S^2 - \sigma_0^2} = \frac{\frac{9}{49} - \frac{1}{4}}{\frac{1}{28} - \frac{1}{4}} = \frac{13}{42} \approx 0.3095 \tag{2.1}$$

In contrast, the sample from Figure 2.8 for a mixed case, results in; $p_t = \frac{3}{7}$ and $\sigma_t^2 = \frac{1}{49}$ which gives us $M = \frac{15}{14} \approx 1.07$

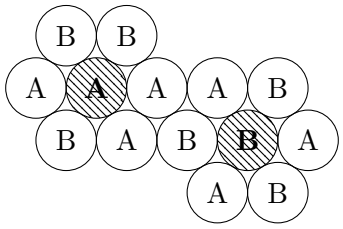


Figure 2.8: PSI Mixed case

Figure 2.9a shows the behaviour as described. For this figure, the p_i variable swept between 0 and 1 with added binomial random noise to simulate variance in p_i. This was done by simulating the number of different particles with $0 \geq \mu \geq 1$.

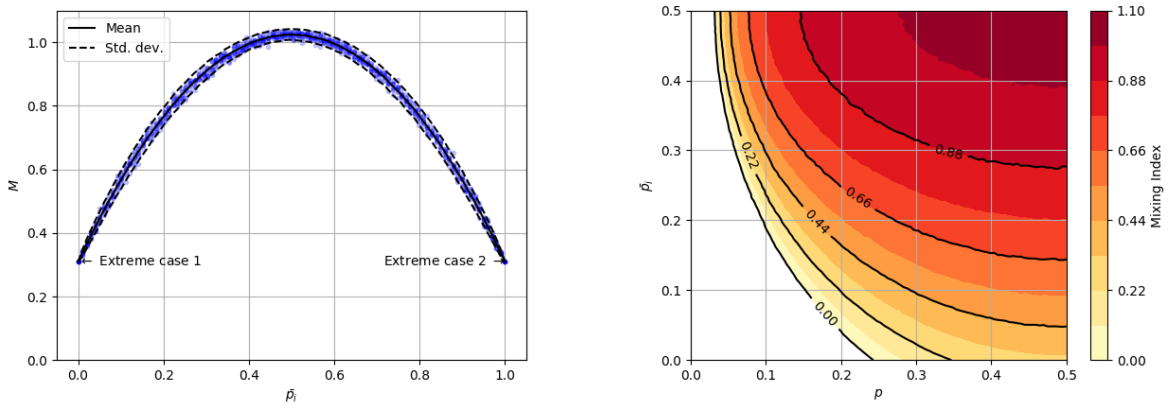
$$n_{(i)diff} = n_{(i)C} - \binom{n_C}{N'} \mu^{N'} (1-\mu)^{(n_C - N')} \tag{2.2}$$

One can see that at the two extreme cases, as shown in Figure 2.9a, where $\bar{p_i} = 0$ and $\bar{p_i} = 1$ the mixing index is $M_{PSI} \approx 0.31$ and where $\bar{p_i} = 0.5$ the mixing index is $M_{PSI} \approx 1.07$.

As discussed in the sampling methods before, sampling on the neighbourhood boundary may result in skewed results. However, this can be countered by looking at different sampling methods such as the Latin Hypercube sampling method, or ensuring a large enough sample is taken to ensure an even distribution of random samples throughout the domain.

Finally, consider this behaviour on a two-axis sweep for p and $\bar{p_i}$. Forming a parabolic contour, as seen in Figure 2.9b, the function is symmetric on both $p = 0.5$ and $\bar{p_i} = 0.5$.

As seen in this plot, it is illustrated how the parameters influence the MI. If a segregated sample is taken, where particle type A has a particle fraction of $p_A = 0.2$. The average localized particle fraction has to be $\bar{p_i} > 0.035$, anything less than this is unlikely. This diagram can serve as a quick sanity check when implementing the PSI method.



(a) $\bar{p_i}$ sweep

(b) p and $\bar{p_i}$ sweep

Figure 2.9: PSI particle fraction relationships

2.2 Displacement dependant methods

These methods are not expected to perform well under rotational invariances and neither do they perform exceptionally well in identifying segregation zones. However, the displacement dependant methods can be used to show the mixing intensity or how thorough the mixture is evolving.

These methods come in handy to see whether the current mixing method is adequate to agitate the mixture enough and usually performs better in mixtures where the particles do not stay in contact, i.e. - suspended mixtures like aerated or fluidized particle beds.

2.2.1 Average Height Method

The Average Height method was developed to investigate the mixed state of mainly fluidized particle beds. It is a method that is easily evaluated with minimal computational power.

As discussed in Section 1.4.2.1 the AHM MI is directionally sensitive. By measuring the centroid in one direction only, the results in Table 2.4 show some interesting results. For all cases below the results are calculated in the y-direction. The method aptly describes the segregation pattern for the two-row case in Figure 2.2a as unmixed.

The diagonal case in Figure 2.2g is determined as $M_{AHM} = 0.35$. This is because roughly 33% of the domain's particle's overlap in the y-direction, shifting the centroids closer to each other by 33%. Due to the method not considering neighbouring particles the contact surface area does not influence the MI. Likewise, the remainder of the cases in Figure 2.2 shows similar behaviour, where the particle centroids overlap in one direction and not the other.

Some of the other pitfalls of this method, it should be noted that if the particle types should swap around in Figure 2.2a the average height of $\bar{x}_{blue} \approx 0.025$ and $\bar{x}_{red} \approx 0.075$ then $X_A \approx 1.5$ and then inflates the mixing degree to $M \approx 2$. The method is sensitive to the chosen reference frame in which to compute the height.

Table 2.4: Average Height performance

Segregation pattern	**MI**
Two-row (Fig. 2.2a)	0.05
Two-column (Fig. 2.2b)	1.00
Three-row (Fig. 2.2c)	1.00
Three-column (Fig. 2.2d)	1.00
Two-block (Fig. 2.2e)	0.98
Three-block (Fig. 2.2f)	1.00
Diagonal (Fig. 2.2g)	0.35
Circle (Fig. 2.2h)	0.95
Teeth (Fig. 2.2i)	1.00

2.2.2 Neighbour distance and Siiria

When considering the ND and Siiria methods, in this case, one does not anticipate that the MI will be indicative of any segregation identification. These methods are inherently designed to either compare the current mixed state to a randomized state or the movement of each particle in the domain. These methods are also particle type agnostic and this creates the problem in itself. Table 2.5 references the results for the nearest distance and Siiria methods, and as mentioned all the outcomes are fully mixed.

Table 2.5: Neighbour distance and Siiria performance

Segregation	**Neig. dis.** $[M]$	**Siiria** $[M]$
Two-row (Fig. 2.2a)	1.01	1.00
Two-column (Fig. 2.2b)	1.01	1.00
Three-row (Fig. 2.2c)	1.01	1.00
Three-column (Fig. 2.2d)	1.00	1.00
Two-block (Fig. 2.2e)	1.00	1.00
Three-block (Fig. 2.2f)	1.01	1.00
Diagonal (Fig. 2.2g)	1.00	1.00
Circle (Fig. 2.2h)	1.01	1.00
Teeth (Fig. 2.2i)	1.00	1.02

The ND and Siiria based methods are however MI that can be used to calculate the mixing performance regardless of a particle type. Due to this experiment in identifying segregation patterns, these methods fail.

2.3 Cell dependant methods

Cell dependant methods have been one of the most trusted ways of measuring mixing performance through the years. As mentioned before it is simple to implement in both numerical and experimental environments. It uses a statistical viewpoint to analyse the domain similar to the PSI method.

The cell dependant methods are in general robust methods to calculate mixing in which one can expect a MI which can identify segregation patterns and calculate the mixing thoroughness throughout the domain. This is thanks to how the method is developed to take into account particle types and particle placements

across the domain.

Similar to all methods described thus far there are some pitfalls to take into account. The first is how the cells are set up. There are many configurations in the that can be set up to most accurately describe the domain, whether it be in a two-dimensional lattice with triangular or quadrilateral cells or three dimensional with any of the various elements available. For this study, a quadrilateral cell structure will be used for the ease of setting up and the accurate representation of the simple mixing volumes used in this study. It is important however to choose a cell size such that there are enough particles in each cell to calculate the mixing, but small enough to capture an adequate resolution of mixing occurring in the domain. As a rule of thumb, a cell should not be large enough to capture at least one coordination number n_C worth of particles so that p_c can be calculated accurately.

Secondly, the placement of the cell boundaries is important. If the cell boundary falls on a neighbourhood boundary layer, the mixing index will be calculated based on the cells on either side of the boundary layer, causing an inaccurate representation of the MI.

2.3.1 Lacey

In Table 2.6, the MI values are shown for each case in Figure 2.2, as before. Due to the cell-dependent methods and taking into account each particle when calculating the MI there is no variance in the data, i.e. $\sigma^2 = 0$ for all cases. As expected the method can perform well in describing the mixing in the domain.

Table 2.6: Lacey performance

Segregation pattern	**MI**
Two-row (Fig. 2.2a)	0.07
Two-column (Fig. 2.2b)	0.07
Three-row (Fig. 2.2c)	0.46
Three-column (Fig. 2.2d)	0.46
Two-block (Fig. 2.2e)	0.07
Three-block (Fig. 2.2f)	0.68
Diagonal (Fig. 2.2g)	0.20
Circle (Fig. 2.2h)	0.52
Teeth (Fig. 2.2i)	1.02

Looking at the contact surface dependency in Figure 2.10 the dependency is similar to the previous cases where there is a direct correlation between the MI and contact surface area.

Interestingly if we inspect the cases in Figure 2.2a, 2.2b, 2.2g and 2.2h, apart from the contact surface area being larger on the diagonal and on the circle perimeter, the cases are very similar - equal amounts of particles bi-segregated. However, in the Lacey method, the diagonal case is calculated at $M_{Lacey} = 0.20$ as opposed to a neighbouring method such as NN, where $M_{NN} = 0.09$. Likewise, in the circle case where $M_{Lacey} = 0.52$ and $M_{NN} = 0.15$. The reason why the MI in the Lacey method is inflated is in the manner in which the cells are being set up.

To explore this behaviour a simple two-row segregation pattern is used to demonstrate the on and off boundary cells as illustrated in Figures 2.11a and 2.11b. In Figure 2.12a, one can see that even though the number of cells is increased in the domain the mixing index is still dependent on the cell boundary.

Another observation regarding this is the MI calculated for the two-blocks case in Figure 2.2e. The MI is calculated at $M_{Lacey} = 0.07$ similar to the two-row and column scenarios. Although the contact surface in

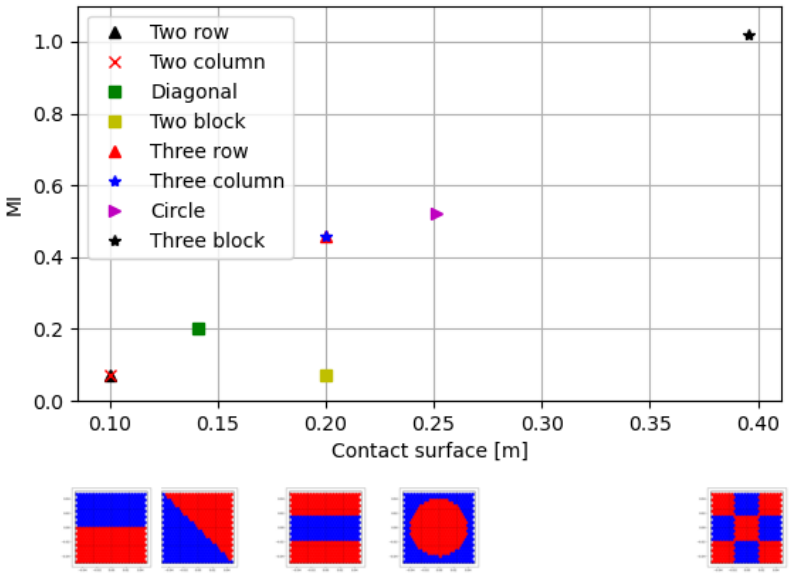


Figure 2.10: Lacey surface area comparison

the two-blocks case is double that of the two-row and column cases.

(a) On neighbourhood boundary

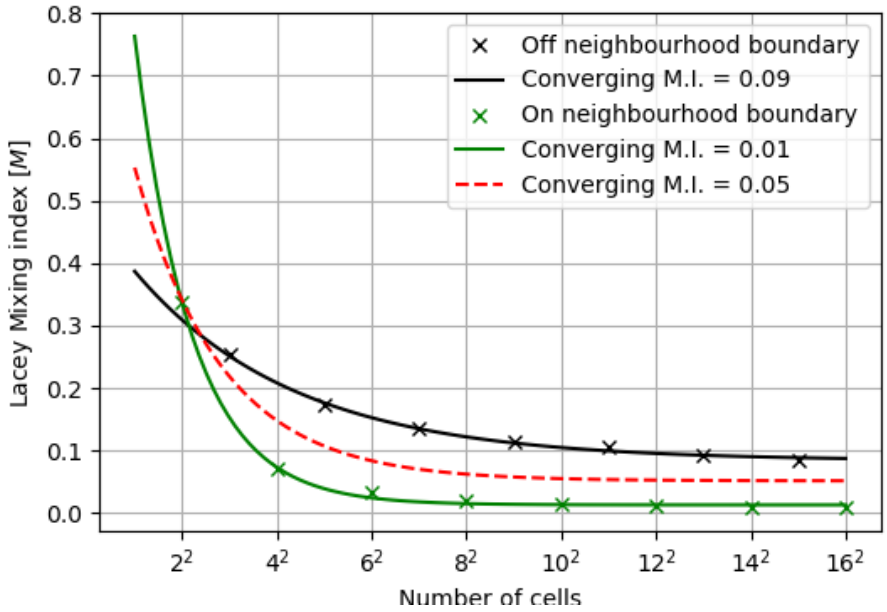
(b) Off neighbourhood boundary

Figure 2.11: On and off boundary cells

Thus it is important to take care in choosing the correct lattice structure and size when measuring cell-based MI.

As discussed above, the diagonal case is subject to the on and off boundary cells and in this case up to 25% of the cells show a $p_c = 0.5$ (fully mixed) and the remainder of the cells will produce a $p_c = 0$ or $p_c = 1$ (unmixed). One would assume this can be counteracted by increasing the number of cells in the domain increasing the resolution around the boundary layers.

However, one of the problems with increasing the number of cells in the domain is that the cell size decreases along with it to a point where one can only fit one particle per cell - causing the method to fail.



(a) Two-row segregation Lacey cell saturation

Figure 2.12: Lacey - cell saturation level

2.3.2 Mixing Entropy

Similar to the results in Wen, Liu, Liu, et al., 2015, the ME method provides values that are rounded and more elegant than those of the Lacey method. This is due to the forgiving approach taken with the energy equations used in this method. Similarly, Figure 2.9a E has this parabolic shape which the method uses as a proxy to calculate the MI. By moving up or down this parabolic curve one can calculate the maximum energy in the system, E_{mix} - this ensures the MI is always $0 \leq M_{ME} \leq 1$.

Additionally, the ME takes into account cells that are less filled, which may skew the results in other methods. These adaptations make this method the more robust MI to consider.

Unfortunately, the ME still inherits the same issues caused by selecting a cell lattice as described in Section 2.3.1. This is evident in the results in Table 2.7 and Figure 2.13 - the two-block and diagonal cases show this.

As previously mentioned in Section 1.4.3, cell-based can be used to map the domain and understand where the mixing hotspots occur. These methods are easier to implement experimentally than some of the displacement and sampling-based methods to understand the transient mixing states. Figure 2.15 it shows each of the cases discussed in this chapter heat-mapped based using the ME method and a higher cell lattice resolution (8×8).

Once again observing the cases' two-row (Figure 2.14a), two-column (Figure 2.14b) and two-block (Figure 2.15b), the cell boundary is aligning with the contact surfaces. In these cases, no mixing is displayed across the domain, causing the results as discussed in Table 2.7.

In cases where the contact surface boundaries run straight through the cells, as shown in the three-row (Figure 2.14c), three-column (Figure 2.15a), three-block (Figure 2.15c), diagonal (Figure 2.15d) and circle

Table 2.7: Mixing Entropy performance

Segregation pattern	**MI**
Two row (Fig. 2.2a)	0.00
Two-column (Fig. 2.2b)	0.00
Three-row (Fig. 2.2c)	0.49
Three-column (Fig. 2.2d)	0.49
Two-block (Fig. 2.2e)	0.00
Three-block (Fig. 2.2f)	0.71
Diagonal (Fig. 2.2g)	0.25
Circle (Fig. 2.2h)	0.55
Teeth (Fig. 2.2i)	0.99

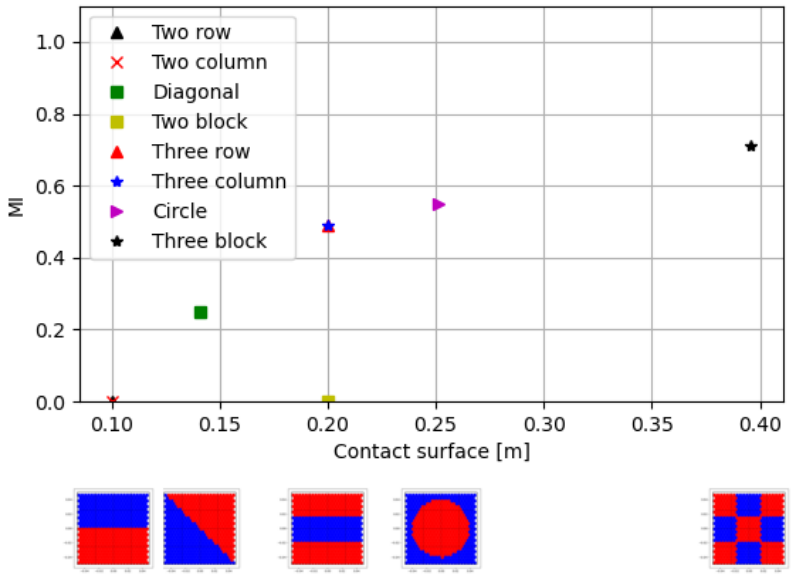
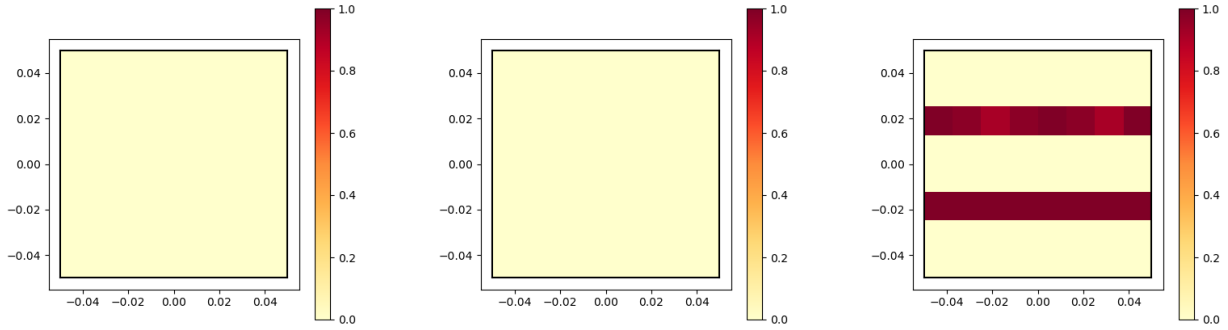


Figure 2.13: ME surface area comparison

(Figure 2.15e) cases, the heat mapping displays where the mixing occurs in the domain.

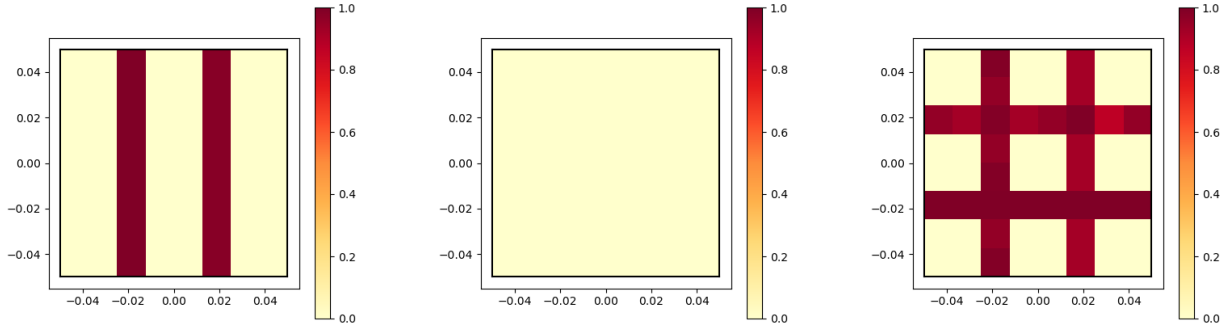
2.4 Neighbourhood methods

The Neighbourhood method was developed in particular to identify the segregation patterns that may be developing in the domain. The methods are robust and reliable methods to use in circumstances where small numbers of particles are used. There is however some drawback to these methods, such as longer computational times and particle packing structures that can cause long chains of neighbouring particles which is well mixed.

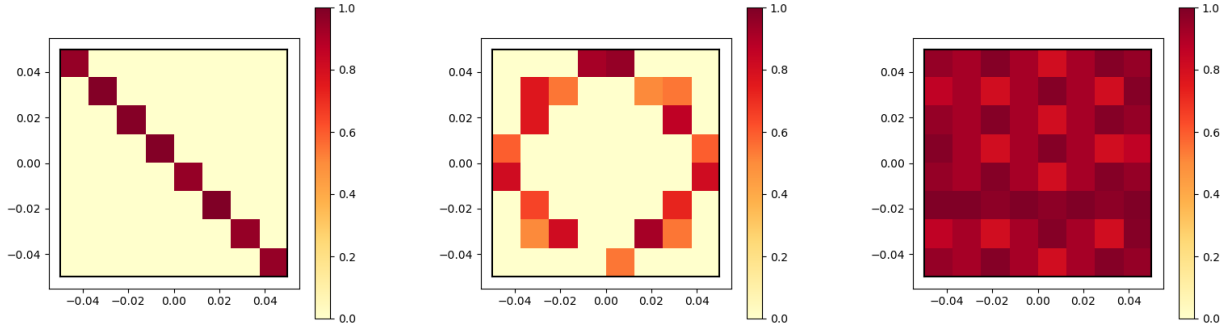


(a) Two-row (b) Two-column (c) Three-row

Figure 2.14: Hypothetical mixed cases - heat mapping



(a) Three-column (b) Two-block (c) Three-block



(d) Diagonal (e) Circle (f) Teeth

Figure 2.15: Hypothetical mixed cases - heat mapping (continued)

2.4.1 Chain Length

The chain length method shows some progress to illustrate and identify the segregation boundaries in the domain. There, however, are some intricacies to understand in the workings of this algorithm. Because of the natural packing of the particles, most sample particles are bound to have a neighbouring particle, as it

should for a mixed volume; this causes some chains to terminate earlier than others and some samples to find longer routes through the domain. This causes the chain lengths to vary quite a bit and thus increase the variance on η_i. Although not included in this study exploring the variances of η_i (similar to the Particle Scale Index, Lacey and Mixing Entropy methods) might be promising to quantify mixing.

Table 2.8: Chain Length performance

Segregation pattern	**MI**	**Var.** $[\sigma^2]$
Two row (Fig. 2.2a)	0.06	3.70E-03
Two col. (Fig. 2.2b)	0.06	2.88E-03
Three row (Fig. 2.2c)	0.37	1.13E-03
Three col. (Fig. 2.2d)	0.39	3.15E-03
Two block (Fig. 2.2e)	0.42	5.78E-03
Three block (Fig. 2.2f)	0.69	1.93E-03
Diagonal (Fig. 2.2g)	0.09	1.24E-02
Circle (Fig. 2.2h)	0.06	6.38E-03
Teeth (Fig. 2.2i)	0.88	1.44E-04

In Table 2.8 the results are shown for each segregation pattern as before. One can however see that the method is successful in determining segregation patterns in the domain with cases like the diagonal, circle and two-row and column cases. In Figure 2.16, the relationship between contact surface area and MI is not as evident as cases like the circle and diagonal methods produce, a lower value for M_{CL}. The long red chain in the teeth case as shown in Figure 2.2i is calculated as one long neighbourhood, thus lowering the MI for M_{CL} as opposed to other MI in this study calculating it as perfectly mixed ($M \approx 1$).

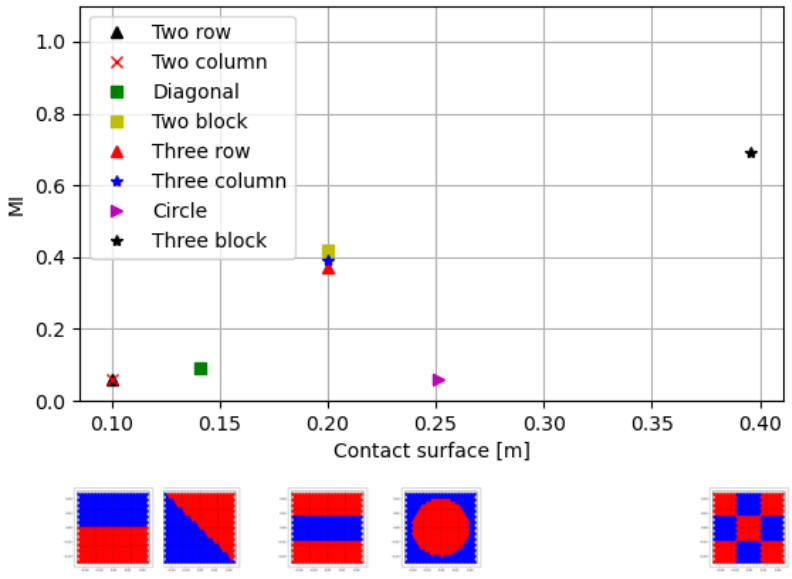


Figure 2.16: CL surface area comparison

The chain length method relies on a random sample of initial points from which to start the chains from. This gives us a chance to look at the variance for each case. If the domain has neighbourhoods with sharp

corners, such as the diagonal method, the chains have a higher likelihood of terminating in each of the corners, causing some chains to be short and others to be longer. This increases the variance in the sample and thus lowering the confidence in the values provided. On the other hand, if the boundary is smooth or straight as in the two-row or column examples the variance is lower and increases the confidence in the values provided. This proves that for the MI to be quantified within reasonable accuracy a larger sample size is needed to calculate more accurate results.

Once again a remark can be made with regards to sampling methods - by using Latin Hypercube sampling the user might have better results before the chain terminates itself in a nook or cranny. As for examples in this study using a brute force approach and sampling well enough throughout the domain, will suffice for the results needed.

2.4.2 Growth Point

The GP method, similar to the CL method, is a new method proposed in this study in the effort to identify neighbourhoods and segregation patterns in the domain. In the manner in which the GP method identifies neighbourhoods, the expectation is for the method to not be able to quantify perfect mixing ($M_{GP} \neq 1$), nor will the method be dependent on the contact surface area.

Looking at the results in Table 2.9 the GP method adequately identifies the domains with a clear bi-segregation layer. These cases are the two-row and column, diagonal and circle cases, all with $M_{GP} = 0$. In the remaining cases where there are multiple neighbourhoods, the method identifies them and assigns a value to each. As expected, none of the values is dependent on the contact surface area as seen in Figure 2.17.

Considering the bi-segregation cases, we can see that the samples show no variance in the MI. This is due to the algorithm identifying a neighbourhood in every iteration, and if there are only two neighbourhoods in a domain the mixing index will be ($M_{GP} = 0$).

Table 2.9: Growth Point performance

Segregation pattern	**MI**	**Var.** $[\sigma^2]$
Two row (Fig. 2.2a)	0.00	-
Two col. (Fig. 2.2b)	0.00	-
Three row (Fig. 2.2c)	0.32	4.97E-04
Three col. (Fig. 2.2d)	0.33	4.95E-04
Two block (Fig. 2.2e)	0.25	6.80E-04
Three block (Fig. 2.2f)	0.34	1.41E-03
Diagonal (Fig. 2.2g)	0.00	-
Circle (Fig. 2.2h)	0.00	-
Teeth (Fig. 2.2i)	0.47	2.51E-03

On closer inspection for each of the cases in Figure 2.2 the following number of neighbourhoods are identified by the GP method and tabled in Table 2.10. This is due to the particle packing on the borders of the segregation layers.

Up to this point, both the Chain Length and Growth Point methods have been a proof of concept. The GP method performs well on paper and brings a new perspective to mixing indexes. However, this method solves some of the issues like contact surface dependencies and segregation identification - the method, unfortunately, is computationally expensive. In Figure 2.18, we can time each of the methods that it

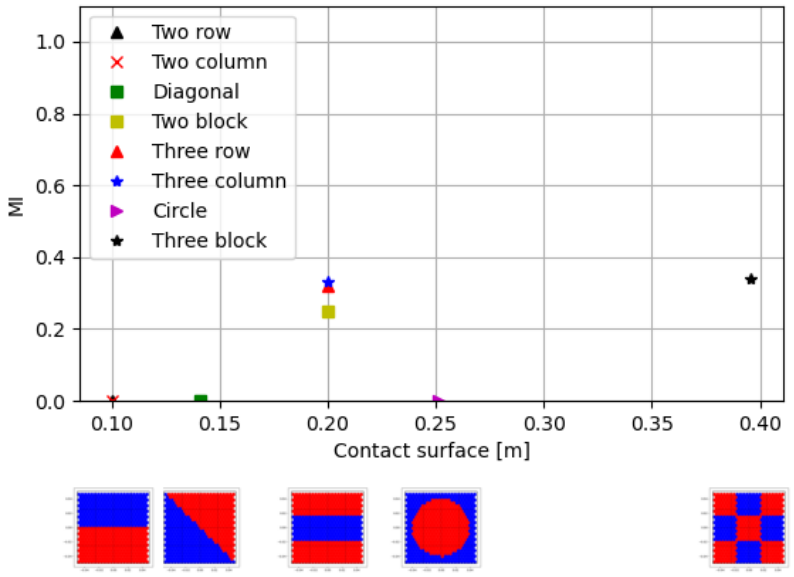


Figure 2.17: GP surface area comparison

Table 2.10: Growth Point neighbourhoods

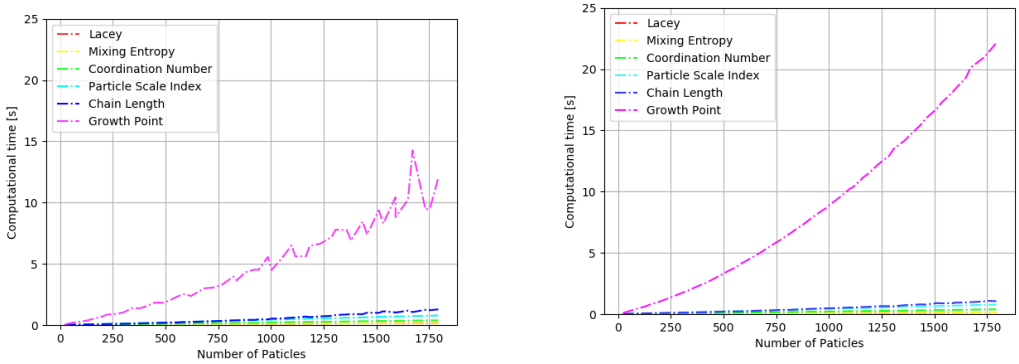
Segregation pattern	**Neighbourhoods**
Two row (Fig. 2.2a)	2
Two col. (Fig. 2.2b)	2
Three row (Fig. 2.2c)	3
Three col. (Fig. 2.2d)	3
Two block (Fig. 2.2e)	3
Three block (Fig. 2.2f)	5
Diagonal (Fig. 2.2g)	2
Circle (Fig. 2.2h)	2
Teeth (Fig. 2.2i)	14

takes to compute for both a mixed and segregated pattern. The mixed state in Figure 2.18a, in general, takes slightly quicker to compute numerically than the segregated pattern in Figure 2.18b. This is due to the mixed pattern neighbourhoods being smaller neighbourhoods which will terminate quicker. Adding a third dimension for a three-dimensional case the neighbourhood can potentially grow even larger quicker, slowing down the algorithm even more.

With this method, one can gain some valuable insights into how domains can be arranged so that it is optimally mixed. This GP method is set up in such a way that if the domain is filled with a checker-board style of particles, the MI will return a perfectly mixed value $M_{GP} = 1 - 1/(N_t N_l)$. This assumes that the average neighbourhood size is equal to one particle $n_t = 1$.

In the case of a simple cubic spherical packing as illustrated in Figure 2.19a a perfectly mixed value can be achieved as no neighbouring particles are of the same type. However, in the case of face-centred cubic packing, as illustrated in Figure 2.19b, there are some neighbouring particles of the same particle type.

11111



(a) Teeth case

(b) Diagonal case

Figure 2.18: Computational time for a segregated case vs. a mixed case

(a) Simple cubic

(b) Face-centred cubic

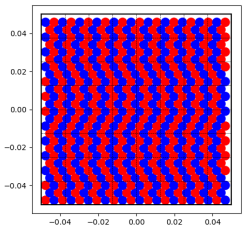
Figure 2.19: Spherical packing

In the cases discussed in this section, all are centre faced cubic. The ideal mixed case can be simulated by increasing the number of rows or blocks in the segregation patterns. In Figure 2.20d, this is illustrated by increasing the number of blocks. When doing this the MI is inconsistent. This happens when the number of blocks starts including the neighbouring blocks, as can be seen in Figure 2.2f, where the red neighbourhood is deemed as one neighbourhood. The saturation point for this specific case is at 22×22 blocks, as illustrated in Figure 2.20a. At this point, the blocks merge into vertical columns, similar to Figure 2.19b.

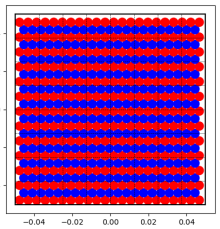
Figure 2.20e illustrates the same principle as above by increasing the number of rows in the pattern. Producing a more consistent MI, the domain allows for a total of 26 rows in the domain as can be seen in Figure 2.20b. In both cases, the maximum MI outperforms the results for a random mixing sample at $M_{GP} = 0.91$ for the case of the blocks and $M_{GP} = 0.92$ for the case of the row.

As discussed previously, the algorithm has an upper bound for each situation different for each initial value, particle amounts, and packing order. This is very difficult to quantify and predict - which leads to the next point. The method at this stage is a proof of concept. The method is computationally expensive and has some inconsistencies for the final mixed state. One thing to consider is that there is no match for the ability to identify boundaries and neighbourhoods. The suggestion is to either compute this mixing index when there is a discrepancy in the final calculation or calculate the mixing index less frequently to save up on computational costs.

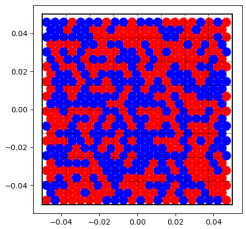
This method can be used in conjunction with other methods, showing mixing performance in the domain.



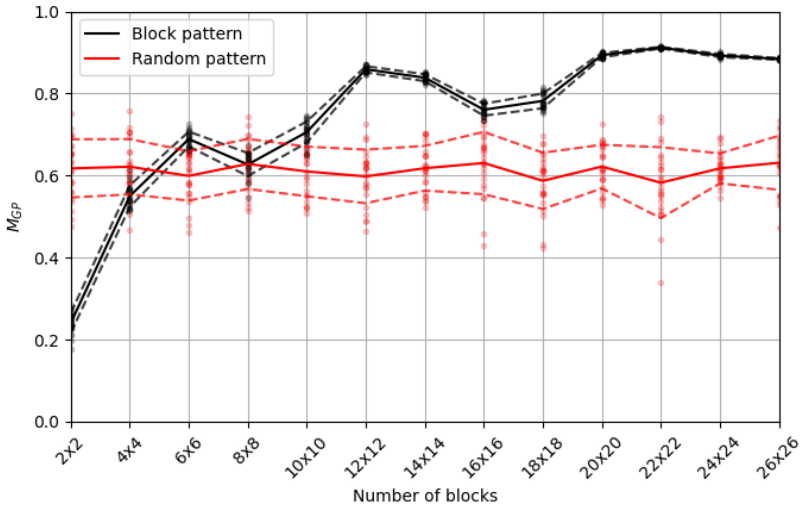
(a) 22×22 case



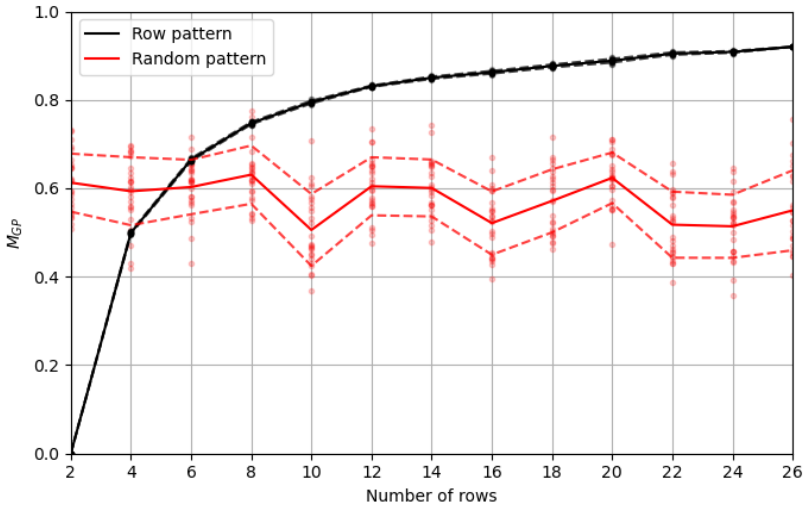
(b) 26 rows case



(c) Random transform



(d) Block case pattern growth



(e) Row case pattern growth

Figure 2.20: Growth Point mixing evolution

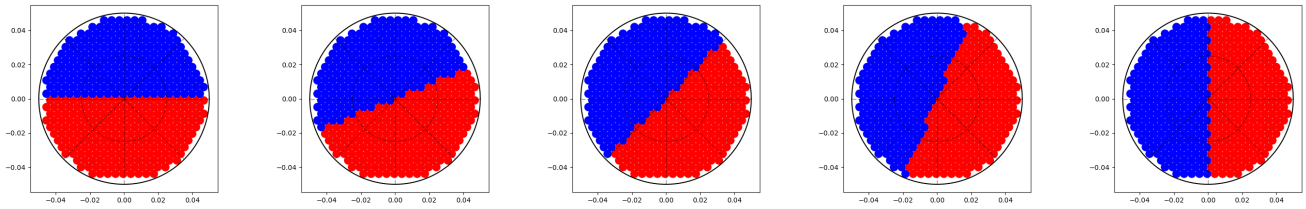
This will provide a holistic view of the mixing in a domain.

Due to processing capabilities and inconsistent results this method will be used minimally in this study going forward.

2.5 Domain rotation

In the previous section the diagonal case has shown some interesting results and it was due to the increased contact surface area between the neighbourhoods. In essence, the diagonal case scenario is a 45° rotation around the centroid of the domain. Here one considers domain rotation at intermediate angles such that the surface area remains unchanged. This is done by using a circular domain rather than a square one.

Illustrated in Figure 2.21 are examples of a 90° rotation.



(a) 0° rotation (b) 22.5° rotation (c) 45° rotation (d) 67.5° rotation (e) 90° rotation

Figure 2.21: Domain rotations

Figure 2.22 illustrates that some indexes are sensitive to rotation, in particular - the cell-based methods and the AHM. To unpack this further it must be understood that the reference point for the AHM is taken from the bottom of the circumference volume and the cells stay constant throughout the rotation.

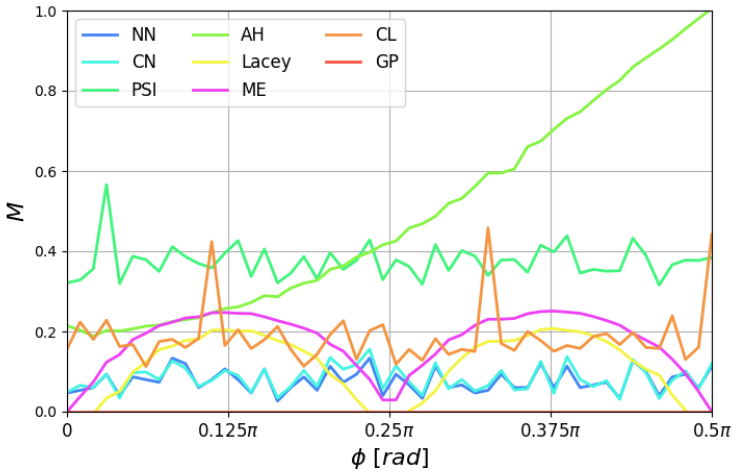


Figure 2.22: Domain Rotation performance[1]

[a]GP method produces a $M_{GP} = 0$

AHM performs as expected. The MI increases as the centroids align. A similar result can be achieved by calculating the centroids of two semi-circles, scaled to the degree of rotation as in equation 2.3.

$$M_{AHM} = 1 - \frac{8}{3\pi}\cos(\theta) \tag{2.3}$$

The Lacey method fluctuates between $0 \leq M \leq 0.213$. As calculated earlier in the chapter, this fluctuation

shows how the segregation boundary moves over the four cells it crosses in the rotation. Similar results are shown using the ME method, confirming that the noise is introduced as a result of the cell setup.

As for the other methods, most of them rely on a sampling method that is sensitive to the contact surface area. The noise for each of these cases is introduced as a result of random sampling throughout the domain. The overall trend stays constant throughout the rotation cycle. The chain length method shows variance peaks where the sampling method runs into a sharp corner and fails.

2.6 Mixing indexes to be used

In this study the three main mixing indexes that will be used to compare the mixed states are - the Particle Scale index, the Average Height method and the Mixing Entropy. Firstly, the Particle Scale index - this mixing index produces accurate results based on the mixed state of a blend based on the individual particles and their neighbours. It also takes into account the particle type composition of the entire blend.

Due to the uni-axial movement of RAM, the Average Height method is used in this study to compare the axial movement of the particles throughout the blend. This method, in particular, would provide valuable information on the results, especially about flow direction.

The Mixing Entropy provides a segmented view of the blend in the cells used to determine the mixed state. The method is a more refined version of the classic Lacey method and is chosen to possibly identify any segregation layers in the blend.

The main reasons for using these three particular methods would be for the ease of calculating the mixing results pre and post blending, without considering historical particle movements. This allows for the determination of the mixing state in both the numerical and the experimental realms.

Chapter 3

Numerical Setup and supporting material

This chapter is focused on the computational simulation of DEM for this specific use case, by making use of batch processes in Star-CCM+ on a high-performance computing system. In addition, this is an outline of the numerical analysis to be conducted in the following chapter. The chapter will take the reader through the process to set up a batch process on the Computing Cluster of the CSIR's Centre for High-Performance Computing. A total of 720 simulations equating to 7670 hours of computing hours will be conducted. A total number of 24 cores are available at a time to compute, thereby reducing the wall clock time from 7670 hours using one core, to 320 hours using the 24 cores.

The Star-CCM+ environment provides a stable backbone to do a discrete element flow, especially when solving multiple flow fields, with the emphasis on the investigation of liquid flow fields and their influence on solid interfaces. This includes wear patterns, thrust calculations, particle flow fields and many more. The upside of using a code such as Star-CCM+ is that the code has been proven, accepted and is trusted by many engineers across the globe - this also means that the code is being implemented in High-Performance Centres at one's disposal. The downside of using such a code is that in some instances it might be slow to implement.

Star-CCM+ offers a range of simulation capabilities in the particle flow field domain. The two dominating flow fields can be split into two main techniques, namely Eulerian and Lagrangian flow fields. The Eulerian flow field uses a continuum mesh field in a control volume to solve the particle flow field. This method ignores the particle-particle interaction because the particles are usually considered small and pass each other with a small particle interaction and the dominant flow field around it dictates the particle movement. The particle-wall interaction is, however, considered a normal boundary condition in a flow field and is similarly enforced in the particle flow field. The particles however can alter the fluid flow field, whereas the Lagrangian flow field includes the particle-particle interaction and the particle-wall interaction as a mechanism for the particle movement and fluid flow field. The Lagrangian method is thus the more effective method for simulating larger particles. In Figure 3.1, we see a comparison between the two techniques.

When solving a discrete element model one needs to consider the following hurdles:

1. When solving a Lagrangian flow model, each particle is subscribed to a computing core. The physics models are simple and easy to solve (position, velocity and acceleration), something that a graphics processing unit can solve using multiple cores (Govender, Wilke, Pizette, et al., 2017). Making

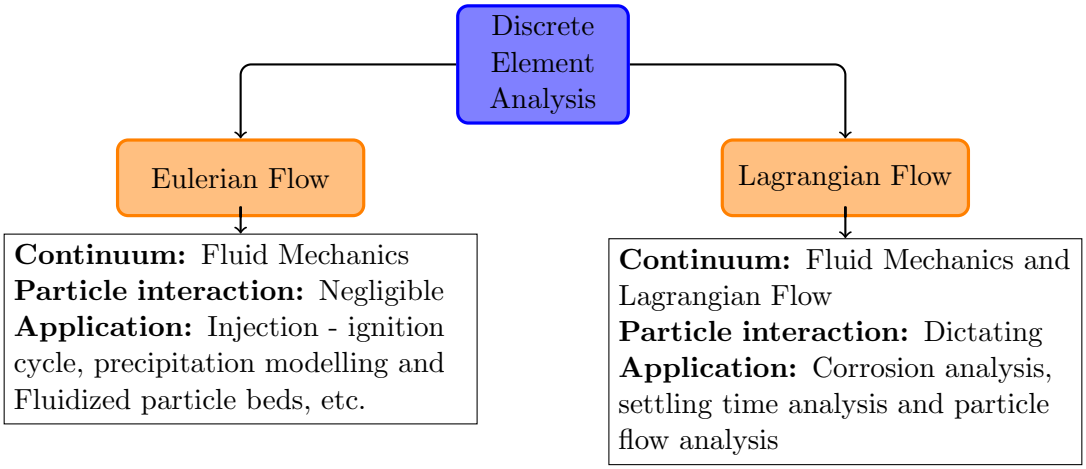


Figure 3.1: DEM-flow field comparison

GPU processing in this case up to 500-1000 times quicker than the standard 4-8 core CPU powered machine. Thus the solution time is highly dependent on the number of particles in the domain.

2. The particle material properties are directly linked to the solving time. Increasing the particle elasticity requires contact models to sub-step to solve the correct penetration for each particle. Sub stepping adds multiple iterations to the simulation, increasing the solving time.
3. Adding complex or non-convex particles to the system drastically increase the solution time (Mann, 2019).
4. Solving more domains causes the process to slow down, i.e. Solving Lagrangian flow and CFD. This may seem obvious but it is vital to 'freeze flow' in doing just Lagrangian flow on Star-CCM+.

3.1 Simulation Initialization

To initialize a DEM model on Star-CCM+ one should first pay attention to a number of initial details, such as the:

- Software licensing
- Creating the internal volume
- Setting up the physics environment
- Setting up the motion field functions

3.1.1 Licensing

Star-CCM+ licenses are typically kept on a server and whenever a user reserves a copy from the server one less license is available. The licenses are allocated to a single core used, thus if one simulation uses two cores to run a simulation, two licenses are reserved. The user has to create an environment variable (`CDLMD_LICENSE_FILE` and `LM_LICENSE_FILE`) in the operating system, pointing to the license server.

The licence files can be quite expensive for a single or small business owner. For a more affordable solution, some companies host licence files to customers at a fee per hour. These tokens are called Power-on-Demand tokens. Star-CCM+ does not allow one to edit a file without a valid licence. In the 'create new-file' or 'open

existing dialogue', the user will be prompted with the option to use Power-on-Demand tokens, in-house licenses or various other options. The dialogue will also prompt the user to select the number of cores that will be used for the simulation.

3.1.2 Create internal volume

The first step to look at is creating the flow volume. The flow volume is a geometric part that will constrain the particle movement to the internal volume of the holder. In this case, a wireframe model is drawn of a cylindrical shape. As seen in Figure 3.2, the volume is constrained by two variables, volume and height. Modelling a cylinder as control volume due to the concave internal surface it provides ensures a continuous and smooth surface where the particles can interact with the wall. Adjusting the volume only requires two parameters to change; height (H) and diameter (D). The geometry is created in Star-CCM+ by using a base sweep function using a circular base.

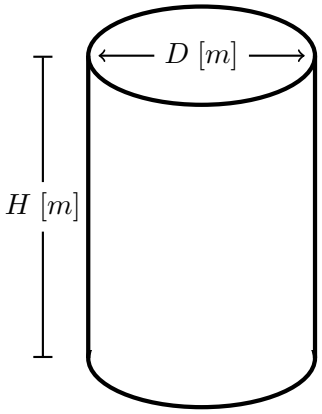


Figure 3.2: Internal Volume

3.1.3 Create Physics Environment

After creating the internal volume, we can create the physics environment; this is where the simulation environment is transformed into a Lagrangian flow environment. Figure 3.3 shows all the boxes that have to be ticked to ensure the right environment. To shed some light on the assumptions made for this specific environment refer to the following:

1. The *Multiphase Interaction* environment was set to accommodate the interaction between particles and the fluid (air) environment. This is necessary to add as Star-CCM+ uses the environment to solve the particle flow, although in this setup the fluid flow is frozen or not calculated.
2. The *Multiphase Lagrangian* environment is automatically selected when selecting the DEM module.
3. *Gravity* plays a large part in the mixing of the particles, as we assume that the largest denominator of large particle flow is the kinetic influence of the particles.
4. *Laminar* flow is cheaper to calculate and as mentioned above the flow is switched off; thus, the cheapest model is chosen to decrease simulation time. *Constant Density* is also cheap to compute and is selected for the same reason, *Gradients* are automatically selected. *Coupled Flow* and *Implicit Unsteady* are also selected to reduce simulation time and is also frozen.
5. a *Three Dimensional* model is required to compute the results as the probability of a symmetrical outcome is very unlikely.

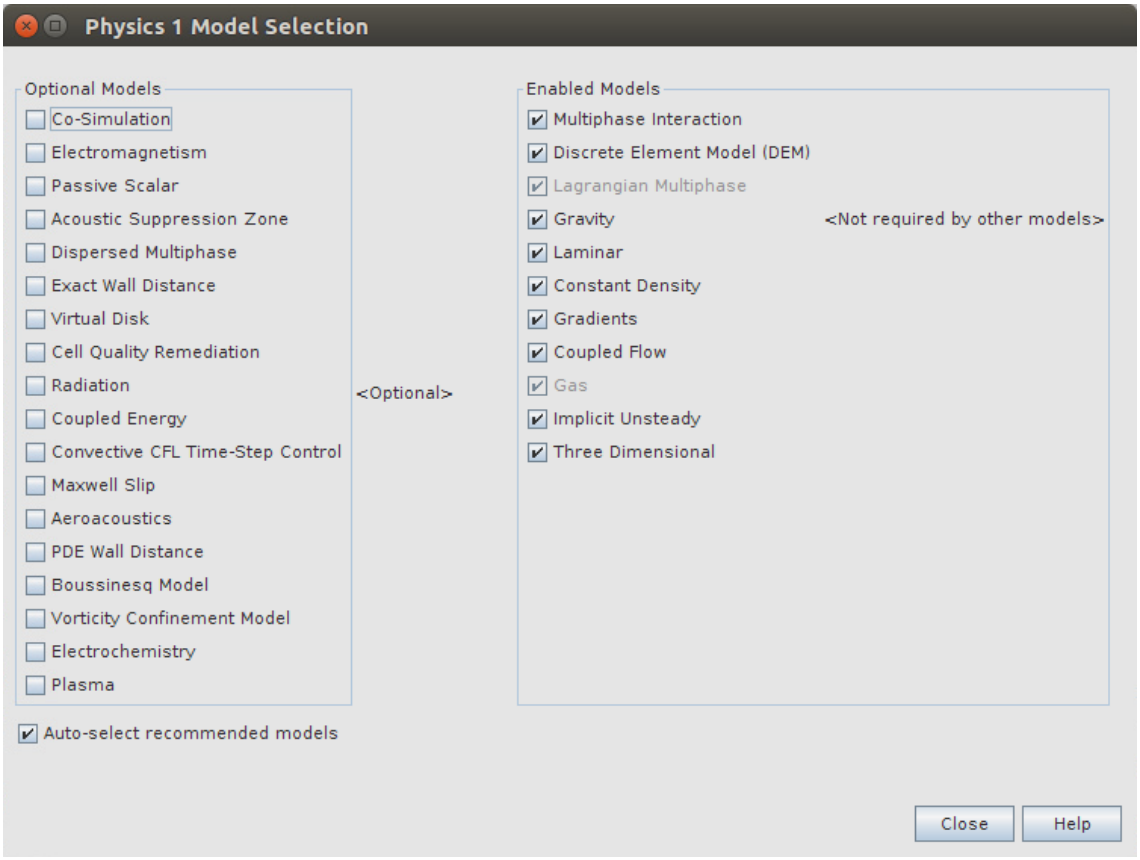


Figure 3.3: Physics environment setup

As illustrated above, a specific flow setting is needed for the simulation to run. It is however unnecessary for this particular study. Nevertheless, if the fluid environment should have an impact these flow settings need to be calibrated and implemented.

From here onwards, various monitors, tables and environments are being set up to monitor the particles and these are being exported to external files to be processed outside of the Star-CCM+ environment. It has been found that the monitors and processes that are being used to track particles in the Star-CCM+ environment are inadequate for this project.

3.1.4 Hertz-Mindlin contact model setup

Star-CCM+ provides two basic contact models for DEM, namely the Hertz-Mindlin models and the Walton-Brawn model (Siemens PLM Software, 2017). The Walton-Brawn model is focused on modelling deformable particles, as where the Hertz-Mindlin model is focused on modelling rigid bodies. In this study, rigid bodies are assumed and thus the Hertz-Mindlin model was chosen.

The model can be extended with the following additions:

- Rolling resistance
- Linear cohesion
- Artificial viscosity

- Parallel bonds
- Conduction heat transfer

As mentioned in Section 1.3.1, only Hertz-Mindlin contacts will be considered for this study- rendering all the additions above unnecessary.

Particle types

Star-CCM+ can model particles in three particle types:

- Spherical - considered and used for this study.
- Composite - Rigid and unbreakable particles, used to model geometric complex particles. The composite particles are made up of clumped smaller spherical particles to represent a larger particle.
- Clumps - Flexible and breakable particles, used to model geometric complex particles. It uses a similar approach to the composite particles; however, the bonds between the spherical particles are flexible and can break apart with enough applied force.

Contact model input variables

The Hertz-Mindlin model as discussed in Section 1.3.2.1 is dependent on four main particle properties.

- Particle elasticity or Young's modulus - E^*
- Particle radius or size - r^*
- Particle mass - m^*
- Particle poisons ratio - ν^*

Star-CCM+ accepts these constraints as part of the simulation setup. Instead of using particle mass, Star-CCM+ rather accepts particle density as an input and calculates the particle mass internally.

Particle restitution and other interaction properties

Additional to the Hertz-Mindlin contact model, Star-CCM+ also accepts various particle interaction properties that aid in the simulation. The additional properties that are used in this study will be:

- Particle and wall restitution
- Particle and wall friction

3.1.5 Field functions

Field functions are user inputs set into the physics and geometric environment of Star-CCM+. It is typically used to control the non-linear behaviour of certain variables. Star-CCM+ accepts a velocity field function as an input to the simulation.

In this case, we want to apply a velocity profile to the domain. A constant velocity profile is impractical and unrealistic to implement, due to the steady acceleration required. A more realistic profile to simulate and replicate experimentally will be a continuous linear displacement. For this a 6^{th}-order polynomial can be used to describe the angular velocity profile (equation 3.1). Figure 3.4a shows the profile for a specific case using the following input parameters $f = 1Hz$, $A = 20mm$ and $1s$ time delays pre and post agitation. The profile parameters can be calculated using a non-linear solver.

$$\omega(t) = a(t+b)^6 + c \tag{3.1}$$

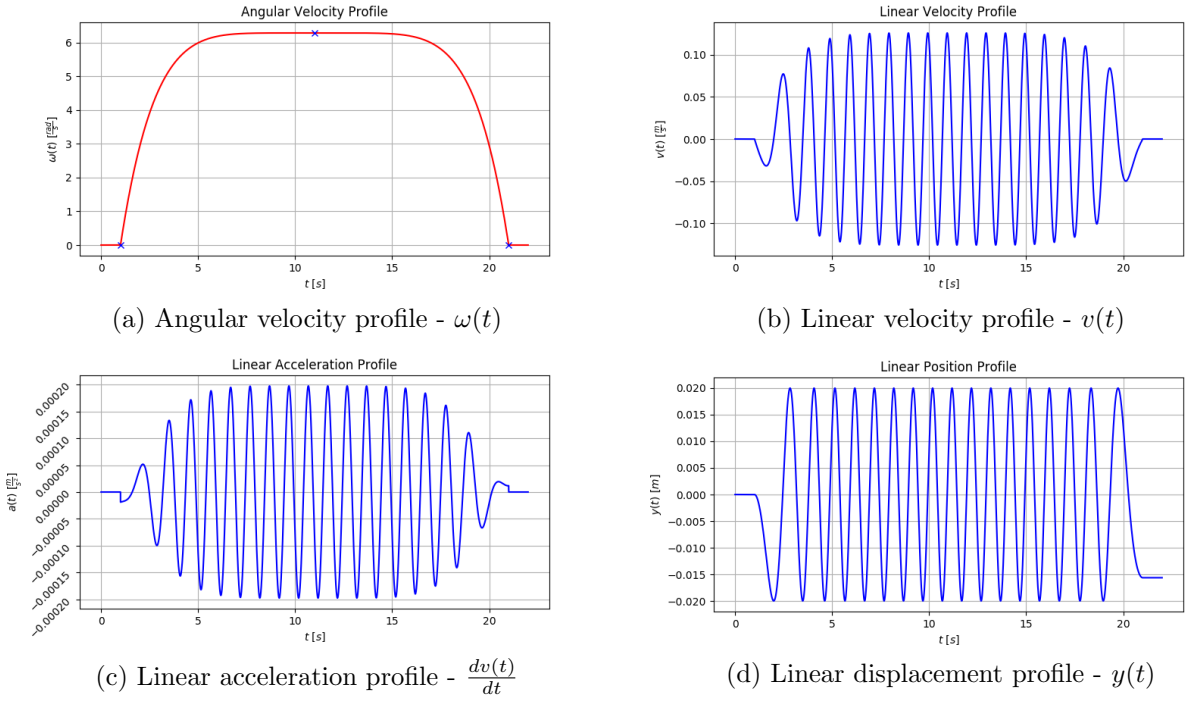
Integrating the angular velocity profile the angular displacement profile can be determined:

$$\theta(t) = \int \omega(t)dt = \frac{a}{7}(t+b)^7 + tc - \left(\frac{a}{7}(t_1+b)^7 + ct_1\right) \tag{3.2}$$

The linear velocity profile $v(t)$ as illustrated in Figure 3.4b can be written as below, where A is the amplitude.

$$v(t) = -A\omega(t)\cos(\theta(t)) \tag{3.3}$$

Equation 3.3 (Fig. 3.4d) will be the input to Star-CCM+. This provides the information to the algorithm to apply to the field function. Figure 3.4c shows the acceleration and position profile of the function mentioned above for interest sake. One can see that the start and end positions differ in Figure 3.4d due to the speed it travels for the time it might not be in sync with the sinusoidal period.



(a) Angular velocity profile - $\omega(t)$

(b) Linear velocity profile - $v(t)$

(c) Linear acceleration profile - $\frac{dv(t)}{dt}$

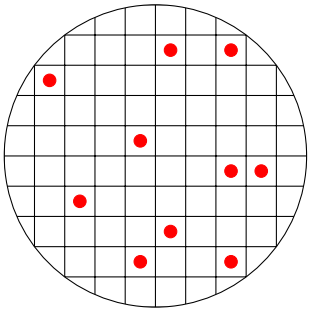
(d) Linear displacement profile - $y(t)$

Figure 3.4: RAM movement profiles

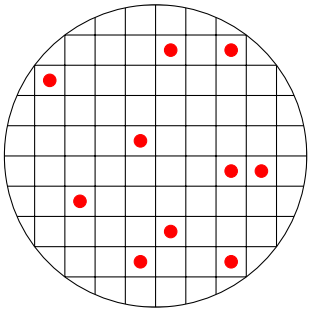
3.1.6 Injector grid

For Star-CCM+ to inject particles into the domain the software uses an injector on a presentation grid or surface by injecting particles at a given particle speed and flow rate into the domain. This grid randomizes the particle injections based on the available injection ports.

In Figure 3.5 the presentation grid is represented. The flow rate is divided over the available real estate and introduced into the domain at the available nodes. The re-randomize function in Star-CCM+ sets the particle inclusion probability of the node. The inclusion probability is the probability that the node will be occupied with a new particle at the next time step.

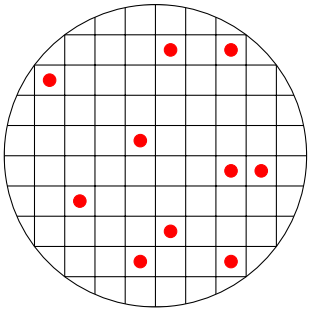


Initial injection - Pass 1

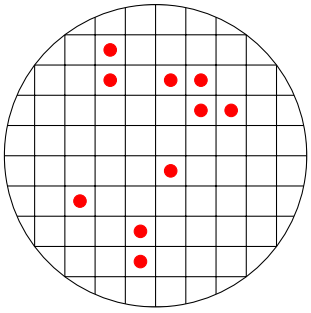


Initial injection - Pass 2

(a) Run same simulation



Initial injection - Pass 1



Initial injection - Pass 2

(b) Re-run the same simulation with clearing the domain

Figure 3.5: Presentation Grid

Star-CCM+, unfortunately limits the consecutive testing in a parallel environment in the following manner; internally the algorithm has a pre-set flow specification on the particle injector. Thus a randomized vector was built into the algorithm to allocate the particles to their real estate in the grid; this causes two simulations to run side by side with the same results, as illustrated in Figure 3.5a. This however is overcome by clearing the simulation after a run and restarting it. This causes the algorithm to recognize a new simulation has started and randomizes the particle allocation vector again; this is illustrated in Figure 3.5b.

When re-randomizing the injector grids for every simulation it introduces stochasticity in the simulations, while keeping the bulk of the simulation characteristics.

3.2 Setup HPC workflow

The Star-CCM+ environment provides a platform where one can record and play macros to automate processes. This is particularly helpful when processing large amounts of simulations, and as mentioned above, no post-processing will be conducted in Star-CCM+. Batch processes were set up where no GUI was needed to interface with the user. A couple of things to take note of are:

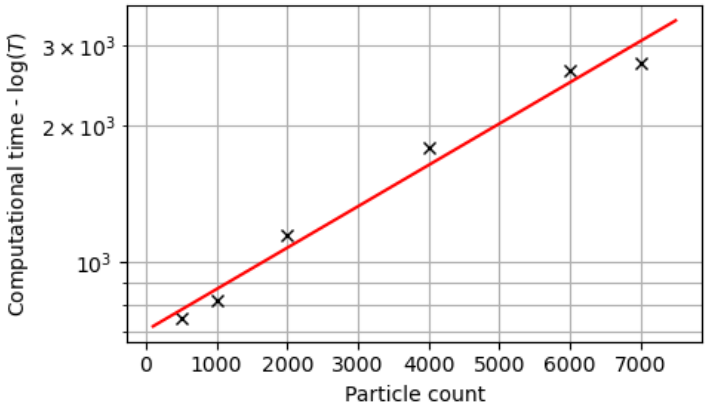
- No GUI means that it is more difficult to troubleshoot a faulty simulation.
- Adding environments and modules to an existing simulation is easier using the GUI.
- Simulation progress is easier to track in the GUI.

- The GUI in Star-CCM+ has been renowned for drawing accurate and realistic simulation renderings, which is impossible when using a server to calculate results that have no graphical processing unit (GPU).
- Setting up a batch simulation in the GUI is impossible, as only one simulation can be shown at a time. The GUI can however connect to a running simulation on the given server to display the results being calculated.
- The GUI is not needed, as stated above; the graphical processing does not provide adequate information to calculate the necessary results for the study.
- The GUI cannot be launched on a server like the CHPC (Centre for High-Performance Computing), because it is a command-line based interface.
- It is easier to run the simulations embarrassingly parallel in this manner.

When considering the problem at hand, a resonant acoustic mixer with a speed of $60Hz$, the shaking amplitude is $20mm$ with spherical particles with an elasticity of $10MPa$. Each simulation was run for $1.0s$ and time steps of $5e-4s$. The aim was to see the simulation time scaling for the number of particles in the system.

Resolving in a logarithmic RMSE curve, Figure 3.6:

$$T(p) \sim 700e^{(2E-04)p}[s] \tag{3.4}$$



Particle Count	Sim. time [s]
500	747.75
1000	817.33
2000	1145.37
4000	1786.99
6000	2647.57
7000	2742.17

Figure 3.6: Particle count time scaling

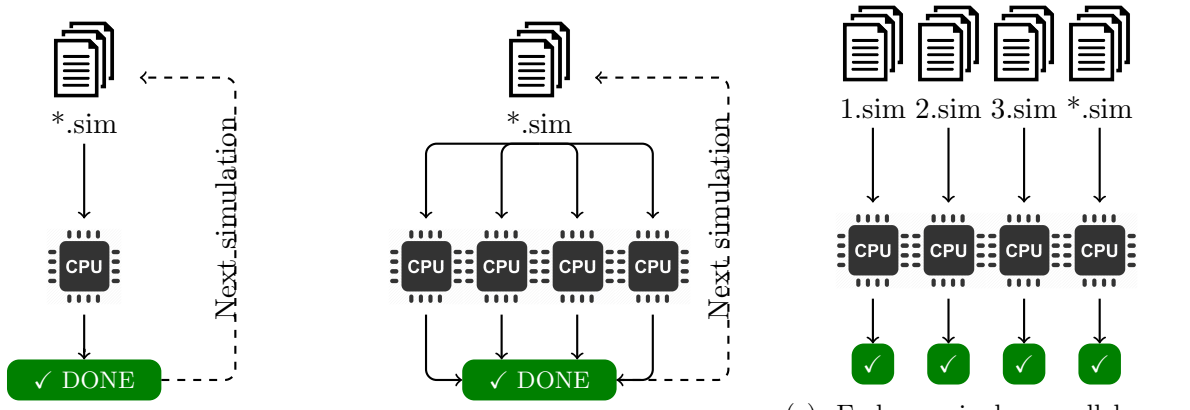
We can expect that the time will scale slightly super-linear as the logarithmic coefficient is close to 0 and then $e^{2E-04} \sim 1$. The base time coefficient of 700 in *equation* 3.4 is merely a guideline as this is a variant of particle elasticity, density and various other factors.

Lagrangian computing generally requires a parallel processing approach to solve a simulation within a reasonable time. This means that a problem of n particles divide the computing power over the m cores in a processor. This is essentially the case in Star-CCM+. The code uses a feature called Load Balancing, which splits up a large number of particles into the various cores used in the simulation. This is however only useful when dealing with huge amounts of particles. Table 3.1 shows the results when simulating with 4, 8, 16 and 32 cores. The load balancing option refers to the iteration frequency that the code refreshes each particle allocation to a specific core.

Table 3.1: Balancing scaling

[HRS:MIN:SEC]	**Load Balancing Option**		
Cores	5	10	20
4	1:16:03	1:17:49	1:17:07
8	1:15:35	1:14:30	1:14:58
16	1:21:33	1:16:49	1:16:56
32	1:20:10	1:16:25	1:16:50

To solve this problem, one can do one of three things. Run the simulations in a series process one after the other. Run the simulations in a parallel process (with load balancing), or run the process embarrassingly parallel. Refer to Figure 3.7 for further information. Stating in conclusion, one serial process is just as time-consuming to do as the same process in parallel processing. Thus, by solving multiple processes at once we can finish all processes at the same time, but more gets done.



(a) Series processing (b) Parallel processing (c) Embarrassingly parallel processing

Figure 3.7: Processing processes

3.2.1 Batch environment

The Star-CCM+ pulls its power from being able to rope in multiple cores from various sources. This is done via creating SSH tunnels between cores within a single processor and/or creating SSH tunnels between multiple computers on a server. Thus it is vital to ensure that the SSH tunnels from the licence file to the respective server nodes are also open. Refer to Listing 3.1 and refer to Table 3.2 for the Port and Host definition used in the listing.

Listing 3.1: SSH Tunnels from License to server nodes

```
1 #!/bin/bash
2 #Export the environment variables:
3 export LM_LICENSE_FILE=<virtual_port>@localhost
4 export CDLMD_LICENSE_FILE=<virtual_port>@localhost
5 #SSH into licence server (Open communication):
6 #lmgrd deamon
7 ssh -f <username>@<server_lic_store> -L <your_lic_port>:<your_lic_server
    >:<your_lic_port> -N
```

```
#vendor deamon
ssh -f <username>@<server_lic_store> -L <your_lic_port_d>:<
   your_lic_server>:<your_lic_port_d> -N

#Code body

#Finds and kill SSH tunnels (Close communication)
kill -9 `ps ux | grep "ssh -f" |  grep -o -E '[0-9]+' | head -1 | sed -e
   's/^0\+//' `
kill -9 `ps ux | grep "ssh -f" |  grep -o -E '[0-9]+' | head -1 | sed -e
   's/^0\+//' `
```

Table 3.2: Ports and Host definition of Listing 3.1

<virtual_port>	The port that will be reserved on the local computing node for the SSH tunnel to enter.
<username>	User name used to log into remote server.
<server_lic_store>	Host on the remote server where all the licences are stored.
<your_lic_port>	a Port on which the licences are made available from the host (users) side.
<your_lic_port_d>	A Port on which the licences are made available from the host (users) side for the vendor daemon (usually one port up from the <your_lic_port> port).
<your_lic_server>	a Server on which the licences are held.

Within the code body, various mechanisms can be placed to allocate folders and link variables to time functions, but the main function to launch Star-CCM+ in batch mode is described in Listing 3.2.

Listing 3.2: Star-CCM+ batch launch command

```
starccm+ -licpath $CDLMD_LICENSE_FILE -batch <my_file>.java -rsh ssh -np
   <number_of_cores> -load <master>.sim > <my_file>.out
```

Table 3.3: Variable and File definition of Listing 3.2

<my_file>	This file identifier will correspond to the Java Macro that is used to alter the various variables and also write to a plain text simulation timeline *.out file.
<number_of_cores>	Allocates the number of cores that will be used in this instance.
<master>	This indicates the master simulation file to be run. This file is the basic skeleton from which the macro will be operating.

3.2.2 Java Macros and Automation

Now that we understand a bit more of how distributed computing is done the attention can be shifted to data sweep and automation. By pre-recording a macro in Star-CCM+, it can be used as a skeleton file, adapting the necessary variables to each case scenario. The variables are optimized within the non-dimensional sweep on a priority basis to ensure the quickest possible simulation. The priorities are as follow:

1. Ensure an elastic material without changing the density to an unrealistic value.
2. Ensure the particle count stays as low as possible.
3. Ensure the physical constraints are such that they can be easily replicated experimentally.

This is done by presenting an initial condition where some of the variables stay constant and the dependant variables are optimized to ensure the correct non-dimensional field is processed. This is achieved by the Optimization and root-finding package (Scipy Optimize - SciPy.org)

The Macro runs through the following commands to set the parameters in the master file.

1. Edit Volume
2. Add particle parameters
   (a) Particle density
   (b) Particle elasticity
   (c) Poissons Ratio
3. Set particle interaction models
   (a) Set Hertz-Mindilin model
   (b) Set interparticle and particle-wall friction coefficients
   (c) Set interparticle and particle-wall restitution coefficients
   (d) Set cohesion models
4. Add particle injector lattice
5. Set particle injection flow rate
6. Set cup speed
7. Freeze flow
8. Set time-step
9. Set simulation time
10. Add export tables, populated by:
    - Particle ID
    - Particle Mass
    - Particle Type
    - Particle Velocity (x,y,z-directions)

- Particle Position (x,y,z-directions)
- Time step

11. Save the simulation under a unique name
12. Run simulation

In summary, the whole process can be described in the flow in Figure 3.8. Each of the processes described above plays a key role in extracting the necessary data. The process starts by initiating a RAM case with a set of initial values. These values are evaluated in the predetermined set of non-dimensional parameters. These parameters are then used to extract an optimized set of parameters. These are the parameters that are used to plug into a Java Macro and run the simulation from the Centre for High-Performance Computing. On the system, a tar-ball is created to compress the .csv tables that are exported from Star-CCM+ per problem, per time step.

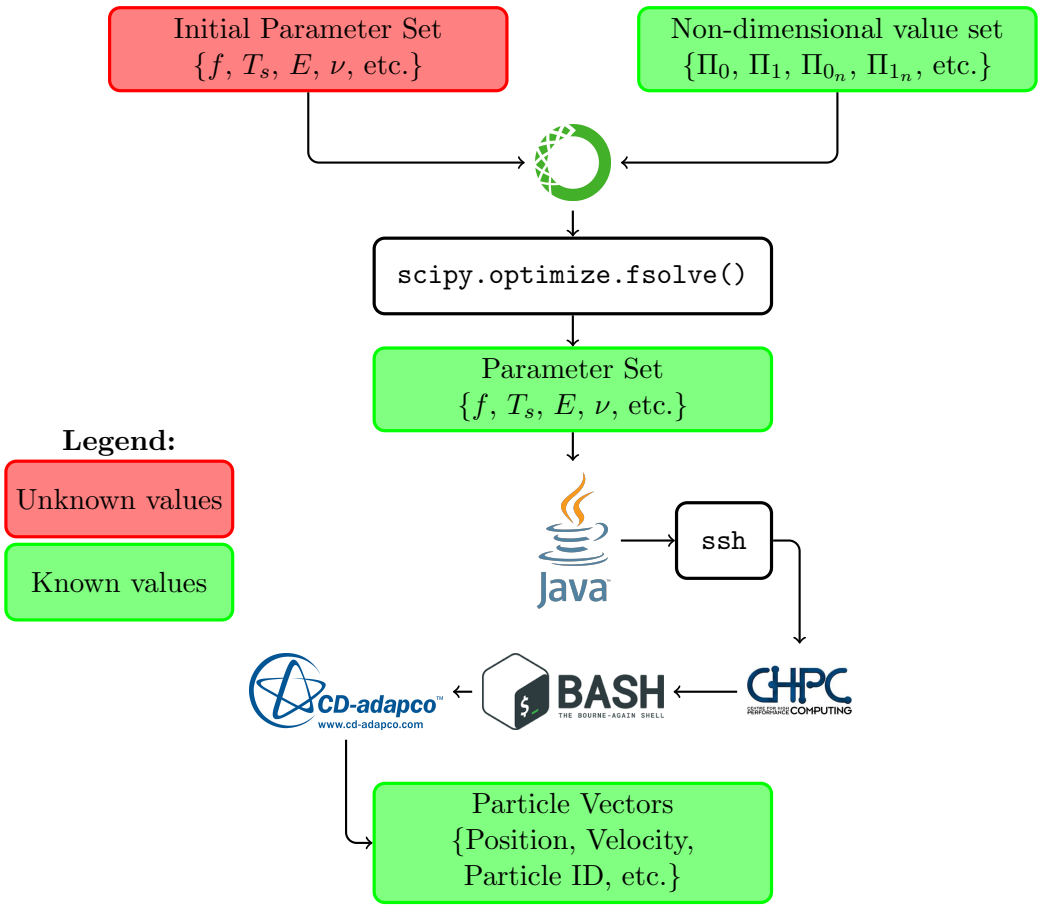


Figure 3.8: Numerical setup process flow

3.3 Batch processing of results

Natively Star-CCM+ does not support mixing index functions. To calculate the MI the post-processing of results will be conducted in Python by exporting tables that are generated in Star-CCM+. The tables contain all the information needed to compute all the mixing indexes mentioned in this study. The data is stored in a comma-separated value file and then processed via the Pandas toolbox in Python to produce

a hierarchical data format (HDF) file that is similar to a comma-separated value format with the added functionality of adding additional sheets to the file. This file manages to compress the data and convert multiple CSV files into one workable file.

These HDF files allow the capability to process the results outside of Star-CCM+ and being able to structure the data in every way possible to extract the most valuable information out of it.

Refer to Listing C.1 for the full script on determining each of the mixing indexes mentioned in this report.

Chapter 4

Dimensional Analysis and Numerical quantification

The next step is to identify and prioritize parameters that affect the mixing outcome and behaviour. Variables affecting mixing ranges from particle scale parameters to macroscopic parameters. In this chapter, a dimensional study is conducted to identify non-dimensional groups and identify independent parameter groups. The idea of a non-dimensional study is to reduce the dimensionality of the system by reducing the variables to fractions of each other.

These variables and parameters range from particle level to volumetric dimensions and energy inputs. Given n parameters and k primary dimensions, we can reduce it to $j = n - k$ independent parameter groups. This is then followed by a numerical investigation, in which we determine the validity and sensitivity of these groups. This will allow for the prioritization of each non-dimensional group, validating the results in an experimental study in Chapter 5.

The sensitivity study will help us understand the parameters and variables that play a larger role in this non-linear problem. There are two main methods in which a sensitivity study can be conducted, namely variable reduction and/or a dimensional study. In Figure 4.1 the processes of variable reduction and dimensional analysis are illustrated.

(a) Variable reduction

(b) Dimensional analysis

Figure 4.1: Variable reduction vs. Non-dimensional analysis

The variable reduction method illustrates that some variables show minimal impact on the system and are insensitive to the results; they can thus be eliminated as an insensitive variable. In the dimensional analysis environment, there are two attributing variable sets - the X_j dimensionless set, and the x_i dimensional set. The x_i-set are all the variables that can be identified in the problem. The X_j-set consists of the dimensional

variables x_i, the X_j variables and are written mathematically to balance the variables so that the X_j variables are dimensionless. In dimensional analysis, it is important to ensure, that by changing one variable (X_j), multiple variables (x_i) can be manipulated. The results consequently can reduce computational time. It also helps to simplify a problem by grouping sets of variables conveniently and descriptively.

It is important to identify valid groupings of variables and test if the variables (x_i) are indeed insensitive of variance within each grouping. It is essential not to misrepresent the dimensionless groups, by grouping irrelevant variables together, as it may skew the results. Thus it is important to verify the validity of the group by varying each parameter within each variable group, such that a constant dimensionless value is obtained and to see if the results stay constant. If this is the case, the group can be classified as a valid dimensionless group.

The method being used to do the dimensional analysis here is the Buckingham-Π method Sonin, 2001, a fairly robust and easy-to-use dimensional study method. As part of the study, later on, a variable reduction will be done on the identified dimensionless groups to identify core parameters that influence the mixing outcome.

Grouping variables in this manner poses some risk to the result of the experiment. These risks are usually inconsistent results when varying the inner variables and keeping the Π group stable, or no sensitivity to change at all. It is therefore important as indicated in Gupta, Sarkar, Chychko, et al., 2014 that the success or failure of the method depends on the selection of these variables. One may not succeed in identifying the best groups at the start and testing the validity and sensitivity of each group is paramount. Due to time constraints, this selection could not have gone further than the first iteration and can be explored at a later stage.

4.1 Dimensional study

The variables that may potentially influence mixing can be grouped into three different categories, volumetric or geometric variables, material properties or variables and energy or dynamic variables.

Figure 4.2 illustrates the variables that were identified for this specific case. In this instant, only two particle types are considered, P_1 and P_2. Additionally, for this study, the cohesiveness of the particles was not considered as this brings additional complexities into the experiment.

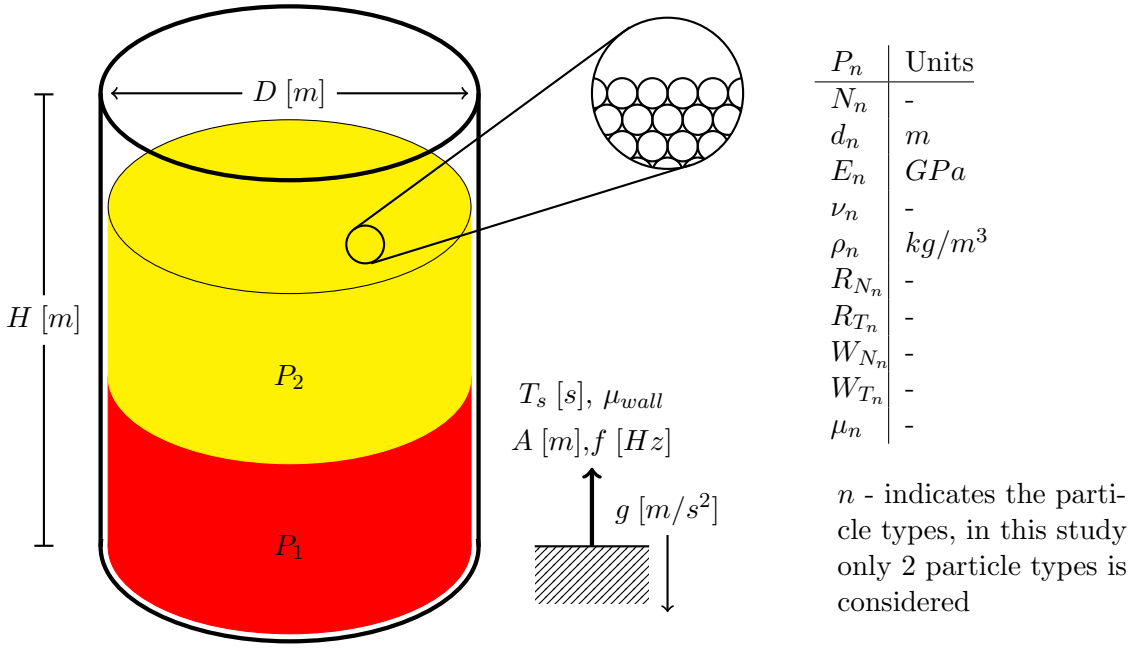
Based on the Buckingham Π theory, the variables are split into two groups - independent and dependent variables. Independent variables are variables that function independently in the problem. Dependent variables are variables that are influenced by the independent variables. For instance, in this problem the cup dimensions are independent of each other, the height and diameter of the cup can be set at any given value, however, the particle position in the cup is a constraint within the dimensions of the cup, making the particle positions a dependent variable.

Within the system, 16 independent variables can be identified. These independent variables can be separated into two main groups, globally defined variables and locally defined variables. Global variables are variables that are not dependent on the particle type. Local variables are variables that can be defined separately per particle set.

The dependent variables for this system are defined by the positional and motion vectors of each particle within each particle type (n). The positional, velocity and acceleration vectors are defined by;

For positional;

$$\mathbf{x}_n = \langle x_i, \quad x_j, \quad x_k \rangle \tag{4.1}$$



P_n	Units
N_n	-
d_n	m
E_n	GPa
ν_n	-
ρ_n	kg/m^3
R_{N_n}	-
R_{T_n}	-
W_{N_n}	-
W_{T_n}	-
μ_n	-

n - indicates the particle types, in this study only 2 particle types is considered

Figure 4.2: Dimensional Analysis - Independent Variables of a non-cohesive two-particle type RAM setup

For velocity;

$$\dot{\mathbf{x}}_n = \langle \dot{x}_i, \quad \dot{x}_j, \quad \dot{x}_k \rangle \tag{4.2}$$

And for acceleration;

$$\ddot{\mathbf{x}}_n = \langle \ddot{x}_i, \quad \ddot{x}_j, \quad \ddot{x}_k \rangle \tag{4.3}$$

Each of these vectors then plug into the various mixing indexes, which will in turn, quantify the mixing performance of each case.

4.1.1 Buckingham Π theorem variable grouping

The Buckingham Π method reduces the variables in a system by k dimensionless variables. Where j is the number of variables in the system and m is the number of dimensions in the system Buckingham, 1914, k is defined by $k = j - m$. The Buckingham Π theorem decomposes the dimensionality of each variable by mathematically rewriting variables to simplify the dimensionality so that the groups are dimensionless. The decomposition of any variable is based on three base units - L representing length or distance, T representing time and M representing mass.

The best way to describe the Buckingham Π method is per example - the well-known Reynolds number is such an example. This is a non-dimensional number that is made up out of various dimensional parameters such as v - fluid velocity $[m/s]$, ρ - fluid density $[kg/m^3]$, l - characteristic length $[m]$ and μ - dynamic viscosity $[Ns/m^2s]$. Below we can see that all the base dimensions are simplified and reduced to a dimensionless variable.

$$Re = \frac{\rho \nu l}{\mu} = \frac{(ML^{-3})(LT^{-1})(L)}{(M(LT)^{-1})} = 1. \tag{4.4}$$

When implementing the Buckingham Π method, a good rule of thumb is to pair variables that relate to each other. The motive behind a dimensional study would be to form functional dimensionless groups by combining the independent variables. It should be grouped in such a way as to describe a type of measurement - for instance, if one would like to describe the shape of a rectangle, the ratio between the width and length of the rectangle would sensibly describe the shape of the rectangle.

The dependant variables $\mathbf{x}_n$, $\dot{\mathbf{x}}_n$ and $\ddot{\mathbf{x}}_n$ feed into various mixing indexes M. The mixing indexes as discussed in Chapter 1 are non-dimensional measurements and will thus replace the need to identify additional non-dimensional variables to describe mixing. The following expression describes the dimensional study, where k is the number of dimensionless variables, $k < j$ and n is the particle type.

$$M(H, D, A, f, T_s, \rho_n, d_n, E_n, \nu_n, \mu_n, \mu_{wall}, R_{N_n}, R_{T_n}, W_{N_n}, W_{T_n}, N_n) = M(\Pi_1, \Pi_2, \Pi_3 \dots \Pi_k) \tag{4.5}$$

Firstly, it is necessary to identify each of the variables base units and after doing so, it is a matter of grouping the variables in such a way that they balance each other out to form dimensionless groups. Table 4.1 shows each of the variables as identified in Figure 4.2 decomposed into their base units.

As seen in Table 4.1, there are 8-dimensional variables identified in this problem and 3 dimensions M, L and T. Furthermore, there are another 8 already dimensionless variables. By only considering the dimensional variables we can expect to reduce the number of inputs to $k = 5$. Additionally, the already dimensionless variables can either be grouped into some of the newly identified groups or be grouped or be kept standalone as a dimensionless variable.

By applying the Buckingham Π theorem the following set of Π groups of $k = 10$ can be identified. The Π groups consist of 8 groups derived from the dimensional variables and 5 additional dimensionless groups added by utilizing the dimensionless variables in Table 4.1. This reduces the number of variables from 16 to 10.

Considering the dimensionless variables in equations 4.6 to 4.15 one can start to understand how the mixing might scale for each of the parameters.

- Equation 4.6: Π_0 is the relationship between the frequency and the run/blending time. If the frequency is slower the runtime will have to be longer to achieve the same result than if the frequency is quicker. When the frequencies are too low this Π-group breaks down. It is anticipated that this dimensionless group is valid at higher frequencies.
- Equation 4.7: The variables in this group describe the aspect ratio of the cup. The variables control the cup height dimension in relation to the cup diameter and vice-versa.
- Equation 4.8: The relationship between the cup acceleration and gravity is captured by Π_2. This group is one of the variables that is being implemented by RAM (Chaudhuri, Muzzio, and Tomassone, 2006)(*LabRAM I - Resodyn Acoustic Mixers*). The variable is defined as the mixing intensity. The relationship dictates whether the acceleration is enough to overcome the gravitational forces on the particles. This is a vital measure to control as described by Osorio and Muzzio, 2015, who proved that the increase in acceleration depreciates the mixing.
- Equation 4.9: Π_3 defines the relationship between the particle volume and the mixing volume. The variable has also been identified by ibid. as a sensitive variable in RAM. It was concluded that, as the

Table 4.1: Dependant and Independent variables in RAM shaking

	Independent variables			
	Description	**Variable**	**Unit**	**Base Unit**
Global	Run/Blending Time	T_s	s	T
	Frequency	f	Hz	T^{-1}
	Cup Height	H	m	L
	Cup Diameter	D	m	L
	Amplitude	A	m	L
Local	Particle diameter	d_n	m	L
	Particle density	ρ_n	kg/m^3	ML^{-3}
	Particle elasticity	E_n	Pa	$ML^{-1}T^{-2}$
	Particle count	N_n	-	1
	Particle Poissons ratio	ν_n	-	1
	Particle restitution - normal	R_{N_n}	-	1
	Particle restitution - tangential	R_{T_n}	-	1
	Wall restitution - normal	W_{N_n}	-	1
	Wall restitution - tangential	W_{T_n}	-	1
	Interparticle friction	μ_n	-	1
	Wall friction	μ_{wall}	-	1
	Dependent variables			
	Particle position	$\mathbf{x}_n$	m	L
	Particle velocity	$\dot{\mathbf{x}}_n$	m/s	LT^{-1}
	Particle acceleration	$\ddot{\mathbf{x}}_n$	m/s^2	LT^{-2}

particle fill level (Π_3) increases, the mixing decreases. It is important to note that this is a volumetric fill level due to the packing density which is dependent on the radius ratio as seen in Table 1.1.

- Equation 4.10: The Π_4 group links the elasticity/potential energy of the particles to the kinetic energy of the particles imposed by the frequency variable, f. For this study, the average of this variable is taken for each particle type n, where m indicates the number of particle types. In this study $m = 2$.
- Equation 4.11: Π_5 is dimensionless by nature - the variable is the average Poissons ratio over the particle types n.
- Equation 4.12 & 4.13: These dimensionless variables compare the restitution of the particle-to-particle interaction and the particle-to-wall interaction. The two variables are weighted to favour either the normal interaction ($\Pi_6 > 1$ and $\Pi_7 > 1$) or the tangential interaction ($\Pi_6 < 1$ and $\Pi_7 < 1$).
- Equation 4.14 & 4.15: The average of the particle-to-particle friction coefficients are taken as Π_8 and the average of the particle-to-wall friction coefficients are taken as Π_9. The friction coefficients are already dimensionless and can form dimensionless groups on their own.

It is important to test the validity of each Π_k group. This is done by changing each variable, whilst keeping Π_k constant. If the results stay the same for every combination of the change in variables the Π_k group is deemed as valid. A study is conducted below to identify whether these Π-groups are valid. In cases where they are not found to be valid, their sensitivities are interpreted to identify whether they could be sensible variable groups for the MI study.

$$\Pi_0 = fT_s \quad (4.6) \qquad \Pi_1 = \frac{H}{D} \quad (4.7) \qquad \Pi_2 = \frac{Af^2}{g} \quad (4.8)$$

$$\Pi_3 = \sum_{n=1}^{m} \frac{d_n^3 N_n}{D^2 H} \quad (4.9) \qquad \Pi_4 = \frac{1}{m} \sum_{n=1}^{m} \frac{E_n}{d_n^2 f^2 \rho_n} \quad (4.10) \qquad \Pi_5 = \frac{1}{m} \sum_{i=1}^{m} \nu_n \quad (4.11)$$

$$\Pi_6 = \frac{1}{m} \sum_{n=1}^{m} \frac{R_{N_n}}{R_{T_n}} \quad (4.12) \qquad \Pi_7 = \frac{1}{m} \sum_{n=1}^{m} \frac{W_{N_n}}{W_{T_n}} \quad (4.13) \qquad \Pi_8 = \frac{1}{m} \sum_{n=1}^{m} \mu_n \quad (4.14)$$

$$\Pi_9 = \frac{1}{m} \sum_{n=1}^{m} \mu_{wall(n)} \quad (4.15)$$

4.2 Dimensionless Π-groups validity check

The independent variables in the study are listed in Table 4.2. Each of these variables is varied between the bounds in Table 4.2. These bounds have been identified to be easily tested in an experimental environment as well as giving the study a wide enough range of data points to draw results from. Each of these variables has its respective Π_k dependencies which are also indicated in the table.

Table 4.2: Variable independence non-dimensional group values

Variable	**Variable bounds**	**Affected dimensionless groups**
T	$1.625s$ - $5.000s$	Π_0 (eq. 4.6)
f	$30Hz$ - $70Hz$	Π_0 (eq. 4.6), Π_2 (eq. 4.8) & $\Pi_{1(n)}$ (eq. 4.10)
H	$50.0mm$ - $143.8mm$	Π_1 (eq. 4.7) & $\Pi_{0(n)}$ (eq. 4.9)
D	$50.0mm$ - $81.2mm$	Π_1 (eq. 4.7) & $\Pi_{0(n)}$ (eq. 4.9)
A	$2.5mm$ - $20.0mm$	Π_2 (eq. 4.8)
d	$3.6mm$ - $10.0mm$	$\Pi_{0(n)}$ (eq. 4.9) & $\Pi_{1(n)}$ (eq. 4.10)
N	-	$\Pi_{0(n)}$ (eq. 4.9)
E	$80GPa$ - $200GPa$	$\Pi_{1(n)}$ (eq. 4.10)
ρ	$600^{kg/m^3}$ - $5000^{kg/m^3}$	$\Pi_{1(n)}$ (eq. 4.10)
ν	0.33 - 0.49	$\Pi_{2(n)}$ (eq. 4.11)
R_{N_n}	0.01 - 0.50	$\Pi_{3(n)}$ (eq. 4.12)
R_{T_n}	0.01 - 0.50	$\Pi_{3(n)}$ (eq. 4.12)
W_{N_n}	0.01 - 0.50	$\Pi_{4(n)}$ (eq. 4.13)
W_{T_n}	0.01 - 0.50	$\Pi_{4(n)}$ (eq. 4.13)
μ	0.01 - 0.90	$\Pi_{5(n)}$ (eq. 4.14)
μ_{wall}	0.01 - 0.90	$\Pi_{5(n)}$ (eq. 4.14)

The variables which are indicated in Table 4.2 should be varied individually to test whether the Π_k groups

are valid. To accommodate the change of the other variables in Π_k, a control state of Π_k should be set to let the variables be adjusted accordingly. The control state of Π_k is defined in Table 4.3. These values will stay constant throughout the validity study to establish whether the Π_k groups are indeed valid variables.

Table 4.3: Π_k benchmark values - Variable sweep

$\mathbf{\Pi_k}$	**Value**	**Equation**
$\log_{10}(\Pi_0)$	1.85	4.6
Π_1	1.75	4.7
Π_2	0.58	4.8
Π_3	0.40	4.9
$\log_{10}(\Pi_4)$	9	4.10
Π_5	0.25	4.11
Π_6	1.25	4.12
Π_7	1.25	4.13
Π_8	0.55	4.14
Π_9	0.55	4.15

In conjunction with setting up a set of non-linear equations to be solved additionally the variables were calculated in the following steps to ensure the variables are calculated correctly. For each variable, a benchmark value is assigned as an initial value for the solver.

1. Either T or f is calculated based on equation 4.6 given the steady state value in Table 4.3. If neither T nor f is varied the benchmark values in each case are assigned as $T = 2s$ and $f = 35.4Hz$.
2. Thereafter the amplitude A is calculated based on the values previously assigned for f. This is done by solving A in equation 4.8 and the steady-state values in Table 4.3.
3. The set of non-linear equations are set up for equations Π_1, Π_3 and Π_4. For this set the following benchmark values are set;
   - $H = 0.0875m$ and $D = 0.05m$
   - $d_n = 0.006m$ and $N_n = 200$
   - $E_n = 80GPa$ and $\rho_n = 1774kg/m^3$
4. Then R_{N_n} and R_{T_n} is calculated, with benchmarks $R_{N_n} = 0.25$ and $R_{T_n} = 0.2$.
5. Likewise W_{N_n} and W_{T_n} is calculated, with benchmarks $W_{N_n} = 0.375$ and $W_{T_n} = 0.300$.
6. ν_n, μ_n and $\mu_{wall(n)}$ is set at the benchmarked values if not varied. The benchmark values are;
   - $\nu_n = 0.25$
   - $\mu_n = 0.55$
   - $\mu_{wall(n)} = 0.55$

4.2.1 Blending time and frequency

To validate a dimensionless variable group it is required to test the independence of all the variables in the system. This is done by keeping the Π_k variables at a fixed value while changing the values of the

variables that are used to compute the Π-group. For a Π-group to be valid it is expected that the MI remains unchanged as the underlying variables change. Variables cannot change one by one otherwise the Π-group values would change - and in turn, the MI is expected to change.

By varying the total blending time the variable change will affect one dimensionless group directly, Π_0, where the frequency f is adjusted and consequently two other variables are affected - the amplitude A and the particle density ρ. All the other variables are set at their respective benchmarked values and remain unchanged. Figure 4.3 illustrates this change in the variables based on their benchmarked values.

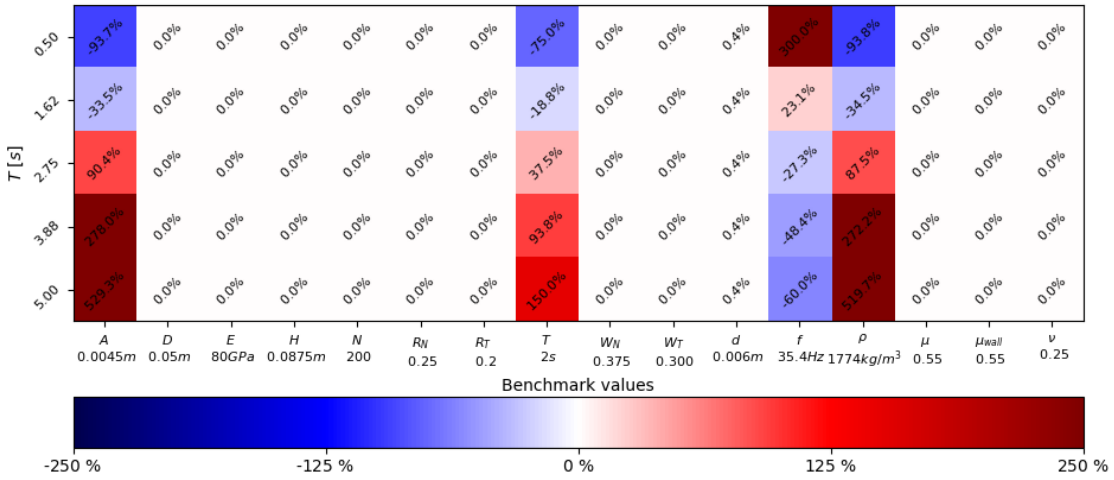


Figure 4.3: Variable adjustment; Blending time independence

In the results that Osorio and Muzzio, 2015 presented, one of the sensitive variables identified was the blending time, where the blending time is adjusted to keep the frequency constant. As a result, it can be seen that the blending time does affect the mixing results.

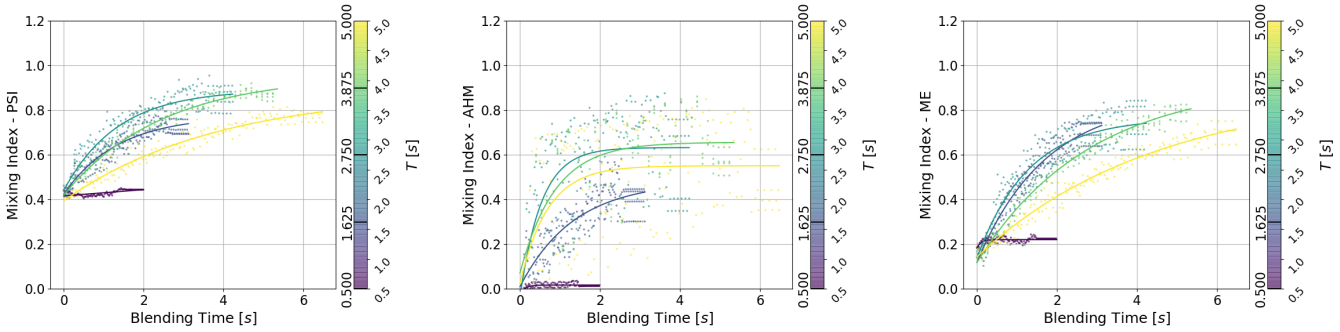
Here a similar trend can be seen as demonstrated in Figure 4.4. As the blending time increases, the mixing results are affected. However, the frequency is adjusted and as a result, it is also shown that, as the blend time increases even further, the mixing decreases. This might be due to the drop in frequency f.

For any of the Π_k variables to be declared independent, the results for each of the data points should align to a certain extent. In Figure 4.4 the blending time is varied from $T = 0.5s$ in blue to $T = 5.0s$ in yellow. As seen in Figure 4.4, the change in blending time T is not independent of change. The figures show that the increase in blend time increases the mixing performance across all mixing indexes.

On the other hand, in Figure 4.3 the decrease in blend time T increases the frequency f. The increase in frequency f and consequently decreasing the amplitude A to account for a balance in Π_2, causes the particles to barely move around. The results in Figure 4.4 show that the amplitude-frequency relationship has a larger impact on the mixing than that of the blend time. With that being said, the blend time was never expected to be independent of the Π_k groups, but rather an integral part of the mixing outcome.

This leads to another key external variable that influences the mixing behaviour - the frequency (f). As mentioned and shown in Figure A.1a this variable, when changed, influences the same variables as the blending time (T).

In Figures A.2a - A.2c similar behaviour is shown. As the frequency varies the mixing result changes. This leads to the conclusion that the frequency is not independent of the Π-group, rendering it not valid.



(a) Particle Scale Index (b) Average Height Method (c) Mixing Entropy

Figure 4.4: Blending time [1]

Note that the blending time on the horizontal axis reflects a longer time than that specified in the actual settings. This is due to the $0.75s$ acceleration and deceleration time respectively in the speed profiles as indicated in Figure 3.4b. The blending time used in Π_0 is the time that the mixture is agitated at the maximum intended frequency.

4.2.2 Primarily cup height and diameter variation subject to constant Π-group constraints

The cylindrical cup height and diameter influence two other variables, the particle diameter and the particle density. Figures A.1b and A.1c illustrate the variable changes over the domain. The cup height and diameter changes are linear to each other.

Π_1 was set up in such a way to represent the blending volume. Therefore, $\Pi_0 \geq 1$ then the cup is a tall and thin cylinder which allows the particles to move in an axial direction (along with the height of the cup). If $\Pi_0 < 1$, then the cup is short and thick, allowing the particles to move more freely in the radial direction. The other Π_k groups are set up in such a way to be adjusted accordingly and in this case, the particle diameter adjusts with the size of the cup - ensuring the fill volume (Π_3) is kept constant, causing the number of particle layers to remain unchanged. In this experimental case, there should be ∼8 layers of particles in the volume due to the circular packing order of the particles.

In Figures A.2d to A.2i, it is shown that when increasing the height or diameter by more than a 100% the Star-CCM+ solver struggles to compute the simulation. One of the reasons why it might struggle to compute the simulation is due to the decrease in particle density, making the particles too light for sufficient penetration using the Hertz-Mindlin model. In the manner in which Π_1 is set up it is still possible to draw valuable insights from limited results.

Due to the height and diameter having a linear relationship it is safe to say that similar observations can be made for the diameter and height independence. Due to complications in the model, some of the diameter data points did not run. Although it did not run it is safe to say that the same variables are being changed in the cases represented in Figures A.2d to A.2i. In all these cases the MI did not remain constant; hence Π_1 is not a valid group for RAM.

4.2.3 Amplitude variance

The amplitude, along with the frequency is identified as one of the key dimensionless groups, first identified by Osorio and Muzzio, 2015 and implemented by Resodyn in their LabRAM shakers (Section 1.2). The Π_2 group is also known as the mixing intensity. In this experiment, the amplitude ranges from $2mm$ to $20mm$.

Figure A.1d illustrates the changes that occur in the variables. As the amplitude is varied, the following additional variables are being changed to keep the Π groups stable - blending time, frequency and particle density. In Figure A.2j - A.2l one can see the effects of varying the amplitude, the mixing outcomes vary drastically over the domain. Figures A.2j to A.2l illustrates that, in none of the cases the MI remains constant and that the Π_2 is not a valid group for RAM.

4.2.4 Varying the particle count and diameter independently

When the particle diameter scales the number of particles (N) in the system has to adjust to ensure that Π_k remains constant - additionally, the particle density (ρ) has to adjust to compensate for particle volume increases to ensure the payload mass remains the same. Figure A.1e shows the variables changing alongside the particle diameters.

By unpacking the mixing indexes independently, some insights can be learned from each, namely:

- Figure A.2m: The Particle Scale index method shows that the variance over a period for each case is quite small compared to previous results in this chapter. It is also shown that the final blending state is similar for each case.
- Figure A.2n: The AHM indicates that the particle mixing varies significantly over the period domain. Each cases' variance is however small and an accurate curve can be fitted. This is due to how the mixing index is set up. This method uses averages to determine the mixing state of a blend and it is known that a smaller sample size could produce a less accurate measurement. In this case, as the particle diameter increases, the number of particles decrease causing the sample domain to be limited and the average height measurement to be skewed. Combined with the ME measure it can be concluded that this is indeed the case and the poor mixing is not due to a segregated sample.
- Figure A.2o: As mentioned above, ME measures indicate a similar trend as in the AHM, this is also due to the sample size. Due to the domain size (D & H) staying constant, the cell parameters will also stay constant. Similarly, in the manner in which ME is calculated, the inaccuracy increases as the sample size in each cell decreases.

In conclusion, if the particles were segregated, the PSI final measurement would show the same pattern of results which are demonstrated in the other measurements.

Although the results are fairly concurrent, it is evident that there is a slow decline in mixing rates as the particle diameter increases. This can be accounted for by the fact that the larger particles are restricting the mixing flow in the domain. It is important to note that the mixing over the particle size variance is slight (in the case of the PSI method) compared to other variables that the domain can be accounted as independent of particle size. There is some difficulties in investigating the lower end of the spectrum where $d < 4mm$. This is due to the number of particles in the domain and the simulation time that is needed to calculate the results. It has been mentioned previously in Chapter 3 that the particle count in the domain, from a simulation perspective, causes a hurdle for the solver.

Looking at the particle count specifically, the same variables will be varied as above. Considering the problem being faced in this study, where the simulation power that is needed to investigate the sweep is not being supported by the current platforms. However, inferring from the data a similar trend will show up when investigating the particle diameter. Therefore it is possible to infer that the amount of particles, along with the particle diameter, has little influence on the MI. As a result, it is still difficult to declare Π_3 as a valid group for RAM. However, this will be discussed in conjunction with the experimental results in Chapter 5.

4.2.5 Particle mechanical properties - density, elasticity and Poisson variance

Each of these particle material properties may influence the mixing outcome. It is however important to note that from these properties there are two factors to consider as these variables have a large role to play in each of the categories, namely:

- The particle damping (β) referred to in this chapter as modelled in the Resodyn RAM process. This particle damping feeds into the following point which is:
- The influences on the Hertz-Mindlin model used to solve the particle impacts. The particle density provides the mass and the Poisson's ratio alongside the particle elasticity that provides the spring and damping behaviour of the model.

In Figures A.1f - A.1h it is shown that by varying the density, the elasticity will vary and visa-versa and as indicated earlier, Poisson's ratio is independent of any other variable.

From Figures A.2p - A.2u it is possible to assume that these two material properties (E & ρ) have a smaller role to play in the mixing behaviour, the true result, if any, and will be displayed where the Π-group will be varied. The PSI and ME methods carry a tight variance over the period thus showing that these variables have minimal impact on the mixing behaviour.

Looking at the results presented, it is important to note that there might be one of either of the following outcomes - one where the Hertz-Mindlin model is equated in such a way where the pay-off between the elasticity and density is balanced and has minimal effect on the mixing when the Π_4 is kept stable, or where the elasticity and density changes of the material has minimal to no effect on the mixing. This will be evident once the Π_4 is changed, and if it stays unchanged the assumption can be made that the material properties have a minimal impact on the mixing outcome.

Another aspect to consider is where the Poissons ratio provides some damping behaviour in the Hertz-Mindlin model, as shown by Koblar and Boltežar, 2016. The particles will experience a compression force when accelerated. The Poisson's ratio might play a role in the way the particles mix. This variable is already dimensionless and thus stands alone to form its own Π-group, Π_5.

We can see that the Poisson's ratio in Figures A.2v - A.2x has little influence on the MI. Deeming the Π group independent of RAM, because the Π group is made up of a single variable the sensitivity to the Π group is also negligible. This assumption does however not state that the damping is not influential in the mixing outcome; the particle elasticity has a part to play in the particle damping.

4.2.6 Normal and Tangential restitution coefficients

The reason for considering the restitution components are not only because restitution plays a large role in DEM, but how RAM agitates the particles. This method causes a lot of movement in the domain and the restitution might play a role in influencing the movement. The variable functions in such a way that the larger the restitution, the more the particle bounces back and thus can cause an impact with higher acceleration and velocity.

It is also important to note that restitution influences the force in which the particles interact with one another. The Hertz-Mindlin model, therefore, has to be impacted due to the particle momentum and force. From the previous sections, we see that the particle material properties may or may not play a role in the mixing of the particles in RAM. Investigating these variables and their impact on the particle behaviour may be an indication that the particle flow in RAM has a smaller impact on the mixing behaviour, such

that the interparticle forces are minimal and may prove that the mixing scheme is better suited for more sensitive particles, such as gunpowder and impact sensitive medicinal chemicals.

4.2.6.1 Particle Restitution

When considering these variables it is important to note that the normal restitution component is not necessarily in the agitation direction. The normal and tangential component is relative to each particle and the direction of impact on the particle is as referenced in Figure 1.6. With that being said, by varying each of the restitution variables as in Figures A.1i & A.1j there would be no effect on the other variables, other than the normal and tangential directions.

In Figures A.2y - A.2aa the change in the normal-direction restitution does not affect the mixing result across the three selected methods. The same can be said about the tangential-direction restitution results in Figures A.2ab - A.2ad. When scaling each of the varaibles the results remain fairly similar, this is because as you scale either of the variables the other one scales in line with the Π group, as shown in Figures A.1i and A.1i. This however does not mean that the variables do not influence the particle movement, but rather that the balance between the tangential and normal restitutions are unaffected.

As per the results in Figures A.1i and A.1j, the particle restitution components have minimal effect on the mixing behaviour when keeping Π_k steady and thus it is possible to assume that Π_6 is a valid group for RAM.

4.2.6.2 Wall restitution

The wall restitution has a slight difference in that the normal wall restitution of the base and cap of the cup is always parallel to the direction of agitation.

Unfortunately by varying the wall restitution component Star-CCM+ does not recognize the variance in the domain and thus returns the same result for each of the cases. This defect has been discussed in Chapter 3 where the subsequent solution is the same. The result remains inconclusive due to the lack of data.

4.2.7 Friction coefficients

The last two variables to consider are the friction coefficients between particles and the wall friction. Due to the fact that the friction coefficients make up their own Π groups, each of these variables can now not only be considered as independent of the non-dimensional groups, but also the effects of each of these variables are to be considered on the mixing result.

In Figures A.2ak - A.2am, it can be seen that the variance in the particle-particle friction of the mixing result is impacted. Although there is a difference in the various methods, a slight trend is shown when the friction increases and the mixing performance also increases. Albeit that the PSI method shows a less aggressive mixing increase, the other methods show that the result plateaus after $\mu > 0.65$. This might be the increase in friction between particles causes the flow to advance quicker, showing that lower friction may cause the particles to slip past each other and delay the mixing process.

Similarly, the results for the wall friction also follow the same type of trend. The slippier the surface the lower the mixing performance. This is evident in Figures A.2an - A.2ap, although in the case of the particle-wall interaction the mixing is more aggressive.

One of the key take-away points from this sweep is that the maximum physical range of the parameters was used to show the results. The friction variables are varied between $0 \geq \mu \geq 0.90$ which shows that

even with the large range the mixing performance is not impacted as much as other variables might have. Even though the materials being mixed are not always viable from a material properties perspective, the impact is significant on the mixing behaviour. This is because even in the slipperiest case where $\mu = 0$, the materials still mixed thoroughly.

4.2.8 Summary

There is still a lot to be learned from each of the variables above and further studies may be required to understand each of the variables influence on RAM. For now, it can be concluded that the variables that do change the outcome of the mixing performance and do not abide by the Π groups are not independent variables for RAM. By continuing with the study it is still possible to investigate the merit of the Π groups regardless. One of these groups is the Π_2 group which contain two variables (A & f) that have been proven not to be independent, but sources such as Osorio and Muzzio, 2015 implemented (*How RAM Mixes - Resodyn Acoustic Mixers*), indicated that this group has a significant role to play in the mixing performance of RAM.

So far there are a couple of Π_k groups that remain invariant of change to each of their variables. The following Π_k groups seem to be valid Π groups:

- Particle material properties - Π_4 and Π_5
- Restitution properties - Π_6 and Π_7

By varying each of these groups one will be able to investigate the sensitivity of each of these groups on the MI.

Additionally, material properties such as μ and μ_{wall} have been tested in the sensitivity study; these parameters are stand-alone particle groups. Upon investigation, the variables do influence the final mixed state; however, the influence is marginal when compared to other Π_k groups.

Furthermore, the cup volume Π_1 does not adequately describe the particle volume and is thus not considered to be a valid Π group. This can be rectified by either carefully selecting variables to describe the domain or to reposition the group to fulfil another need.

The particle fill level Π_3 will still need to undergo some investigation and stress testing. Unfortunately due to a simulation error, minimal results were populated for the particle count N variances. In the experimental results in Chapter 5 the influence on the particle count will be discussed in more detail.

4.3 Π_k variable sensitivity study

The following section will focus on the Π_k parameters and the influences they have on the mixing result. This is done by choosing a benchmark set of Π_k parameters for this application. Table 4.4 shows the benchmarked values chosen for the sensitivity study. The values are chosen similarly as above where the parameters are both valid material properties, and more or less a midway of the experiment for which it was designed. For instance, if Π_0 is varied between 70 and 230, the benchmarked value is set at 150 which are halfway between 70 and 230.

Table 4.4: Π_k benchmark values - Non-dimensional sweep

$\mathbf{\Pi_k}$	**Value**	**Equation**
Π_0	150	4.6
Π_1	1.25	4.7
Π_2	1.00	4.8
Π_3	0.16	4.9
$\log_{10}(\Pi_4)$	10.0	4.10
Π_5	0.28	4.11
Π_6	1.00	4.12
Π_7	1.00	4.13
Π_8	0.50	4.14
Π_9	0.50	4.15

4.3.1 Time, Frequency, and Mixing intensity (Π_0 & Π_2)

The first variable to look at is the correlation between the mixing performance of RAM concerning the Π_0 and Π_2. The Resodyn machines allow these variables to be controlled by the user. In Figure 4.5 to 4.6, we can see the variables associated with the varying of these variables; T, f, A and ρ. In the case of Π_0, the blending time is assumed constant at $T = 5s$ allowing the adjustment of f to compensate for the change in Π_0. The same for Π_2, the blending time and frequency variables were assumed constant allowing the amplitude A to vary to compensate for the change in f.

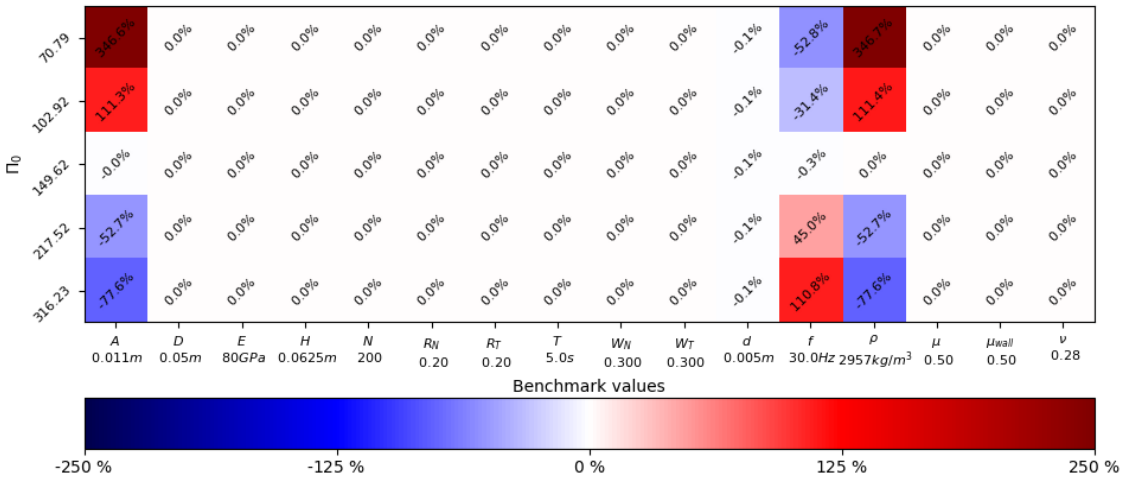


Figure 4.5: Variable adjustment - Π_0

Figures 4.7a to 4.7f show the experimental results for the changes in Π_0 and Π_2. Firstly, the Π_0 variable controls the blending time with regard to the frequency. Knowing that the mixing outcome is highly dependent on the T and f variables, makes it difficult to infer a result when varying Π_0. However, it is evident that the mixing increases in performance as the variable increases and deteriorates when the frequency increases too much. This was also shown earlier in the frequency study. This may indicate that there is an optimal frequency to mix particles in a RAM environment. In this case, the optimal frequency lies somewhere in the range of $30Hz$ to $44Hz$.

As referred to the mixing intensity by sources such as Osorio and Muzzio, 2015, the Π_2 variable plays a

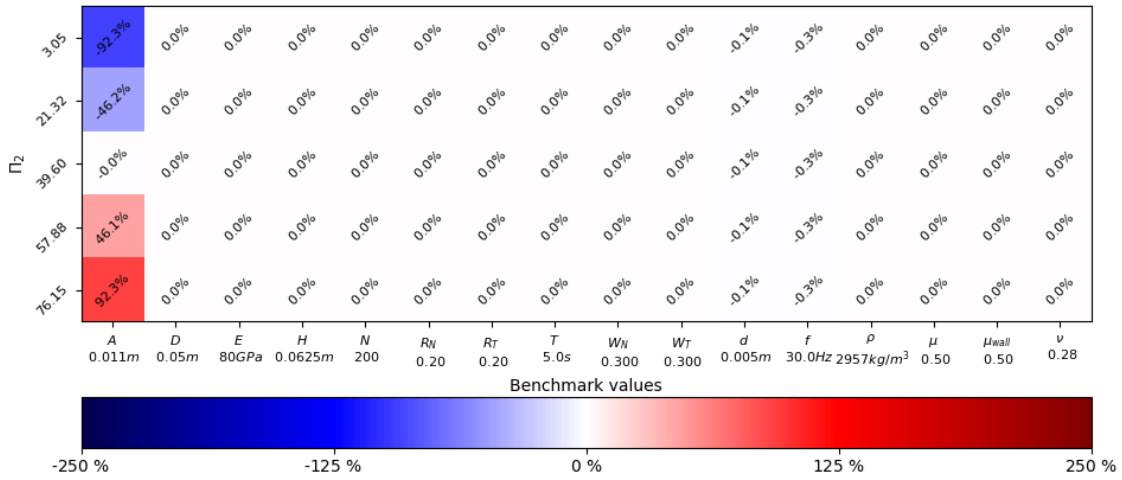
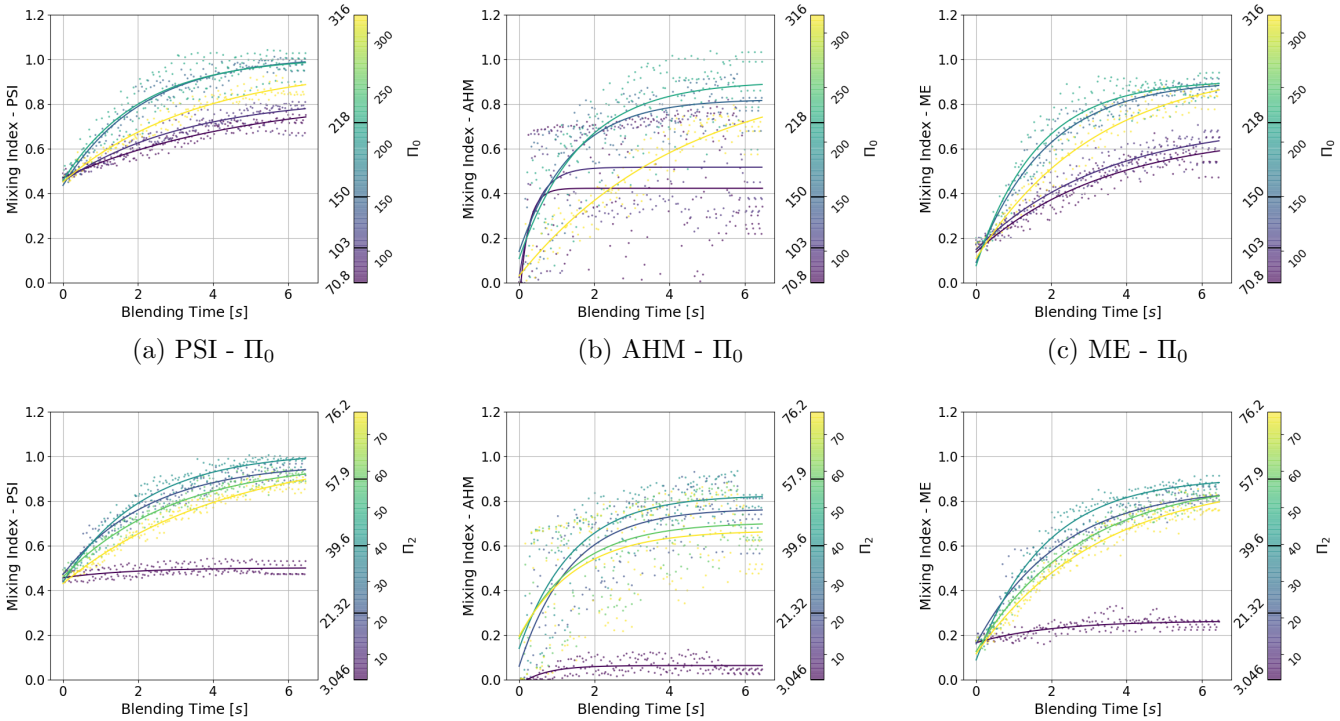


Figure 4.6: Variable adjustment - Π_2



(a) PSI - Π_0 (b) AHM - Π_0 (c) ME - Π_0

(d) PSI - Π_2 (e) AHM - Π_2 (f) ME - Π_2

Figure 4.7: Π sensitivity analysis - Π_0 and Π_2

vital role in RAM. As mentioned previously, this variable indicates the G-force applied to the mixing. This is typically measured in G's which is the relationship between the force applied by the system acceleration and the natural gravitational forces. This says that a $100G$ indicates an acceleration force on the system that is 100 times larger than that of the gravitational forces.

In the cases in Figures 4.7d to 4.7f one can see that the mixing intensity Π_2 has a significant influence on

the MI. However, it needs to be considered that the variables A and f are not independent of the mixing outcome. Regardless it seems like the mixing intensity has to be larger than 1 for it to have any effect on the mixing outcome.

4.3.2 Volumetric variables (Π_1 & Π_3)

Looking at the two variables Π_1 and Π_3 in Figures A.3a and A.3d, one can see that the change in variables is independent of the blending time (T) and frequency (f) - showing that the two variables Π_1 and Π_3 can be analysed without the time and frequency variables interfering with the results. The only variables that are dependent on the Π variables are:

- H - Cup height;
- d - Particle diameter;
- N - Number of particles; and
- ρ - Particle density

When taking into account that the H and D variables are not independent of the mixing outcome, the particle diameter, particle count and particle density are independent of the mixing outcome.

In the case where Π_1 is varied an interesting behaviour is observed; it is, therefore, necessary to look at both the PSI as well as the ME to understand this behaviour, and as shown in Figure A.4a the larger the base of the cup ($\Pi < 1$) the higher the initial mixing state. This is due to all the particles which are spread out across a wider surface area causing the particles to form less segregated layers and causing more neighbouring particles to be indifferent and in turn, increasing the mixing. It is also shown that the mixing rates vary slightly.

In conclusion, looking at Π_1 we can infer that the volume ratio impacts the rate of mixing more than the final state. It is necessary to keep in mind that a shallow cup ($\Pi_1 < 1$) will by definition be mixed well before any agitation takes place, but will take longer to mix even further. Cups with a larger volume ratio ($\Pi_1 > 1$) will start to mix faster, but will be less mixed from the initial state.

The fill level Π_3 relates the particle volume to the cup volume. As a result, if the cup is less full, the particles occupy less space in the domain and form less segregation layers. This, combined with the dependency of the particle position, the average height method results are not as reliable for this Π variable. It is then necessary to consider the PSI and mixing entropy methods as an accurate measurement.

This variable is one of the most sensitive Π variables in the system. The mixing is not only different in the final mixing state but also the rate of mixing differs between experiments. The initial state is dependent on the number of layers in the domain - similar to the Π_1 example. The mixing rate is quite sensitive as there is less space in the volume available for particles to flow freely past each other which throttles the mixing rate significantly. We can see in Figures A.4d to A.4f for each of the methods that the mixing results dramatically decrease as Π_3 increases.

4.3.3 Particle Material properties (Π_4 & Π_5)

In the current configuration, most of the particle material properties are captured in one Π variable: Π_4. This variable compares the elastic energy in each of the particle types to the kinetic energy that is being forced into the system by the frequency f. A logarithmic scale is used to map Π_4 to ensure comparable results.

Figure A.3c shows the two main variables that interchangeably balances the equation, namely, the particle density ρ and particle elasticity E. These variables are both independent and thus the behaviour shown in this section will be a true representation of the influence of Π_4 on RAM.

Furthermore, the actual mapping of the domain in Figures A.4g to A.4i suggests that the variable has a slight and almost insignificant impact on RAM. It can also be seen that there is a slight uptake in the mixing rate as Π_4 increases. If it is considered to rewriting the equation by the use of the speed of sound through the material, a variable which is defined by $V = \sqrt{\frac{E}{\rho}}$, then the following equation can be written:

$$\Pi_4 = \frac{V^2}{d^2 f^2} \tag{4.16}$$

Considering a material like a diamond, which is notorious for having one of the highest speeds of all commonly known materials, at $V \sim 12000 m/s$ Figure 4.8 shows the range of changing the particle diameter as well as the upper and lower boundaries of the frequency experimented within this study and the effect on the Π_4 variable. It is shown that the smaller the particle diameter the larger the Π variable, indicating a possible increase in the mixing rate. Even in this case where V is considered high, the particle diameter has to be in the μm range to show an increase in mixing performance. This Π group can be deemed as having an insignificant impact on RAM.

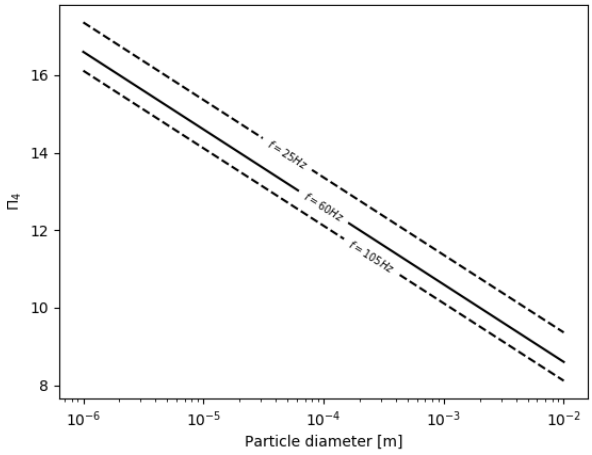


Figure 4.8: Π_4 realistic parameters

The particle Poisons ratio ν or Π_5 were investigated in Section 4.2.5 - where the conclusion was made that the variable also has little to no effect on RAM.

4.3.4 Particle and Wall restitution groups (Π_6 & Π_7)

In both variables[2], the particle and wall restitution variables, the Normal restitution is used to balance the Π variables to the desired value, keeping the tangential component constant. This is illustrated in Figure A.3d.

Looking at the results in Figures A.4j and A.4l, the mixing outcome does not change significantly over the domain of Π_6. It can be seen that the mixing rates slightly increase over the domain. This behaviour is

[2]Note that the Wall restitution variable fails in the StarCCM+ modelling and will have one result across the sweep. For this section the wall variable will be ignored.

quite interesting in the Hertz-Mindlin model which means that particles with a higher normal restitution increase the mixing rate. Particles that bounce off each other easier would promote the rate at which the mixing occurs. This is however a difficult variable to control, as most materials have a set restitution variable.

The ranges of the Π_6 variable must be positive to make it possible for the mixing rate to increase even more when $\Pi_6 >> 2$. However, the rate change is slow over the domain and one has to consider how realistic these variables are. It might be worth to keeping in mind that this behaviour is possible despite being improbable.

4.4 Reflection and future

Although a lot has been covered in the last two sections of this chapter, it is valuable to note that there is benefit in taking this study further to understand the effects of some of the variables that might have an impact on the mixing outcome. It will be beneficial to gain some insights into a higher resolution testing of the domain, possibly decoupling some of the dependent variables from the respective Π variables and letting them function as a variable on their own. In the sensitivity study it is also evident that the following variables do not have a significant impact on the mixing outcomes, namely;

- Π_4 - The material properties hold little to no significance in the performance of the mixing outcome. This means, for simulation purposes one may assume a lower value for E, speeding up the simulation time with softer particles, without forgoing mixing performance.
- Π_5 - The Poissons ratio does not affect the mixing outcome.
- Π_6 and Π_7 - The restitution parameters of smooth spherical particles at low values holds little significance in the mixing outcome.

As demonstrated, the following Π variables did carry some significance in the RAM domain and were identified to be tested experimentally.

- Π_1 - The volume ratio of the cup significantly impacts the outcome of mixing and can easily be tested in an experimental environment.
- Π_2 - The acceleration variable was identified by reputable sources, as well as this study, to have an influence on the mixing outcome.
- Π_3 - The particle fill level is similarly significant and can also easily be tested in an experimental setup.

These experiments serve as a verification method of what has been discovered in this chapter and show that the numerical results may be considered as accurate and correct.

Chapter 5

Experimental analysis

One of the main concerns when addressing a problem such as RAM is the validity between experimental results and numerical computation. There is some uncertainty regarding the random stochastic motion of particles suspended in the air and the validity of the contact model used in the numerical computation. The aim of this experimental study is then to test the accuracy of assuming a Hertz-Mindlin contact model.

Using these three influential variables Π_1, Π_2 and Π_3, it should be clear if these parameters are yielding consistent results in the numerical experiments and help to gain further insight into the sensitivity of the variables.

5.1 Experimental setup

5.1.1 Particle Shaker

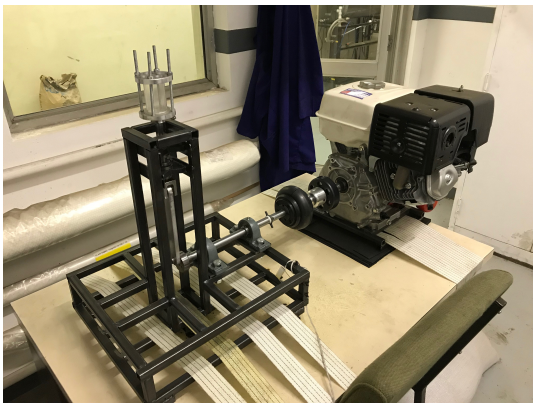
For the setup, a large amplitude shaker was constructed which can shake $100g$ of particles at an acceleration of $109Gs$. The experimental setup was designed, constructed and commissioned as part of this study at the University of Pretoria.

The shaker is designed in a piston or crank and slider fashion, allowing the user to manually adjust variables such as rotational speed and amplitude. Due to budget and design restrictions, a combustion engine and centrifugal clutch were selected to drive the experimental setup. This makes it difficult to control using basic equipment and reputable results. Using an Arduino and a speed sensor, it is possible to record and display the data on the screen to hand-adjust the accelerator.

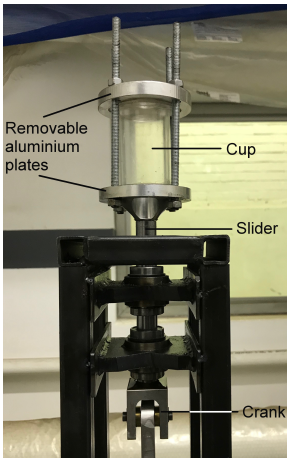
The amplitudes are adjustable to different settings ($2.5mm$, $5mm$, $7.5mm$, $10mm$, $12.5mm$ and $15mm$) and the motor has a top speed of $3600RPM$. The maximum cup size is currently limited to a $50mm$ outside diameter with an adjustable cup height. The cup is clamped between two removable aluminium plates which can also be modified to the cup specification. The clutch gives the user the ability to start the motor and ramp up the accelerator to the desired speed without shaking the cup whilst the motor idles. Releasing the clutch starts rotating the axle and starts the mixing process.

This was one of the more cost-effective methods of achieving the needed results. All parts were manufactured in house and are made from high-quality building materials ensuring a safe working environment.

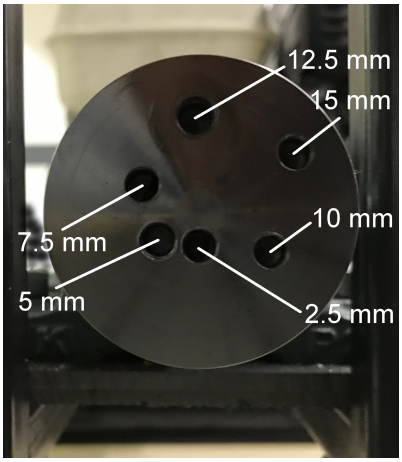
Currently, the shaker (Fig. 5.1) is installed and in operation at the University of Pretoria as part of the Composite Material laboratory.



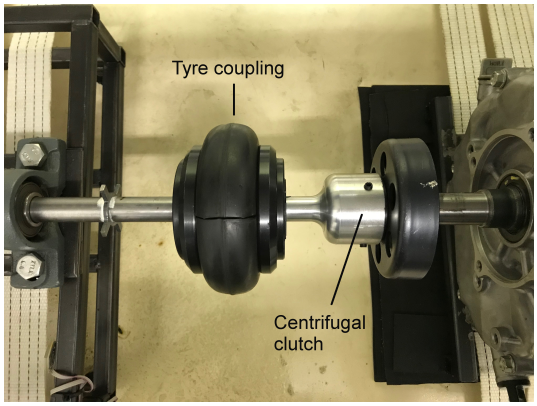
(a) Shaker setup



(b) Cup installation



(c) Amplitude adjustment



(d) Motor coupling and speed sensor

Figure 5.1: Experimental setup

For comparable results, large particles were selected in different colours to easily distinguish particles. Other than the colours being different the particles are identical. The particles that were used were $6mm$ BB bullets. The reason for this choice was that these particles have a uniform shape and diameter that exists in multiple colours.

For the final result to remain in situ, a diluted casting resin was used to encapsulate the particles. The resin was cast and placed in a vacuum chamber ensuring all areas were filled without void. Figure 5.2 shows the procedure that was followed to mix one sample and Table 5.1 shows the particle properties of the BB bullets.

5.1.2 Interpretation and preparation of results

To ensure continuity throughout the study it is needed to compute the same primary mixing indexes that were used in the numerical analysis. This will help to draw meaningful comparisons and conclusions. The challenge arises when measuring transient mixing in a practical environment. This is unfeasible without the use of high-speed photography - this was not considered for this study. The final mixed state can however be calculated as described below.

Property	**Dimension**	Π_k **benchmark** (Table 4.4)
Description	BB-pellet	-
Material	High Impact Polystyrene	-
Blending time	$5s$	$6s$
Cup ratio - Π_1	0.50 - 2.00	1.25
Cup acceleration - Π_2	1.00 - 13.87	1.00
Particle fill level - Π_3	0.20 - 0.40	0.16
Material properties - $\log_{10}(\Pi_4)$	5.40 - 6.20	10.00
Diameter	$6mm$	-
Density	$1.430g/cm^3$	-
Youngs' Modulus	$2.22GPa$	-
Weight per particle	$0.12g$	-
Frequency	$19Hz$ - $46Hz$	-
Poissons ratio (Π_5)	0.30	0.28
Restitution coefficient (Π_6 & Π_7)	0.80	1.00
Friction coefficient (Π_8 & Π_9)	0.20	0.50

Table 5.1: Experimental properties parameters

To analyse the results, a series of photographs were taken sequentially by rotating the cup in small increments at each step. From each of the photos, a slice was taken and stitched together to make up one large photograph. This was done using a DSLR camera with a fixed focal point and a rotary table which is controlled by an Arduino controller board. Figure 5.3 shows the result of such a photo. As seen in Figure 5.3 afterwards a hexagonal structure is overlaid on tothe photo that best fits the particle structure in the photo. Each one of these hexagonal cells is then classified using a Support Vector Machine with a Linear Kernel to produce the image in Figure 5.3. This result is a fair representation with accurate positional readings of each experimental result, which allows for consistent and accurate mixing index measurements.

This method enabled the calculation of the surface mixing of the experiments. Furthermore, this allowed one to compare the results of the Numerical experiments to that of the Experimental results. If one had to calculate the internal mixing of these experiments one would be faced with two issues.

1. The destruction of the sample and removal of quite a lot of material using a saw blade. This would compromise the final results.
2. Adding the dimension would not significantly change the number of visible particles. This is due to the large particle sizes in the smaller diameter cups.

Additionally, the results show a small discrepancy between the numerical and experimental measures and thus, for this part of the study, it is safe to conclude that looking at the surface mixing would be enough to verify that the numerical trends correlate with the experimental trends.

The results were calculated for each sample and are included in the appendix along with the images used to analyse the results. Any interesting behaviour and images are included in the report.

5.2 Experimental results

For the experimental analysis, the three main Π groups were investigated. The choices of each are based on readily available and uniform particles - the BB-pellets is a cheap and readily available source that is

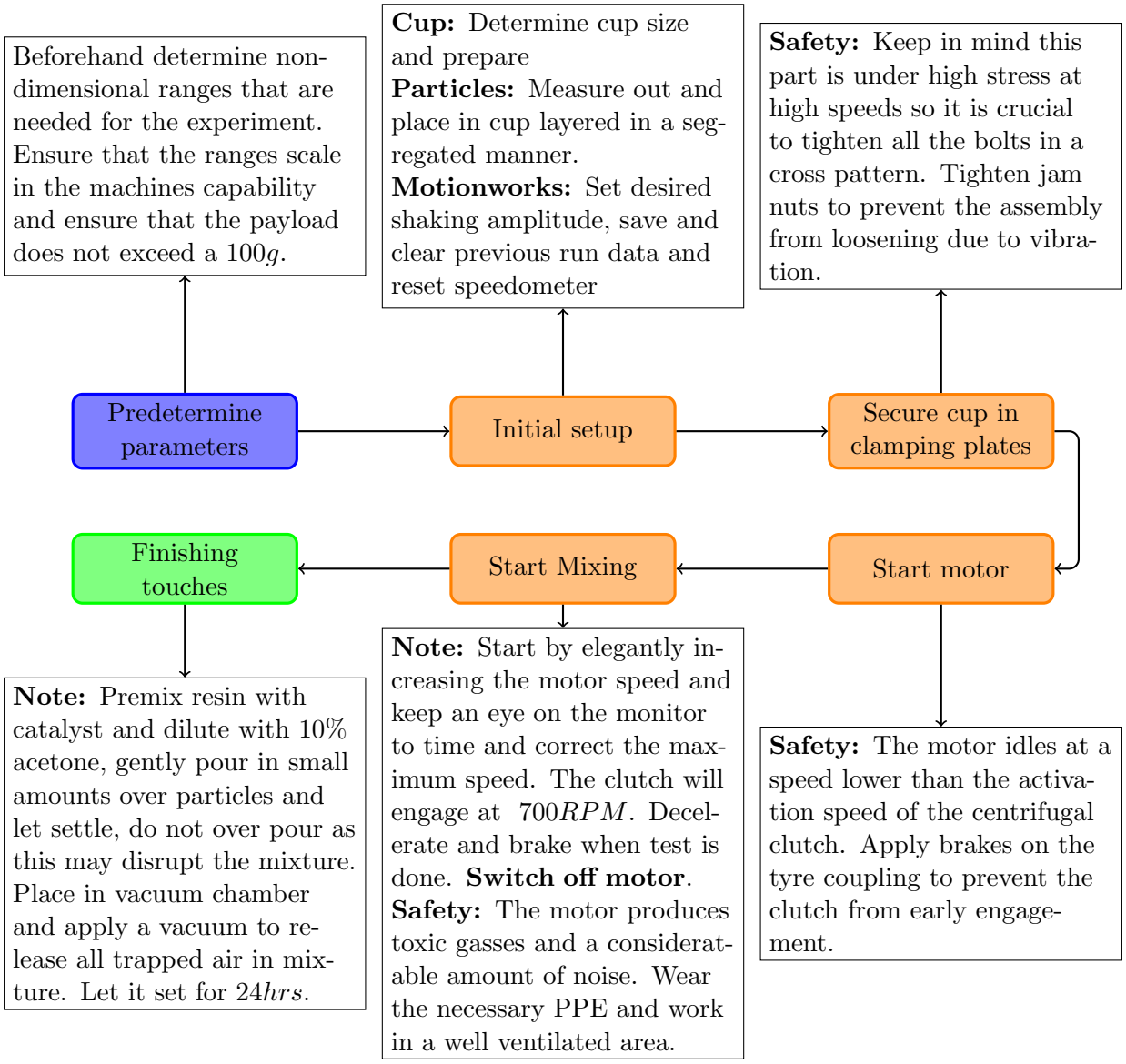


Figure 5.2: Experimental procedure

available in a variety of colours as well as the mechanical limits to which the shaker can withstand.

Each of the Π-groups below will be analysed in both numerical and experimental form;

- Experimental results - this shows the mixing outcomes of the actual experiments.
- Numerical results - this represents the same variable changes that were done in the numerical study.

Ideally, the numerical and experimental variables should match up. Due to running out of computational time at the CHPC and fixed particle choices on the experimental side, the closest match to the experimental variables was chosen and displayed in each graph.

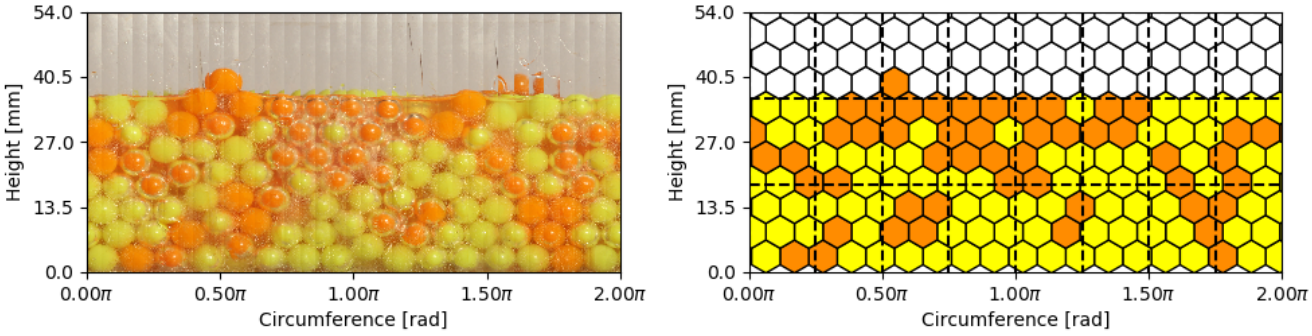


Figure 5.3: Experimental Example

5.2.1 Particle Fill level - Π_3 experimental results

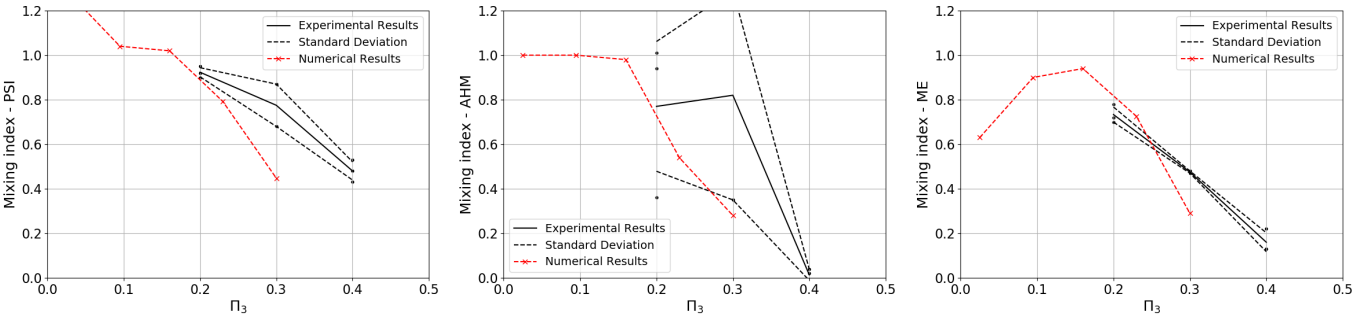
As noted in the previous chapter, there is one Π-group that displays a distinction in its influence on the mixing outcome. The fill level or Π_3 variable demonstrate the volume occupied by particles in the available space. This is calculated per particle type.

Figure 5.4 displays the impact of the change in the fill level with each of the mixing indexes. In each of these examples the experimental and numerical results are shown.

The known mixing index issues show, once again where the average height method has a large variance in results. The mixing entropy fits well on the results and shows the behaviour as expected - which is; as the fill level increases the mixing performance decreases. As $\Pi_3 \to 0$. As the mixing moves closer to $M = 1$, this effect has been described earlier in the report as: "...the number of particle layers decreases the more the mixing increases". The PSI is slightly over calculated, which might be due to the low sample size or variances in the model, as seen in Figure 5.4a.

In this variable study, the behaviour remains very similar between the numerical and experimental results. This shows that how the numerical results are calculated in the commercially available packages, using a Hertz-Mindlin contact model, is a fair representation of how particles flow in RAM.

Figures B.5a - B.7c shows the results used to calculate the mixing performance for this experiment.



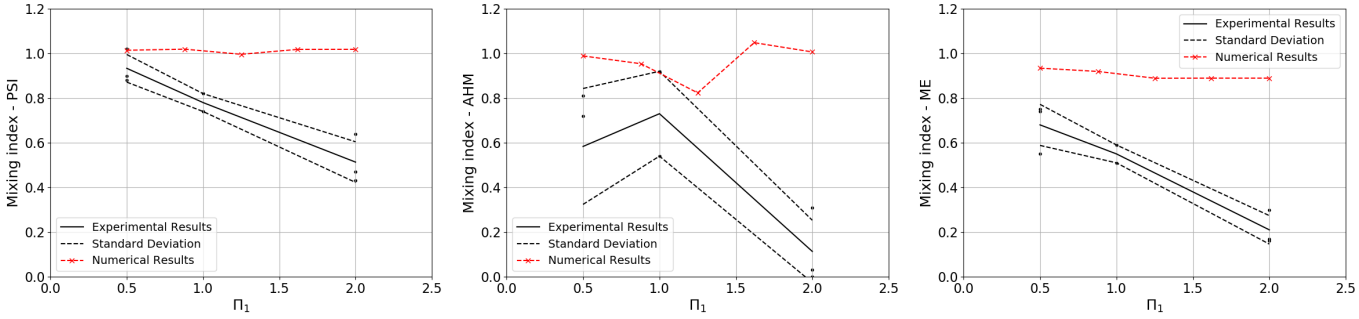
(a) PSI - Π_3 (b) AHM - Π_3 (c) ME - Π_3

Figure 5.4: Experimental Π_3 results

5.2.2 Cup Volume or Π_1 experimental results

The relationship between the cup height and the diameter as seen in the previous chapters are quite involved in the performance of RAM. The number of particle layers in the domain affects the MI. One can observe in Figures B.8a - B.10c that the layers in each of the samples vary as the Π_1 parameter increases.

Figure 5.5 shows that the average mixing behaviour for these samples decreases as the Π_1 parameter increase, however with a low sample size this behaviour is difficult to predict. In this case, the numerical and experimental results do not match up. This is quite interesting behaviour as there is a definite change in the mixing from an experimental point of view.



(a) PSI - Π_1 (b) AHM - Π_1 (c) ME - Π_1

Figure 5.5: Experimental Π_1 results

Experimentally, the decline in mixing is due to the dependency on the number of particles in the cup N. The only way to ensure that all Π variables remain the same throughout the experimental study is to vary the number of particles in the cup, as the cup dimensions change and the particle diameters stay the same. In the numerical study, if the cup dimensions are varied the particle diameter is adjusted to accommodate for the change; therefore, in conclusion, the number of particles in the cup has a more significant impact than changing the cup ratios.

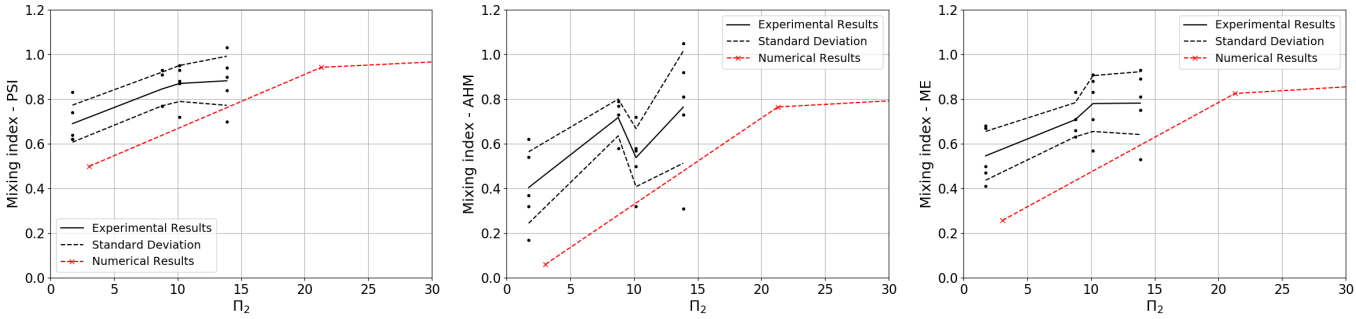
Ideally, one would have liked to vary the Π-groups numerically with the experimental groups; however, it was not feasible due to computing constraints at the CHPC.

5.2.3 Acceleration or Π_2 experimental results

The final experimental result to consider is the acceleration parameter Π_2, where a combination for A and f is varied to get the respective Π_2 values.

Figure 5.6 show these results - like the mixing increases, the Π_2 variable increases. This is concurrent with the numerical trends. The PSI again has a higher MI in the experimental samples, which can be attributed to the lower coordination number in the surface mixing case.

As expected, the AHM indicates a large variance in the captured data. However, the numerical results follow the same trend. The mixing entropy shows that the experimental results indicate similar outcomes across the numerical and experimental results.



(a) Particle Scale Index (b) Average Height Method (c) Mixing Entropy

Figure 5.6: Experimental Π_2 sweep

5.3 Summary

When doing experimental studies one strives to create near-perfect conditions to mimic the numerical experiment. However, the aspect of human error and the rigorous experimental setup can attribute to some inconsistent results. Here one observes that the results have a variance on each tested point throughout the domain. Despite all of the inconsistencies, it is still evident that there is a correlation in the experimental and numerical domains.

Although there are inconsistencies in the data there are a few things that stood out in this chapter, namely:

- The numerical and experimental results follow the same trend, proving that the numerical approach to modelling RAM with the Hertz-Mindlin model can provide accurate results and insights when modelling RAM.
- In the case of Π_1 where the values do not correlate this can be mitigated in the future by changing the number of particles in the numerical domain instead of the particle diameter. This attests to the findings made in Section 4.3.2 on the Π group being invalid due to the dependencies on each of the variables in the group.
- In these cases where the numerical trends align with the experimental trends, one can see that the Π-groups are sensitive to the MI. These groups are the Particle Fill level Π_3 and mixing intensity Π_2 parameters.
- Although the Π_2 and Π_3 are not valid Π groups due to the frequency f, amplitude A, particle diameter d_n, particle count N_n, cup height H and diameter D, being independent of the Π groups; the Π groups can be used as a proxy when calculating the MI of a RAM experiment. This is due to two factors:
    1. Previously identified sensitive parameters as in Osorio and Muzzio, 2015 have proven that the group can be used to validate results.
    2. The close correlation between trends in the numerical and experimental results can be accepted, although there was noise introduced in the form of human error, as well as the discrepancies in the initial values as described in Table 5.1.

Conclusion

The study aimed to gain a better understanding of how the specific phenomenon called - Resonant Acoustic Mixing works, and which parameters and variables influence the mixing behaviour in a cylindrical domain with spherical particles. The study consisted of many different branches of research from understanding how the mixing indexes are designed through to particle physics as well as a variable and sensitivity study to identify what parameters to test experimentally and verify data.

As a result, it was possible to identify the sensitive parameters in the problem. Out of the parameters identified in the non-dimensional study, four major non-dimensional variables were identified that influenced the mixing outcomes with a significant effect - the following variables were identified:

- The cup fill level,

$$\Pi_3 = \frac{d^3 N}{D^2 H} \tag{5.1}$$

- The cup acceleration,

$$\Pi_2 = \frac{A f^2}{g} \tag{5.2}$$

- The height to diameter ratio of the mixing volume,

$$\Pi_1 = \frac{H}{D} \tag{5.3}$$

To support the information above, additional research was done and implemented in the study to ensure the success of the results presented here. The following additional information was used to support the findings:

- Particle contact model research and understanding.
- Particle mixing indexes and additional methods to quantify mixing.
- Computational methods and numerical simulations to ensure large volumes of information can be processed in a reasonable amount of time.
- Experimental results to confirm findings in the numerical experiments.

The cup fill level

This variable is characterized by considering each particle type and the physical volume it occupies in the domain provided by the mixing volume. This parameter looks at the following variables; d - the mean

particle diameter, N - the number of particles in the volume, D - the cup volume base diameter and H - the cup height.

When looking at the mixing result when varying the cup fill level from 0% to 100% full, the results should fit a hyperbolic tangent curve. It starts to decrease as the volume fills up with particles and remains unmixed if the cup is full. This is because the cup is too full and the particles are inhibited in movement and unable to mix.

On the other end of the spectrum, if the cup is not filled up at all, the steady-state mixing will converge to $M = 1$. In this study, the threshold at which the steady-state mixing deteriorates below $M = 0.50$ mixed is when the volume is between 55% and 60% filled.

One of the other parameters that play a major role in the mixing outcome is the acceleration of the mixing volume.

Cup acceleration

In some commercial environments, this parameter is referred to as the mixing intensity. This non-dimensional variable consists of two variables that influence the mixing outcome, f - the frequency and A - the amplitude. This expression is based on the relationship between the acceleration provided by the frequency of the particles and the ability to overcome the gravitational forces on the particles.

In this study, the individual parameters in each of these equations were investigated to identify the sensitivity of each variable to the mixing outcome. In this specific case, the amplitude variable has proven to be sensitive to amplitudes of less than $7.5mm$. The frequency domain is also sensitive to variation. However undeniable that these two parameters do play a major role in the mixing behaviour as it is the only kinetic energy input to the system.

Height to diameter ratio

This parameter gave some insight as to whether the mixing occurs from impulses between the top and bottom surfaces of the mixing volume, or if the mixing occurs to form a natural particle flow when the particles are suspended in the domain.

Numerically the parameter remains unaffected; however, the experimental results show a decline in mixing performance as the variable increases. The variable is depended on the cup height (H) and the cup diameter (D).

The experimental results from this variable, however, show that the mixing of particles is affected when the height to diameter ratio is changed. This result demonstrates the dependency on the number of particles in the system and not necessarily the particle sizes. Additionally, assuming the fill level is kept constant, as the height decreases the particle mixing deteriorates when particle movement is restricted in the mixing axis. On the other hand, once the height increases infinitely the particles have nowhere to induce an impulse from the top surface and the rate of mixing is significantly impacted.

Future work

Looking into the variables and parameters of RAM assisted one in gaining a greater perspective on how the mixing of particles evolve when they are subjected to large amplitude shaking. The study showed there

is still an opportunity for further investigations that can assist in understanding RAM better. Here is a few examples that can be investigated in future studies.

- Particle agglomeration and separation abilities - RAM is a renowned method for mixing viscous materials. Due to a minimal energy transfer in-between particles in RAM (Section 1.1), it will be interesting to see the ability of RAM to mix cohesive particles.
- Understanding the mixing performance of RAM in different shaped vessels.
- Using GPU based computing methods (Govender, Wilke, Pizette, et al., 2017) to scale the sample size and extend the sensitivity study.
- Expanding on the sensitivity analysis by diving deep into the variables that affect the mixing behaviour.
- Verify experimental transient mixing states with numerical simulations with the help of high-speed photography.

Bibliography

1. Akao, Yoji et al. (Nov. 1976). "Estimation of mixing index and contact number by spot sampling". In: *Powder Technology* 15.2, pp. 207–214. DOI: 10.1016/0032-5910(76)80050-2. URL: https://doi.org/10.1016/0032-5910(76)80050-2.
2. Arntz, M. M. H. D. et al. (Dec. 2008). "Granular mixing and segregation in a horizontal rotating drum: A simulation study on the impact of rotational speed and fill level". In: *AIChE Journal* 54.12, pp. 3133–3146. DOI: 10.1002/aic.11622. URL: https://doi.org/10.1002/aic.11622.
3. Board, M. (Aug. 1989). *Examination of the use of continuum versus discontinuum models for design and performance assessment for the Yucca Mountain site*. Tech. rep. DOI: 10.2172/137516. URL: https://doi.org/10.2172/137516.
4. Buckingham, E. (Oct. 1914). "On Physically Similar Systems; Illustrations of the Use of Dimensional Equations". In: *Physical Review* 4.4, pp. 345–376. DOI: 10.1103/physrev.4.345. URL: https://doi.org/10.1103/physrev.4.345.
5. Chandratilleke, G. R. et al. (May 2011). "A particle-scale index in the quantification of mixing of particles". In: *AIChE Journal* 58.4, pp. 1099–1118. DOI: 10.1002/aic.12654. URL: https://doi.org/10.1002/aic.12654.
6. Chaudhuri, Bodhisattwa, Muzzio, Fernando J., and Tomassone, M. Silvina (Oct. 2006). "Modeling of heat transfer in granular flow in rotating vessels". In: *Chemical Engineering Science* 61.19, pp. 6348–6360. DOI: 10.1016/j.ces.2006.05.034. URL: https://doi.org/10.1016/j.ces.2006.05.034.
7. Chou, Shih-Hao, Song, Yue-Lou, and Hsiau, Shu-San (2017). "A Study of the Mixing Index in Solid Particles". In: *KONA Powder and Particle Journal* 34, 275–281. DOI: 10.14356/kona.2017018.
8. Clark, J. Peter (2009). "Dry Mixing". In: *Food Engineering Series*. Springer New York, pp. 5–15. DOI: 10.1007/978-1-4419-0420-1_2. URL: https://doi.org/10.1007/978-1-4419-0420-1_2.
9. Cundall, P. A. and Strack, O. D. L. (Mar. 1979). "A discrete numerical model for granular assemblies". In: *Géotechnique* 29.1, pp. 47–65. DOI: 10.1680/geot.1979.29.1.47. URL: https://doi.org/10.1680/geot.1979.29.1.47.
10. DEM Solutions Ltd. (2016). *EDEM 2017 User Guide*. DEM Solutions Ltd. Edinburgh, Scotland, UK.
11. Directory, IQS (Mar. 24, 2020). *Bulk Material Handling Equipment & System Manufacturers. Accessed on:* 2020-04-13. URL: https://www.iqsdirectory.com/bulk-handling-and-equipment-systems/#mixers-and-blenders.
12. Ebert, J. (Sept. 2014). "Sealing ability of different versions of GuttaFlow2 in comparison to GuttaFlow and AH Plus". In: *RSBO* 2014, pp. 224–9.

13. Etsion, Izhak (Apr. 2010). "Revisiting the Cattaneo–Mindlin Concept of Interfacial Slip in Tangentially Loaded Compliant Bodies". In: *Journal of Tribology* 132.2. DOI: 10.1115/1.4001238. URL: https://doi.org/10.1115/1.4001238.

14. Godlieb, W. et al. (2010). "DEM and TFM simulations of solids mixing in gas-solid fluidized bed". English. In: *Proceedings of the 7th International Conference on Computational Fluid Dynamics in the Minerals and Process Industries (CFD 2009), 9-11 December 2009, Melbourne, Australia*. Australia: CSIRO, 097GOD–1/7. ISBN: 978-0-643-098.

15. Govender, Nicolin et al. (2017). "BlazeDEM3D-GPU A Large Scale DEM simulation code for GPUs". In: *EPJ Web of Conferences* 140, p. 06025. DOI: 10.1051/epjconf/201714006025.

16. Gupta, Govind S. et al. (2014). "Process Concept for Scaling-Up and Plant Studies". In: *Treatise on Process Metallurgy*. Elsevier, pp. 1100–1144. DOI: 10.1016/b978-0-08-096988-6.00040-7. URL: https://doi.org/10.1016/b978-0-08-096988-6.00040-7.

17. Hertz, H. (1881). "On the contact of elastic solids". In: *Journal für die reine und angewandte Mathematik* 92, pp. 156–171.

18. Jaeger, J. (Feb. 1994). "Analytical Solutions of Contact Impact Problems". In: *Applied Mechanics Reviews* 47.2, pp. 35–54. DOI: 10.1115/1.3111070. URL: https://doi.org/10.1115/1.3111070.

19. Kenneth Langstreth Johnson, Kevin Kendall and Roberts, A. D. (Sept. 1971). "Surface energy and the contact of elastic solids". In: *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences* 324.1558, pp. 301–313. DOI: 10.1098/rspa.1971.0141. URL: https://doi.org/10.1098/rspa.1971.0141.

20. Koblar, D. and Boltežar, M. (Feb. 2016). "Evaluation of the Frequency-Dependent Young's Modulus and Damping Factor of Rubber from Experiment and Their Implementation in a Finite-Element Analysis". In: *Experimental Techniques* 40.1, pp. 235–244. DOI: 10.1007/s40799-016-0027-7. URL: https://doi.org/10.1007/s40799-016-0027-7.

21. labcompare. *Laboratory Platform Shaker — Labcompare.com. Accessed on:* 2020-04-13. URL: https://www.labcompare.com/General-Laboratory-Equipment/222-Laboratory-Platform-Shaker/.

22. Lacey, P. M. C. (May 2007). "Developments in the theory of particle mixing". In: *Journal of Applied Chemistry* 4.5, pp. 257–268. DOI: 10.1002/jctb.5010040504. URL: https://doi.org/10.1002/jctb.5010040504.

23. Luding, Stefan (Aug. 2008). "Introduction to discrete element methods". In: *European Journal of Environmental and Civil Engineering* 12.7-8, pp. 785–826. DOI: 10.1080/19648189.2008.9693050. URL: https://doi.org/10.1080/19648189.2008.9693050.

24. Mann, David (June 11, 2019). *STAR-CCM+ v12.04 preview: Model realistic particle shapes with polyhedral DEM particles — MDX. Accessed on:* 2020-04-13. URL: http://mdx2.plm.automation.siemens.com/zh-hans/node/9717.

25. Mindlin, R. D. (1989). "Compliance of Elastic Bodies in Contact". In: *The Collected Papers of Raymond D. Mindlin Volume I*. Springer New York, pp. 197–206. DOI: 10.1007/978-1-4613-8865-4_24. URL: https://doi.org/10.1007/978-1-4613-8865-4_24.

26. Osorio, Juan G. and Muzzio, Fernando J. (July 2015). "Evaluation of resonant acoustic mixing performance". In: *Powder Technology* 278, pp. 46–56. DOI: 10.1016/j.powtec.2015.02.033. URL: https://doi.org/10.1016/j.powtec.2015.02.033.

27. Potyondy, D.O. and Cundall, P.A. (Dec. 2004). "A bonded-particle model for rock". In: *International Journal of Rock Mechanics and Mining Sciences* 41.8, pp. 1329–1364. DOI: 10.1016/j.ijrmms.2004.09.011. URL: https://doi.org/10.1016/j.ijrmms.2004.09.011.

28. Rao, Singiresu and Nguyen, V. T. T (Jan. 2011). "Basic Mechanical Vibrations". In: p. 271. ISBN: 978-0-13-212819-3.

29. Resodyn Acoustic Mixers. *How RAM Mixes - Resodyn Acoustic Mixers. Accessed on:* 2020-04-13. URL: https://resodynmixers.com/how-ram-mixes/.

30. — *LabRAM I - Resodyn Acoustic Mixers. Accessed on:* 2020-04-13. URL: https://resodynmixers.com/products/labram-1/.

31. Rosato, Anthony et al. (Mar. 1987). "Why the Brazil nuts are on top: Size segregation of particulate matter by shaking". In: *Physical Review Letters* 58.10, pp. 1038–1040. DOI: 10.1103/physrevlett.58.1038. URL: https://doi.org/10.1103/physrevlett.58.1038.

32. Siemens PLM Software (2017). *STAR-CCM+ Documentation*. English. Version Version 12.02. Siemens PLM Software, p. 3729. 11243 pp.

33. Siiriä, Simo and Yliruusi, Jouko (Dec. 2009). "Determining a value for mixing: Mixing degree". In: *Powder Technology* 196.3, pp. 309–317. DOI: 10.1016/j.powtec.2009.08.009. URL: https://doi.org/10.1016/j.powtec.2009.08.009.

34. Sonin, Ain A. (2001). *The Physical Basis of Dimensional Analysis*. Ed. by Ain A. Sonin. Second. Cambridge, MA: Department of Mechanical Engineering, MIT.

35. Tasirin, S.M., Kamarudin, S.K., and Hweage, A.M.A. (May 2009). "Mixing Process of Binary Polymer Particles in Different Type of Mixers". In: *Modern Applied Science* 3.6. DOI: 10.5539/mas.v3n6p88. URL: https://doi.org/10.5539/mas.v3n6p88.

36. Toofan, Jahansooz (Feb. 1994). "A Simple Expression between Critical Radius Ratio and Coordination Number". In: *Journal of Chemical Education* 71.2, p. 147. DOI: 10.1021/ed071p147. URL: https://doi.org/10.1021/ed071p147.

37. Walton, O. R. and Braun, R. L. (Nov. 1986). "Stress calculations for assemblies of inelastic spheres in uniform shear". In: *Acta Mechanica* 63.1-4, pp. 73–86. DOI: 10.1007/bf01182541. URL: https://doi.org/10.1007/bf01182541.

38. Wassgren, Carl (2008). *PSL DEM Lecture 09: Tangential Contact Force Models*. URL: https://pharmahub.org/resources/253.

39. Wen, Yuanyun et al. (2015). "Comparative Study on the Characterization Method of Particle Mixing Index Using DEM Method". In: *Procedia Engineering* 102, pp. 1630–1642. DOI: 10.1016/j.proeng.2015.01.299. URL: https://doi.org/10.1016/j.proeng.2015.01.299.

40. Wiley, Dianne and Weihs, Gustavo Fimbres (2016). "Mixing Index". In: *Encyclopedia of Membranes*. Ed. by Enrico Drioli and Lidietta Giorno. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 1315–1317. ISBN: 978-3-662-44324-8. DOI: 10.1007/978-3-662-44324-8_2080. URL: https://doi.org/10.1007/978-3-662-44324-8_2080.

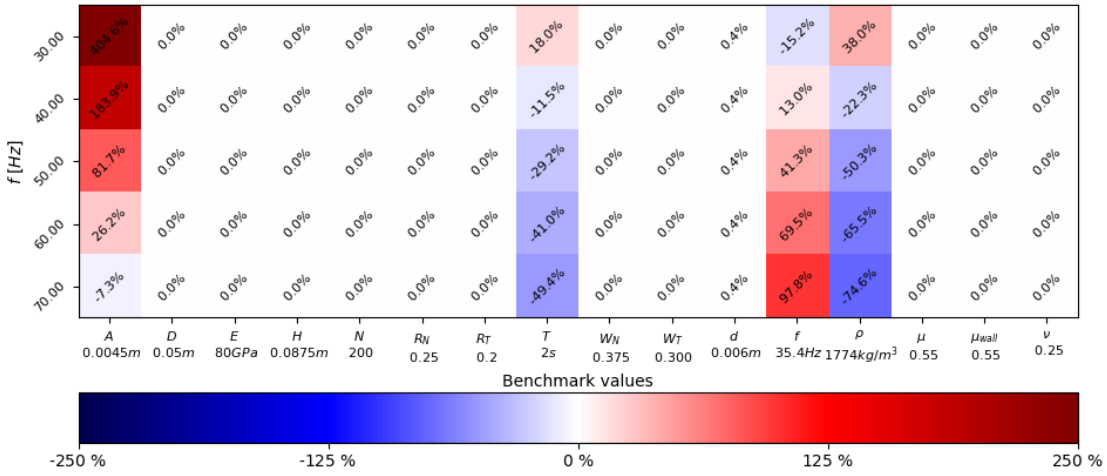
41. Wills B. A., Finch J. A. (2016). "Wills' Mineral Processing Technology - An Introduction to the Practical Aspects of Ore Treatment and Mineral Recovery". In: Eight. Elsevier, p. 451. URL: https://app.knovel.com/hotlink/toc/id:kpWMPTAI01/wills-mineral-processing/wills-mineral-processing.

42. Zhou, Y.C. et al. (July 1999). "Rolling friction in the dynamic simulation of sandpile formation". In: *Physica A: Statistical Mechanics and its Applications* 269.2-4, pp. 536–553. DOI: 10.1016/s0378-4371(99)00183-1. URL: https://doi.org/10.1016/s0378-4371(99)00183-1.

Appendix A

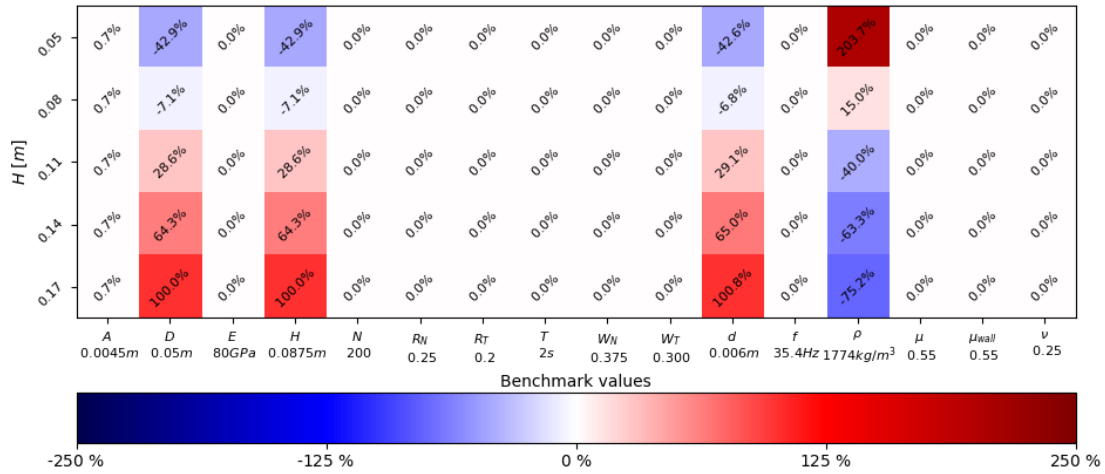
Variable Independence data

A.1 Variable independence adjustment

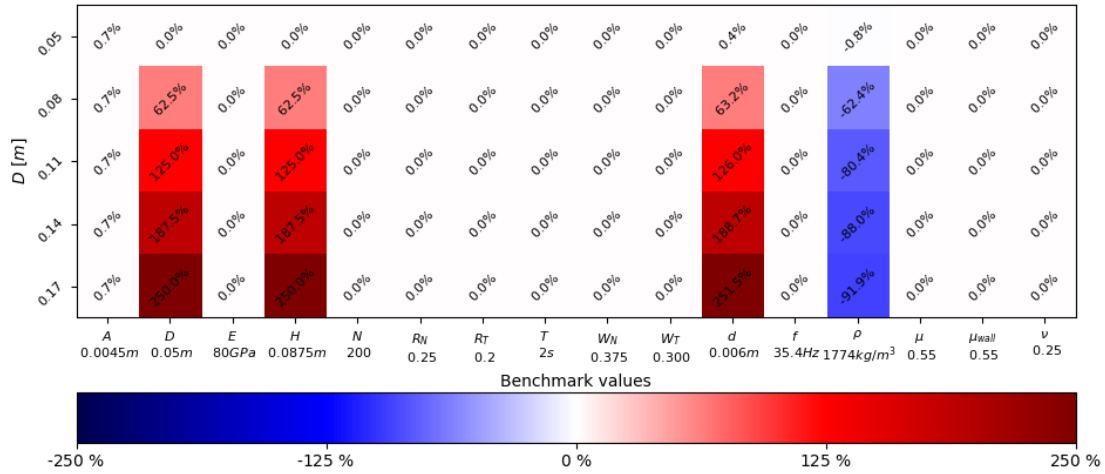


(a) Frequency independence

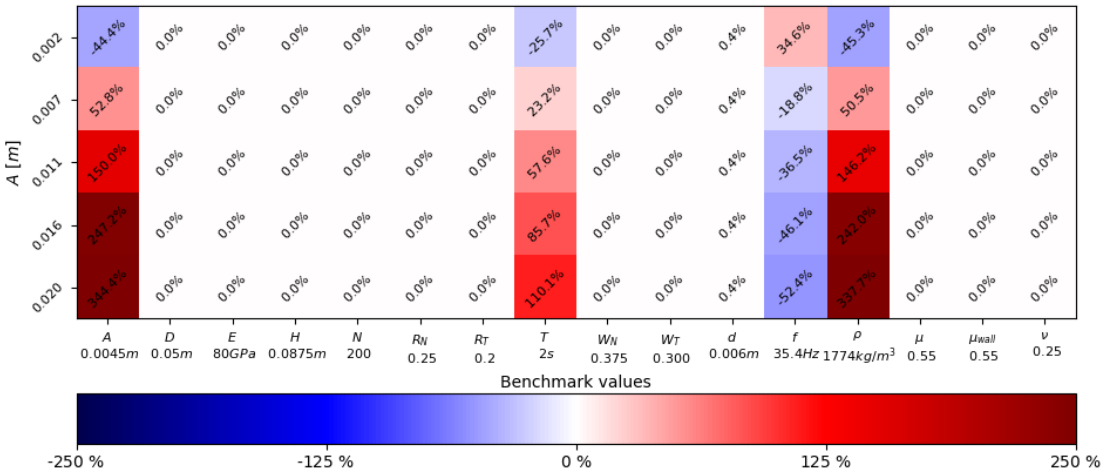
Variable adjustment



(b) Cup height independence

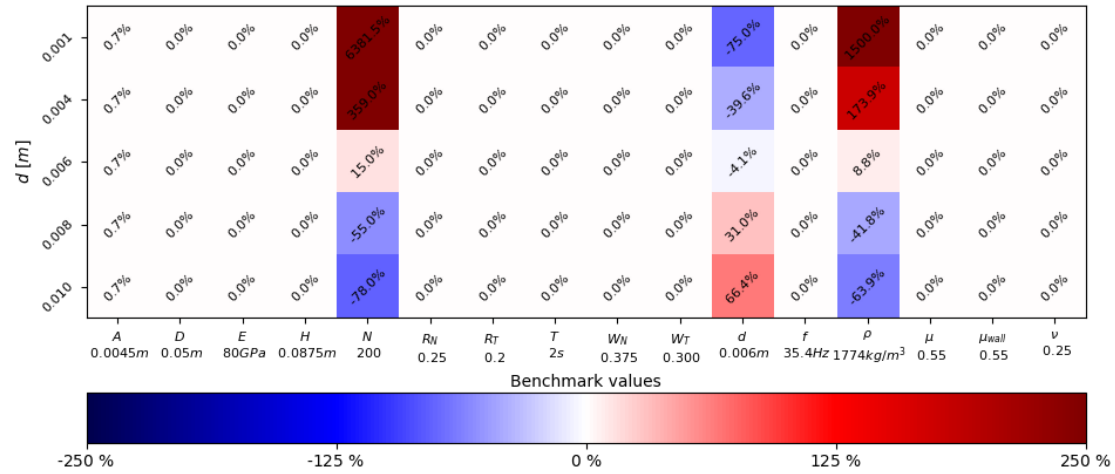


(c) Cup diameter independence

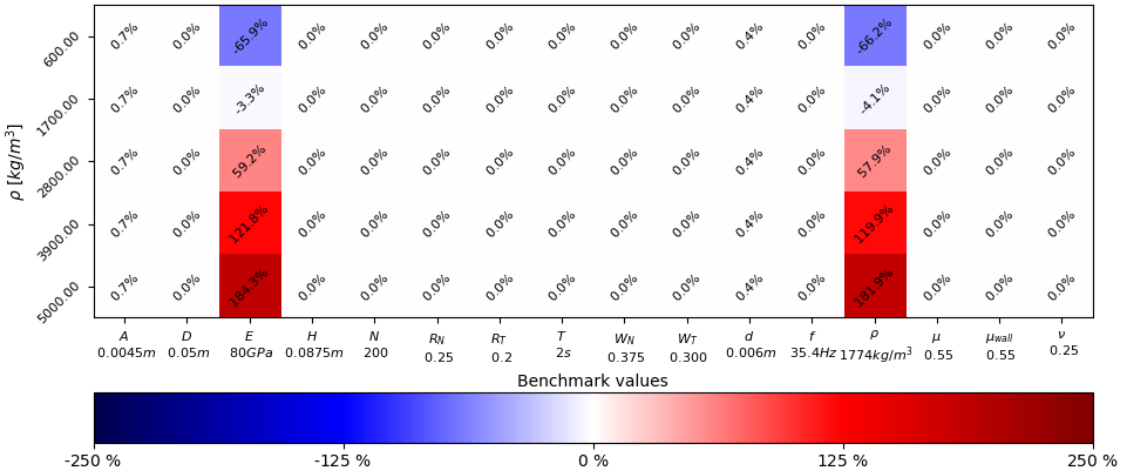


(d) Amplitude independence

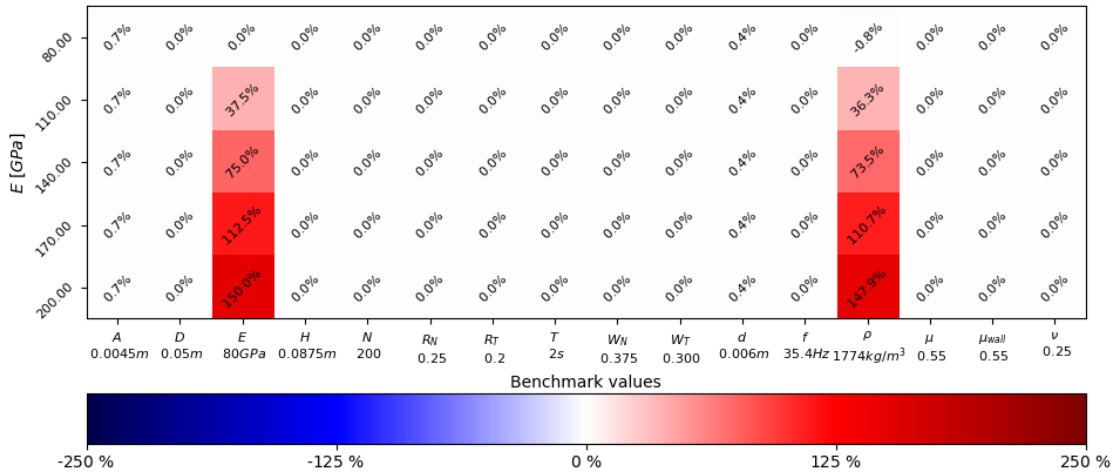
Variable adjustment



(e) Particle diameter independence

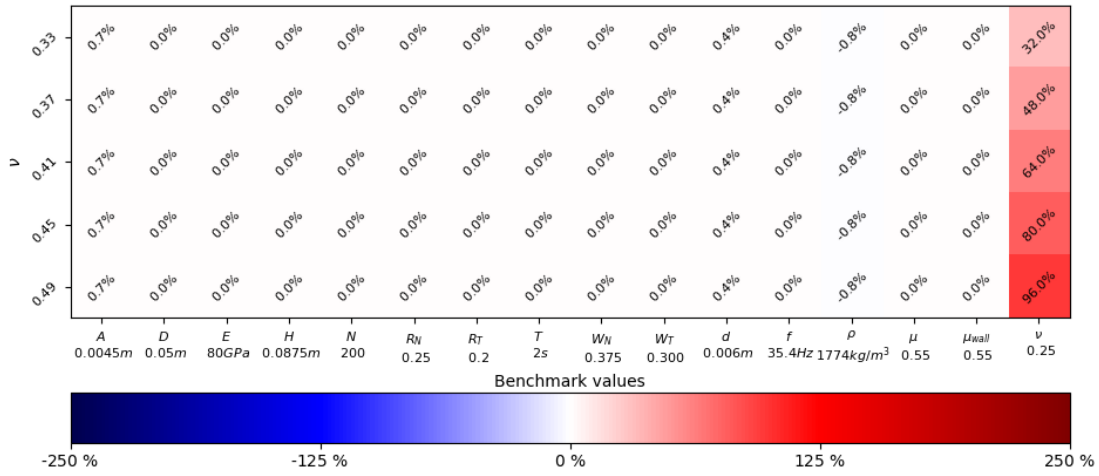


(f) Particle density independence

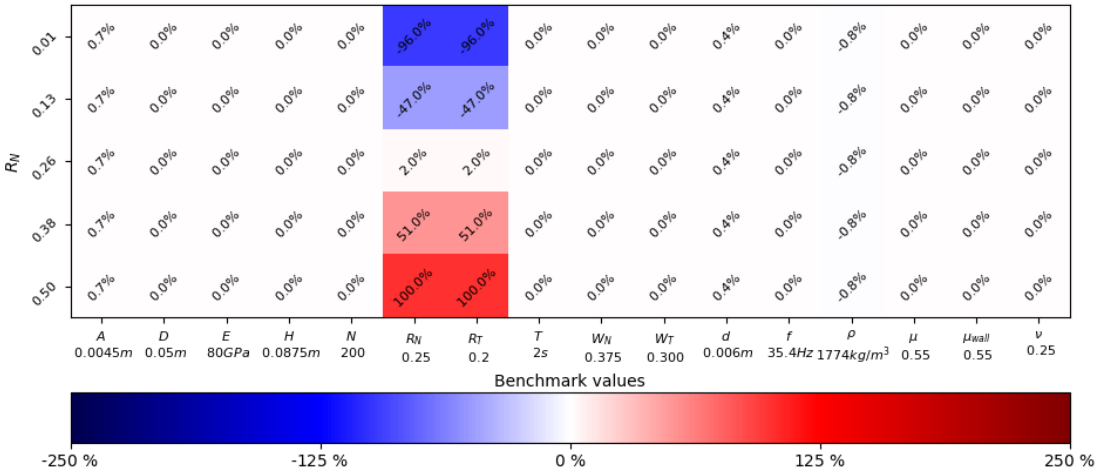


(g) Particle elasticity independence

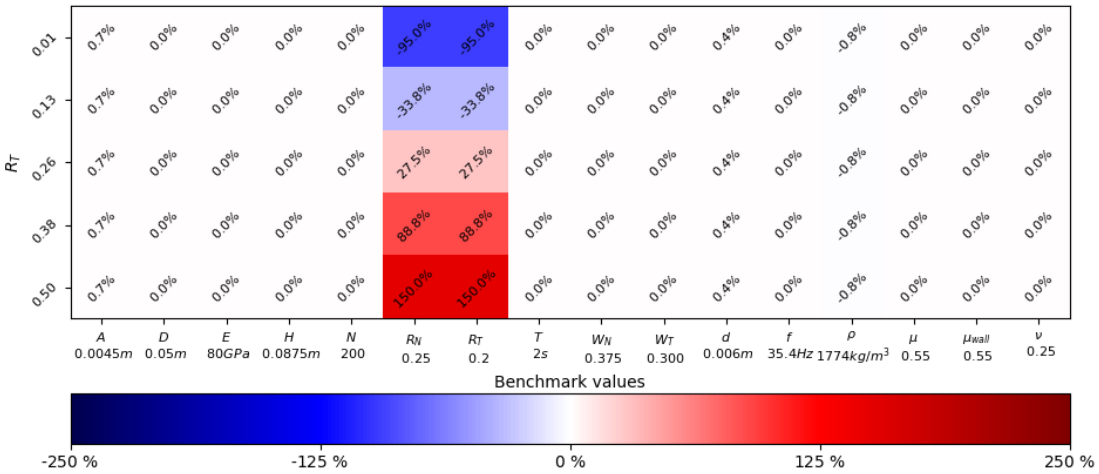
Variable adjustment



(h) Variable adjustment; Particle Poissons ratio independence

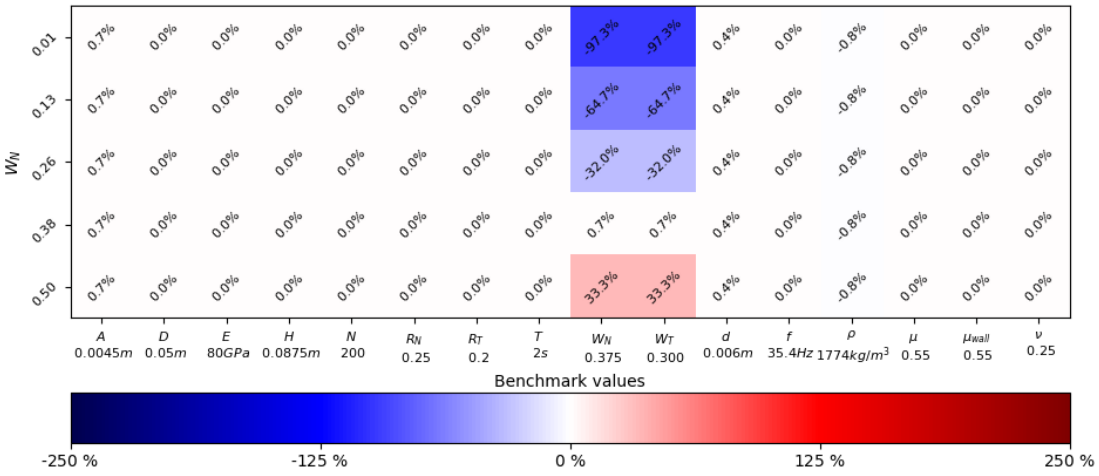


(i) Particle Normal restitution independence

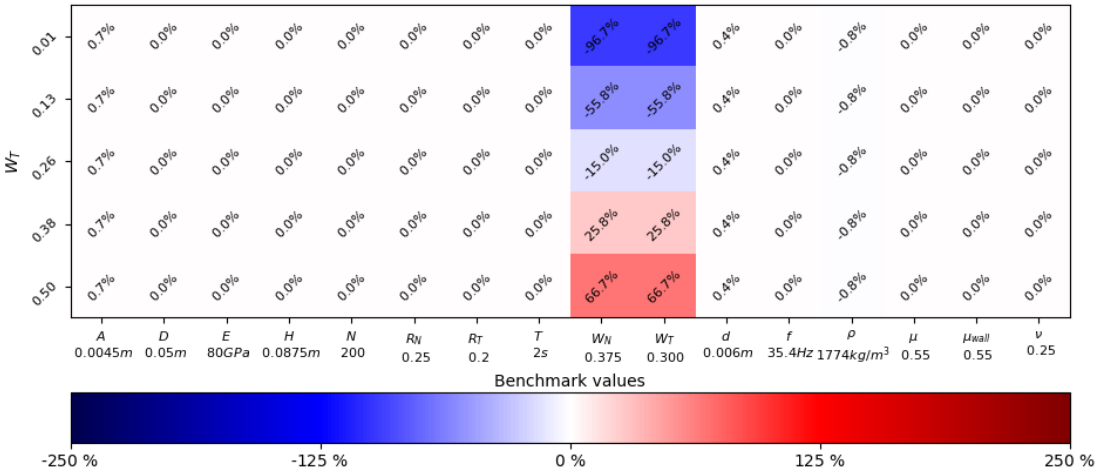


(j) Particle Tangential restitution independence

Variable adjustment



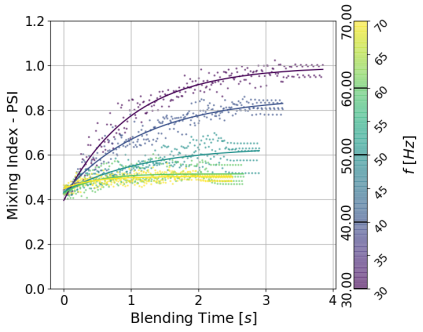
(k) Wall Normal restitution independence



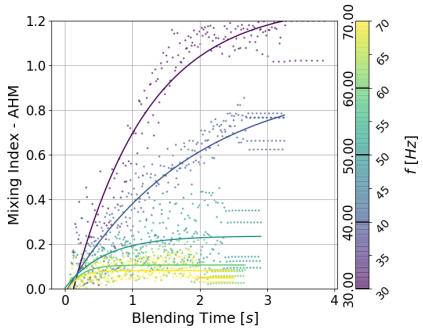
(l) Wall Tangential restitution independence

Variable adjustment

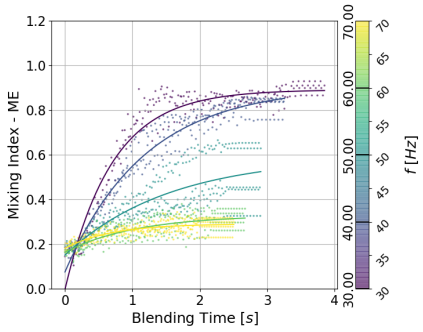
A.2 Variable independence data



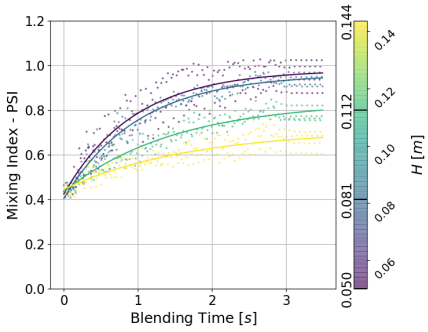
(a) PSI - Frequency



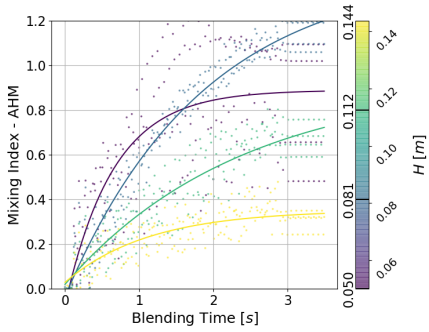
(b) AHM - Frequency



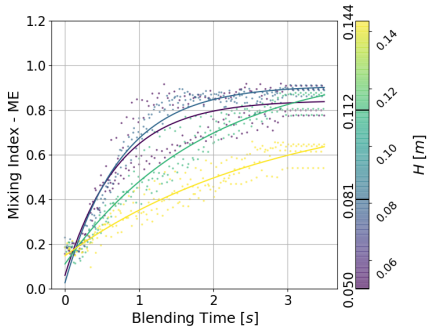
(c) ME - Frequency



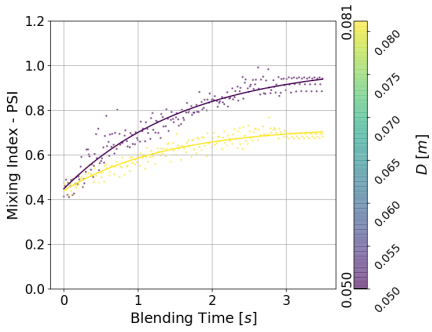
(d) PSI - Cup height



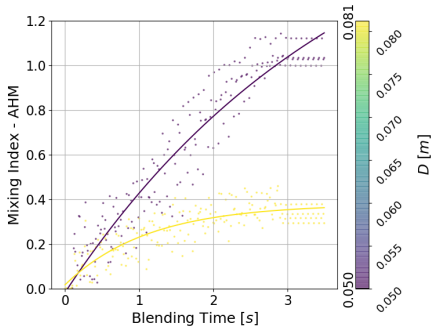
(e) AHM - Cup height



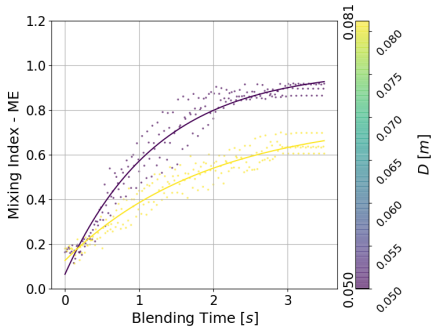
(f) ME - Cup height



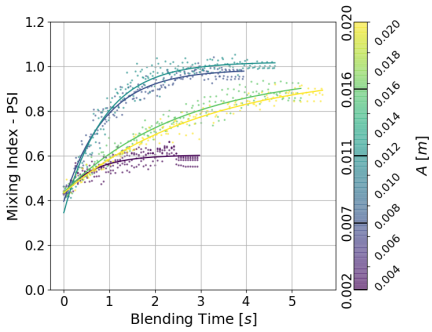
(g) PSI - Cup diameter



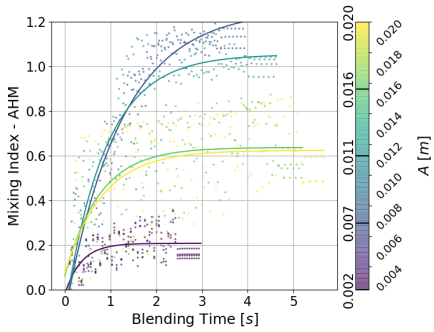
(h) AHM - Cup diameter



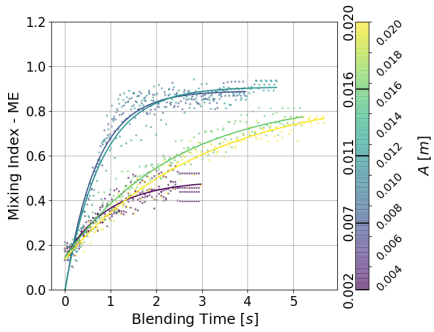
(i) ME - Cup diameter



(j) PSI - Amplitude

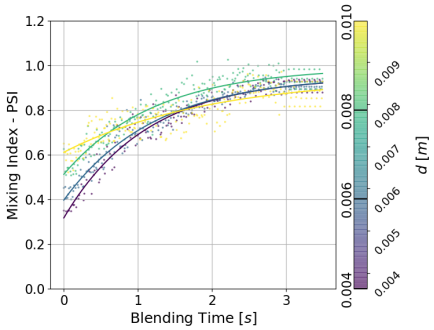


(k) AHM - Amplitude

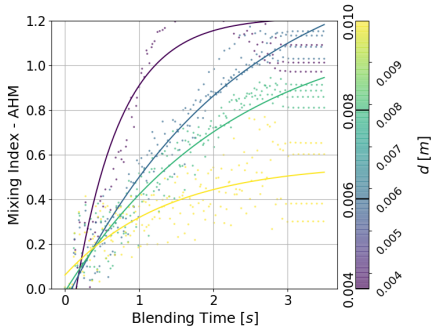


(l) ME - Amplitude

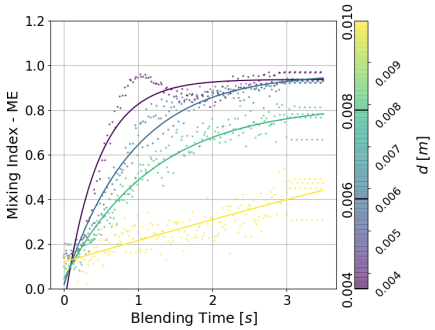
Variable Independence analysis - Global Variables



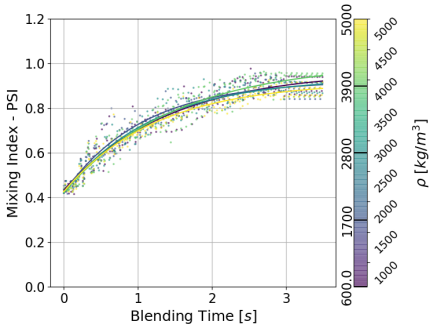
(m) PSI - Particle diameter



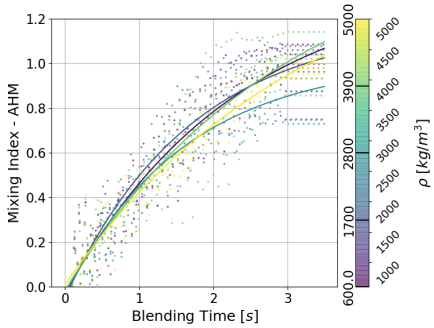
(n) AHM - Particle diameter



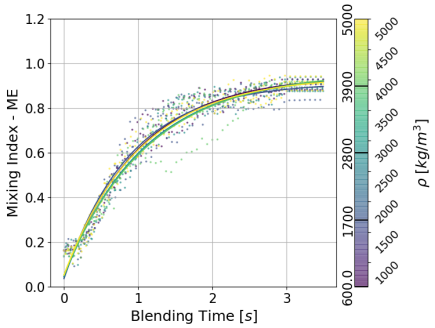
(o) ME - Particle diameter



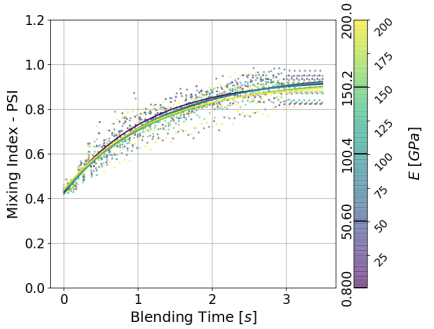
(p) PSI - Particle density



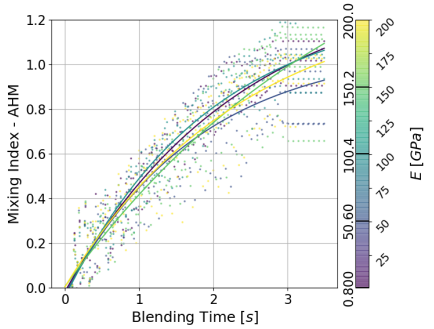
(q) AHM - Particle density



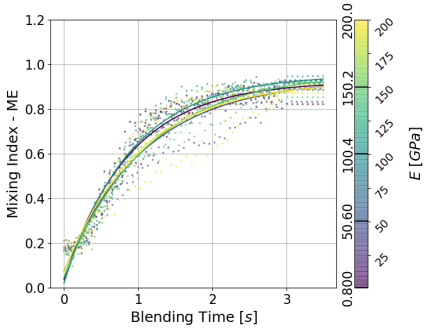
(r) ME - Particle density



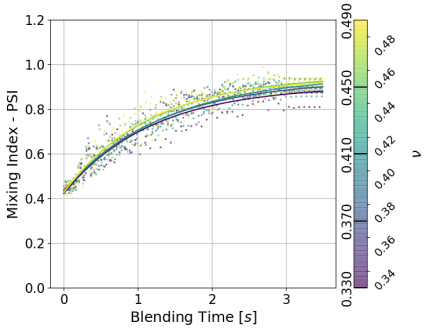
(s) PSI - Particle elasticity



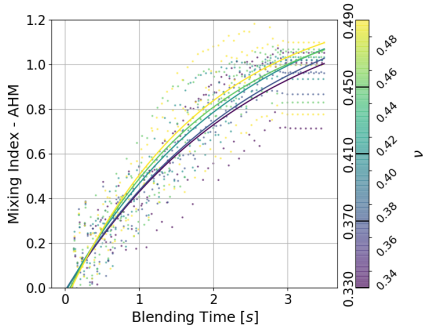
(t) AHM - Particle elasticity



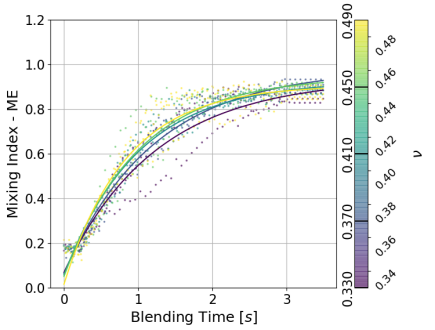
(u) ME - Particle elasticity



(v) PSI - Particle Poissons ratio

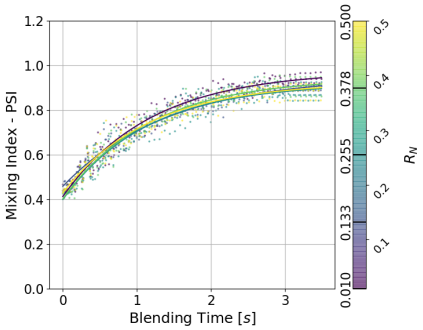


(w) AHM - Particle Poissons ratio

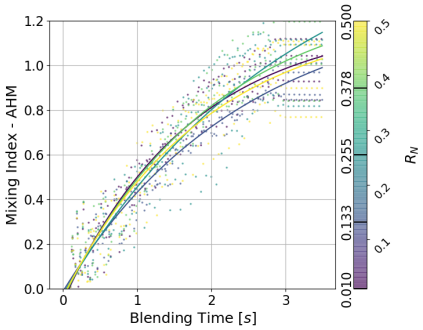


(x) ME - Particle Poissons ratio

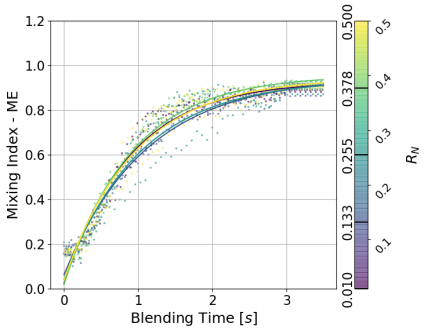
Variable Independence analysis - Local Variables, Material properties



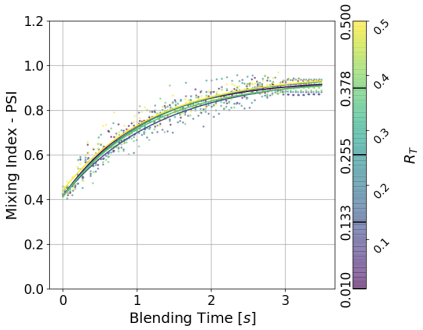
(y) PSI - Part. restitution N.



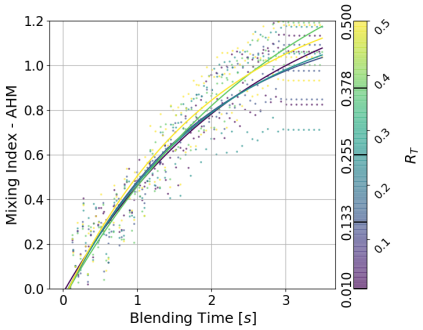
(z) AHM - Part. restitution N.



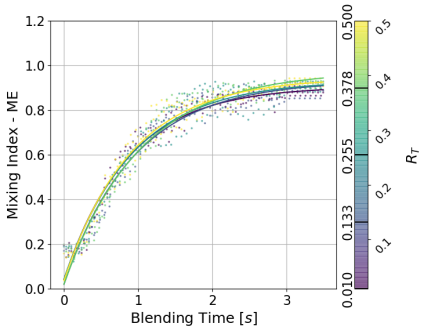
(aa) ME - Part. restitution N.



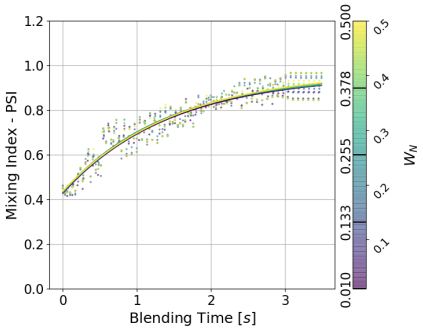
(ab) PSI - Part. restitution T.



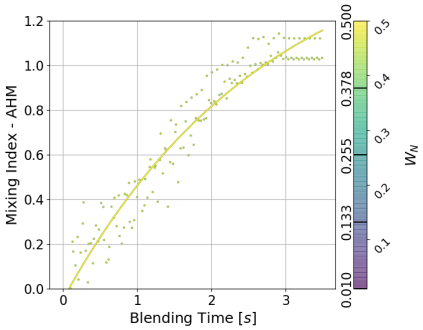
(ac) AHM - Part. restitution T.



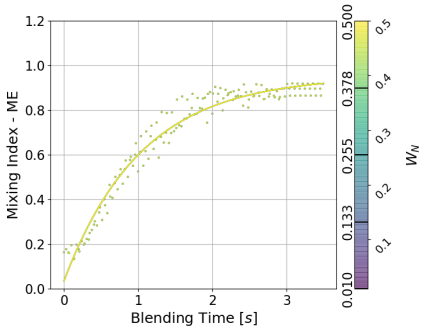
(ad) ME - Part. restitution T.



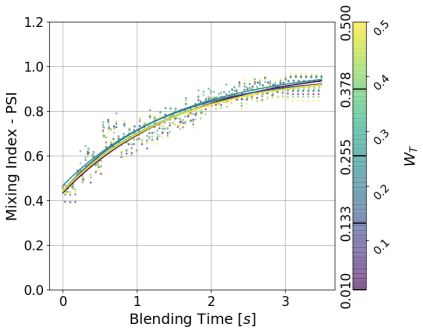
(ae) PSI - Wall restitution N.



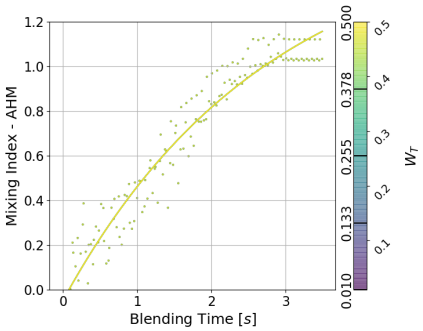
(af) AHM - Wall restitution N.



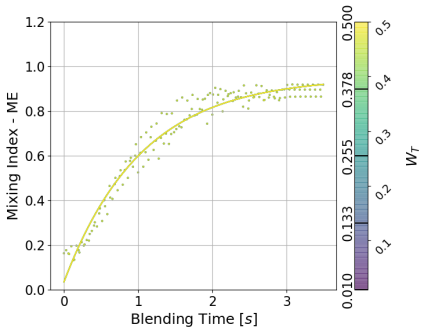
(ag) ME - Wall restitution N.



(ah) PSI - Wall restitution T.

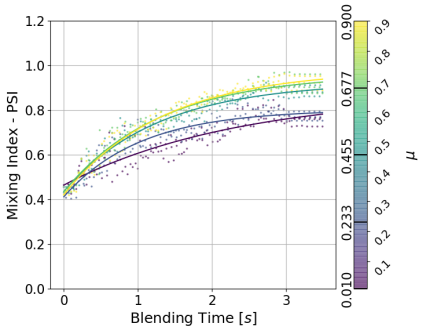


(ai) AHM - Wall restitution T.

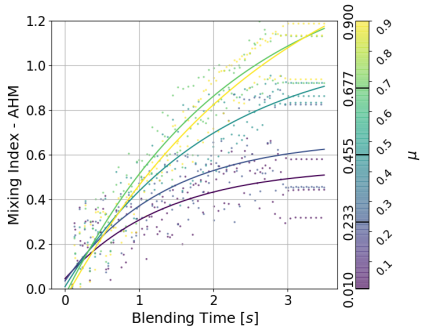


(aj) ME - Wall restitution T.

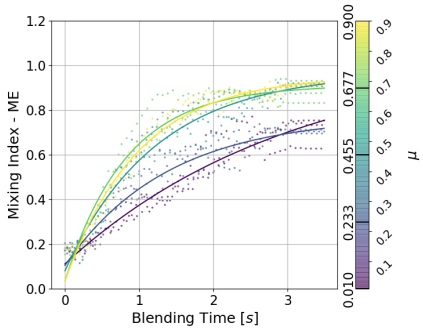
Variable Independence analysis - Local Variables, Restitution properties



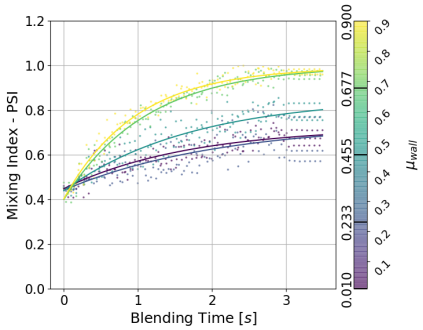
(ak) PSI - Interparticle friction



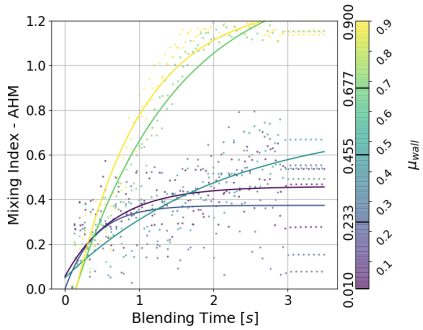
(al) AHM - Interparticle friction



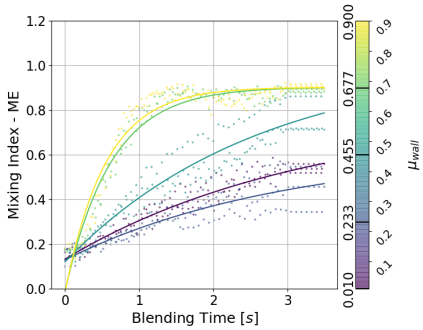
(am) ME - Interparticle friction



(an) PSI - Wall friction



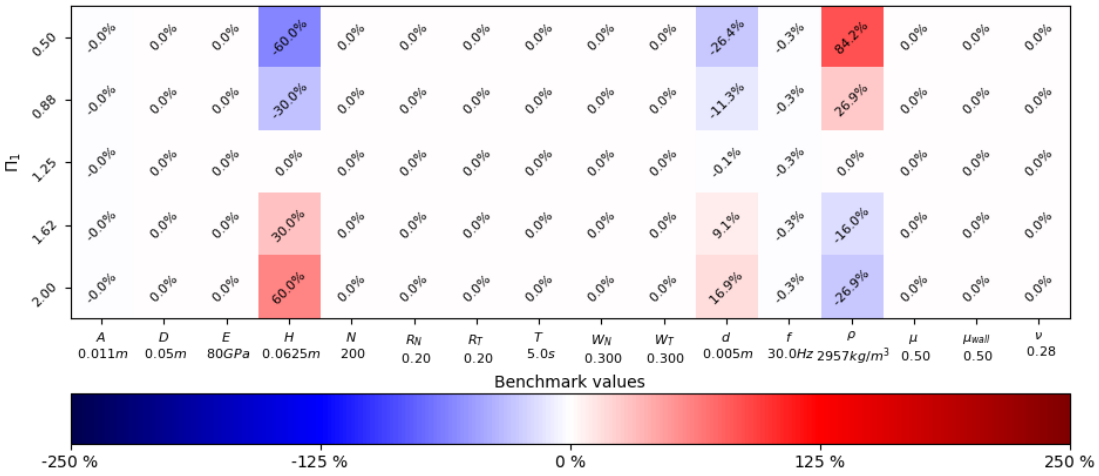
(ao) AHM - Wall friction



(ap) ME - Wall friction

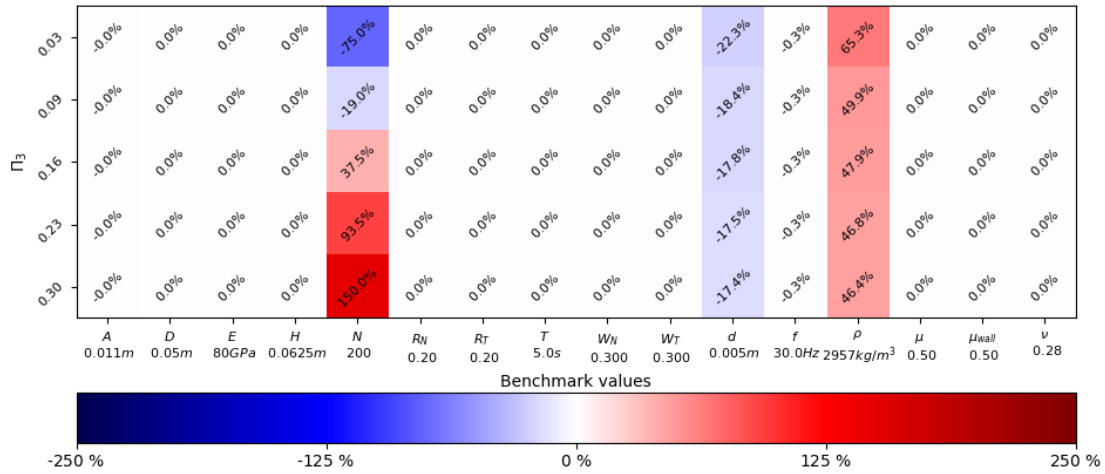
Variable Independence analysis - Local Variables, Friction properties

A.3 Sensitivity study adjustment

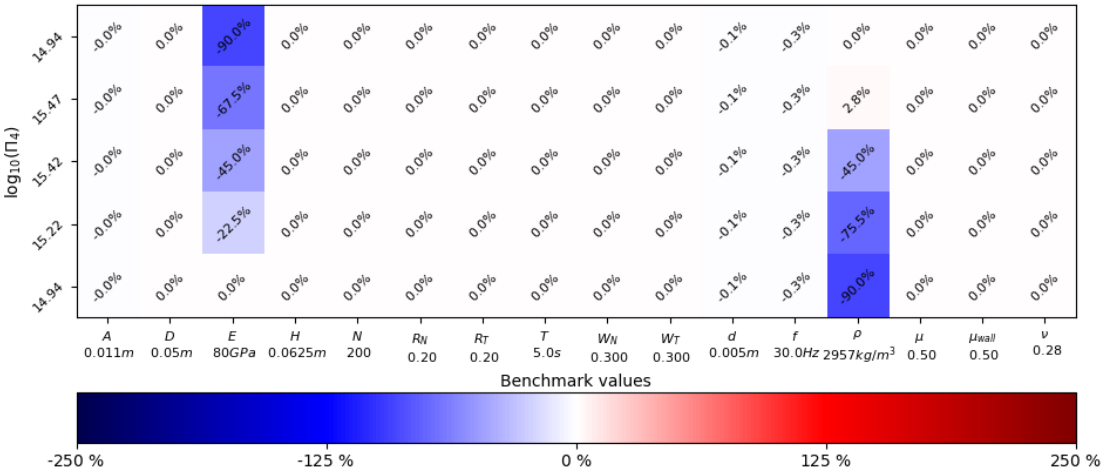


(a) Π_1 sensitivity

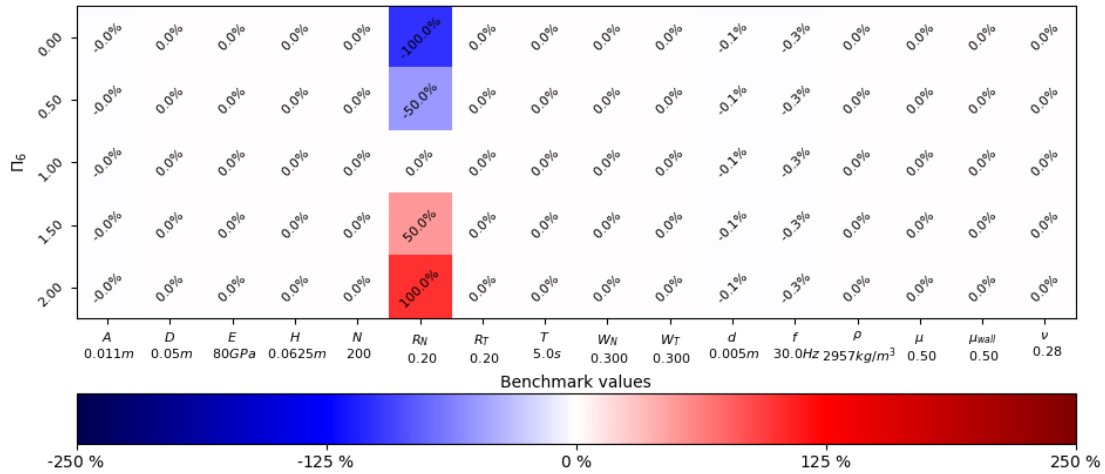
Π_k variable adjustment



(b) Π_3 sensitivity



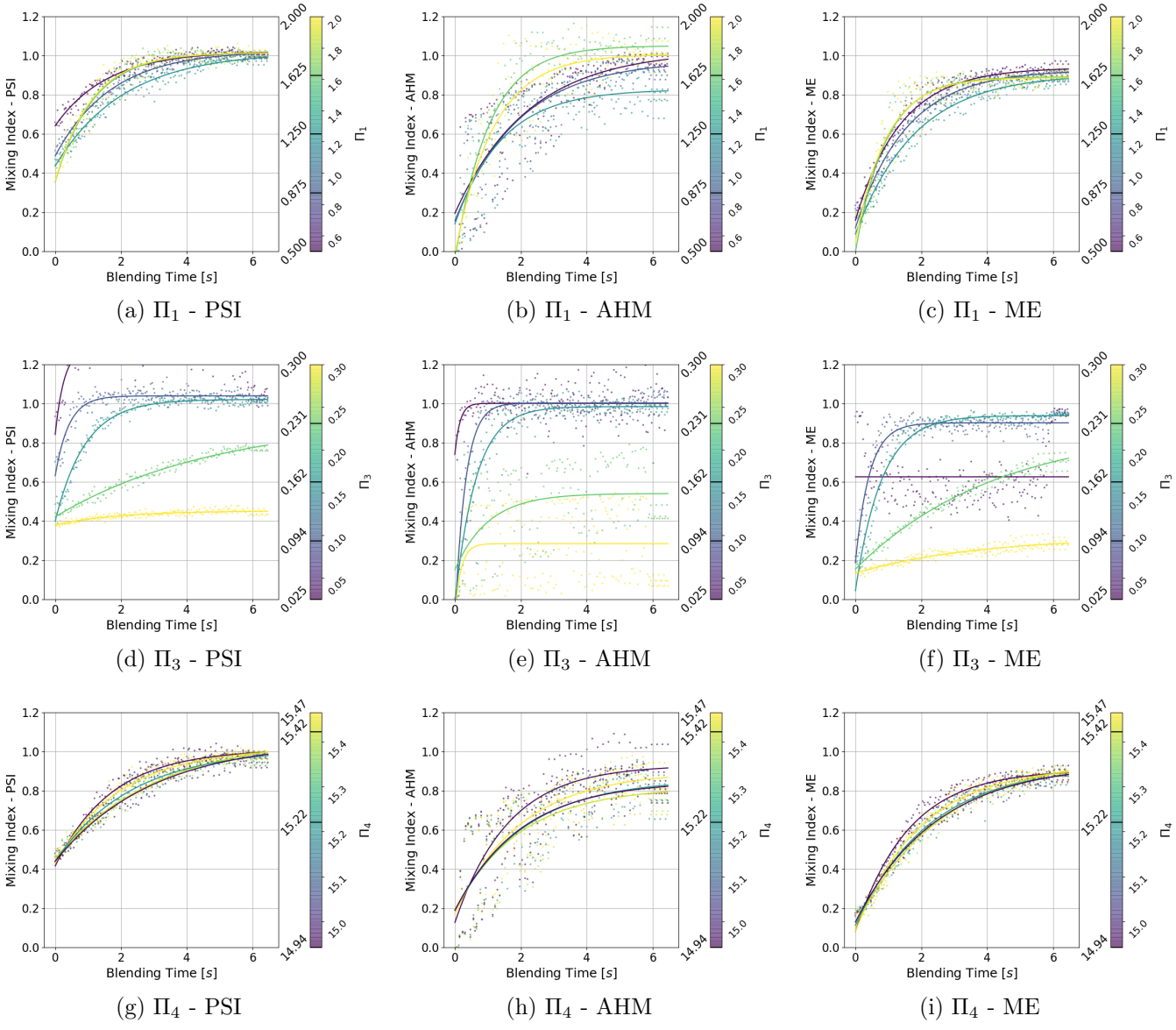
(c) Π_4 sensitivity



(d) Π_6 sensitivity

Π_k variable adjustment

A.4 Sensitivity study data

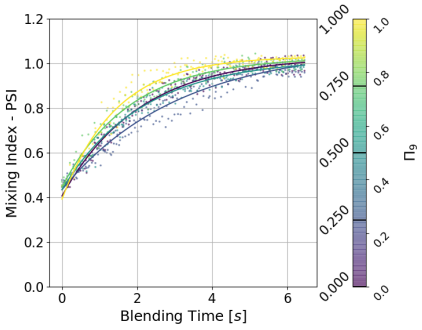


(a) Π_1 - PSI (b) Π_1 - AHM (c) Π_1 - ME

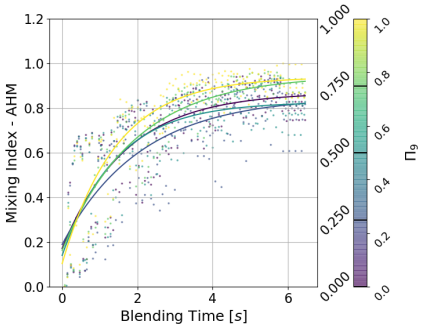
(d) Π_3 - PSI (e) Π_3 - AHM (f) Π_3 - ME

(g) Π_4 - PSI (h) Π_4 - AHM (i) Π_4 - ME

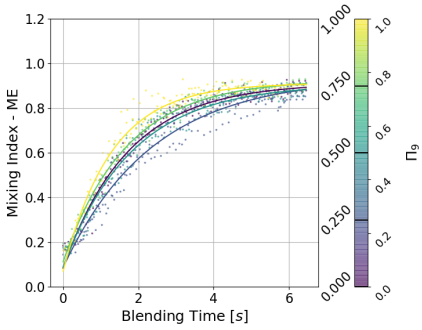
Π_k sensitivity analysis



(j) Π_6 - PSI



(k) Π_6 - AHM



(l) Π_6 - ME

Π_k sensitivity analysis

Appendix B

Experimental Data

Date	**Test No.**	H $[mm]$	D $[mm]$	N	T $[s]$	A $[mm]$	f $[Hz]$	μ	μ_{wall}
15/11/2017	2	18	36	64	5	15	40	0.2	0.2
15/11/2017	2	36	36	130	5	15	40	0.2	0.2
16/11/2017	1	72	36	260	5	15	40	0.2	0.2
16/11/2017	5	69	46	270	5	15	40	0.2	0.2
16/11/2017	6	69	46	406	5	15	40	0.2	0.2
16/11/2017	7	69	46	540	5	15	40	0.2	0.2
17/11/2017	1	54	36	194	5	7.5	19	0.2	0.2
17/11/2017	2	54	36	194	5	7.5	19	0.2	0.2
17/11/2017	3	54	36	194	5	7.5	19	0.2	0.2
17/11/2017	4	54	36	194	5	7.5	19	0.2	0.2
22/11/2017	6	54	36	194	5	7.5	19	0.2	0.2
17/11/2017	5	54	36	194	5	10	37	0.2	0.2
17/11/2017	6	54	36	194	5	10	37	0.2	0.2
17/11/2017	7	54	36	194	5	10	37	0.2	0.2
17/11/2017	8	54	36	194	5	10	37	0.2	0.2
17/11/2017	9	54	36	194	5	7.5	46	0.2	0.2
17/11/2017	10	54	36	194	5	7.5	46	0.2	0.2
17/11/2017	11	54	36	194	5	7.5	46	0.2	0.2
17/11/2017	12	54	36	194	5	7.5	46	0.2	0.2
22/11/2017	7	54	36	194	5	7.5	46	0.2	0.2
22/11/2017	1	54	36	194	5	15	38	0.2	0.2
22/11/2017	2	54	36	194	5	15	38	0.2	0.2
22/11/2017	3	54	36	194	5	15	38	0.2	0.2
22/11/2017	4	54	36	194	5	15	38	0.2	0.2
22/11/2017	5	54	36	194	5	15	38	0.2	0.2

Table B.1: Experimental parameters

Date	**Test No.**	H $[mm]$	D $[mm]$	N	T $[s]$	A $[mm]$	f $[Hz]$	μ	μ_{wall}
22/11/2017	8	54	36	194	5	10	36	0.2	0.2
22/11/2017	9	54	36	130	5	15	40	0.2	0.2
22/11/2017	10	54	36	260	5	15	40	0.2	0.2
23/11/2017	1	39	26	48	5	15	40	0.2	0.2
23/11/2017	2	39	26	74	5	15	40	0.2	0.2
23/11/2017	3	39	26	98	5	15	40	0.2	0.2
27/11/2017	1	13	26	24	5	15	40	0.2	0.2
27/11/2017	3	52	26	98	5	15	40	0.2	0.2
27/11/2017	6	23	46	152	5	15	40	0.2	0.2
27/11/2017	7	46	46	270	5	15	40	0.2	0.2
27/11/2017	8	92	46	540	5	15	40	0.2	0.2

Table B.1: Experimental parameters (continued)

Date	**Test No.**	$\log_{10}(\Pi_0)$	Π_1	Π_2	Π_3	$\log_{10}(\Pi_4)$	Π_8	Π_9
15/11/2017	1	2.3	0.5	2.4	0.6	5.6	0.2	0.2
15/11/2017	2	2.3	1.0	2.4	0.6	5.6	0.2	0.2
16/11/2017	1	2.3	2.0	2.4	1.2	5.6	0.2	0.2
16/11/2017	5	2.3	1.5	2.4	0.4	5.6	0.2	0.2
16/11/2017	6	2.3	1.5	2.4	0.6	5.6	0.2	0.2
16/11/2017	7	2.3	1.5	2.4	0.8	5.6	0.2	0.2
17/11/2017	1	2.0	1.5	1.73	0.6	6.2	0.2	0.2
17/11/2017	2	2.0	1.5	1.73	0.6	6.2	0.2	0.2
17/11/2017	3	2.0	1.5	1.73	0.6	6.2	0.2	0.2
17/11/2017	4	2.0	1.5	1.73	0.6	6.2	0.2	0.2
22/11/2017	6	2.0	1.5	1.73	0.6	6.2	0.2	0.2
17/11/2017	5	2.3	1.5	8.76	0.6	5.6	0.2	0.2
17/11/2017	6	2.3	1.5	8.76	0.6	5.6	0.2	0.2
17/11/2017	7	2.3	1.5	8.76	0.6	5.6	0.2	0.2
17/11/2017	8	2.3	1.5	8.76	0.6	5.6	0.2	0.2
17/11/2017	9	2.4	1.5	10.16	0.6	5.4	0.2	0.2
17/11/2017	10	2.4	1.5	10.16	0.6	5.4	0.2	0.2
17/11/2017	11	2.4	1.5	10.16	0.6	5.4	0.2	0.2
17/11/2017	12	2.4	1.5	10.16	0.6	5.4	0.2	0.2
22/11/2017	7	2.4	1.5	10.16	0.6	5.4	0.2	0.2
22/11/2017	1	2.3	1.5	13.87	0.6	5.6	0.2	0.2
22/11/2017	2	2.3	1.5	13.87	0.6	5.6	0.2	0.2
22/11/2017	3	2.3	1.5	13.87	0.6	5.6	0.2	0.2
22/11/2017	4	2.3	1.5	13.87	0.6	5.6	0.2	0.2
22/11/2017	5	2.3	1.5	13.87	0.6	5.6	0.2	0.2
22/11/2017	8	2.3	1.5	1.0	0.6	5.6	0.2	0.2

Table B.2: Experimental Π parameters

Date	**Test No.**	$\log_{10}(\Pi_0)$	Π_1	Π_2	Π_3	$\log_{10}(\Pi_4)$	Π_8	Π_9
22/11/2017	9	2.3	1.5	2.5	0.4	5.6	0.2	0.2
22/11/2017	10	2.3	1.5	2.5	0.8	5.6	0.2	0.2
23/11/2017	1	2.3	1.5	2.5	0.4	5.6	0.2	0.2
23/11/2017	2	2.3	1.5	2.5	0.6	5.6	0.2	0.2
23/11/2017	3	2.3	1.5	2.5	0.8	5.6	0.2	0.2
27/11/2017	1	2.3	0.5	2.5	0.6	5.6	0.2	0.2
27/11/2017	3	2.3	2.0	2.5	0.6	5.6	0.2	0.2
27/11/2017	6	2.3	0.5	2.5	0.6	5.6	0.2	0.2
27/11/2017	7	2.3	1.0	2.5	0.6	5.6	0.2	0.2
27/11/2017	8	2.3	2.0	2.5	0.6	5.6	0.2	0.2

Table B.2: Experimental Π parameters (continued)

Date	**Test No.**	**AHM**	**CN**	**Lacey**	**ME**	**PSI**
15/11/2017	1	0.72	0.77	0.43	0.55	0.90
15/11/2017	2	0.54	0.68	0.31	0.59	0.82
16/11/2017	1	0.00	0.21	0.12	0.16	0.47
16/11/2017	5	1.01	0.76	-0.08	0.70	0.90
16/11/2017	6	0.35	0.51	0.26	0.48	0.68
16/11/2017	7	0.04	0.17	0.16	0.22	0.43
17/11/2017	1	0.37	0.40	0.27	0.50	0.62
17/11/2017	2	0.17	0.44	0.29	0.41	0.62
17/11/2017	3	0.54	0.66	0.19	0.67	0.83
17/11/2017	4	0.32	0.46	0.20	0.47	0.64
22/11/2017	6	0.62	0.59	0.42	0.68	0.74
17/11/2017	5	0.58	0.57	0.15	0.66	0.77
17/11/2017	6	0.79	0.63	0.48	0.63	0.77
17/11/2017	7	0.73	0.79	0.34	0.83	0.93
17/11/2017	8	0.77	0.81	0.34	0.71	0.91
17/11/2017	9	0.72	0.86	0.61	0.91	0.93
17/11/2017	10	0.32	0.55	0.45	0.57	0.72
17/11/2017	11	0.57	0.73	0.66	0.83	0.87
17/11/2017	12	0.50	0.72	0.69	0.71	0.88
22/11/2017	7	0.58	0.80	0.34	0.88	0.95
22/11/2017	1	0.31	0.51	0.40	0.53	0.70
22/11/2017	2	1.05	1.00	0.69	0.93	1.03
22/11/2017	3	0.92	0.82	0.62	0.89	0.90
22/11/2017	4	0.81	0.68	0.44	0.75	0.84
22/11/2017	5	0.73	0.84	0.50	0.81	0.94
22/11/2017	8	0.39	0.65	0.17	0.63	0.82
22/11/2017	9	0.94	0.76	-0.26	0.78	0.95
22/11/2017	10	0.02	0.24	0.14	0.13	0.48

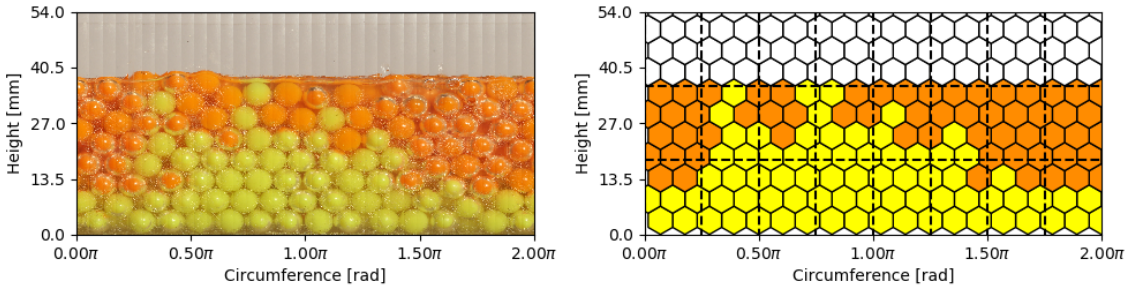
Table B.3: Experimental Mixing Information

Date	Test No.	AHM	CN	Lacey	ME	PSI
23/11/2017	1	0.36	0.61	0.95	0.72	0.92
23/11/2017	2	1.29	0.67	0.16	0.47	0.87
23/11/2017	3	-0.02	0.23	0.02	0.13	0.53
27/11/2017	1	0.22	0.93	-0.09	0.07	1.02
27/11/2017	3	0.31	0.41	0.26	0.30	0.64
27/11/2017	6	0.81	0.74	0.38	0.69	0.88
27/11/2017	7	0.92	0.56	0.49	0.51	0.74
27/11/2017	8	0.03	0.15	0.12	0.17	0.43

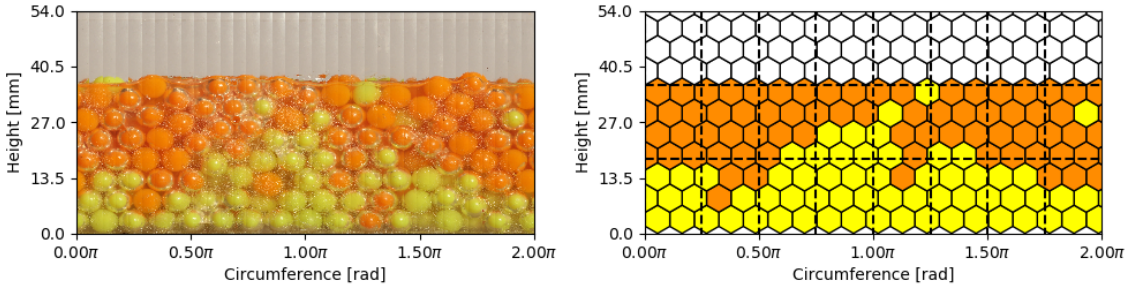
Table B.3: Experimental Mixing Information (continued)

B.1 Acceleration experiment

B.1.1 Acceleration - $\Pi_2 = 0.3$

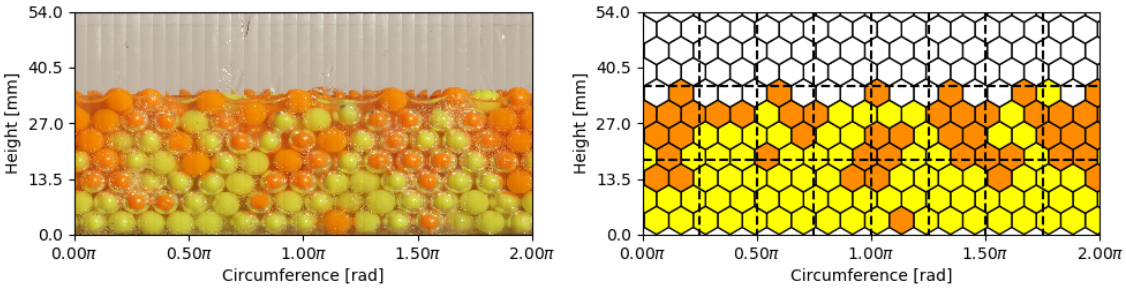


(a) Sample 1

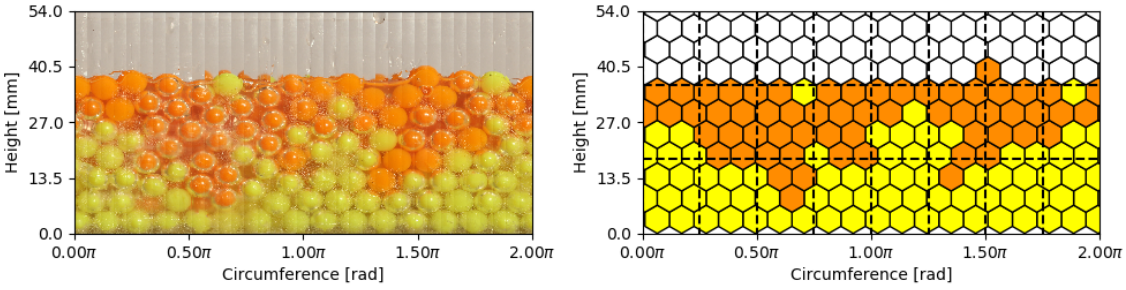


(b) Sample 2

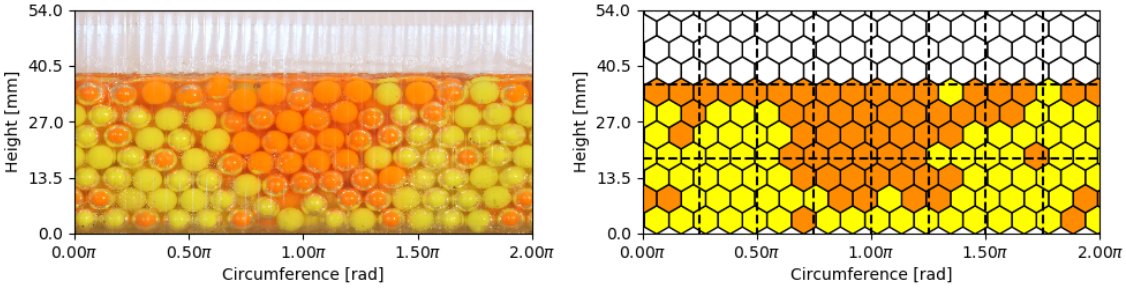
Figure B.1: $\Pi_2 = 0.3$



(c) Sample 3



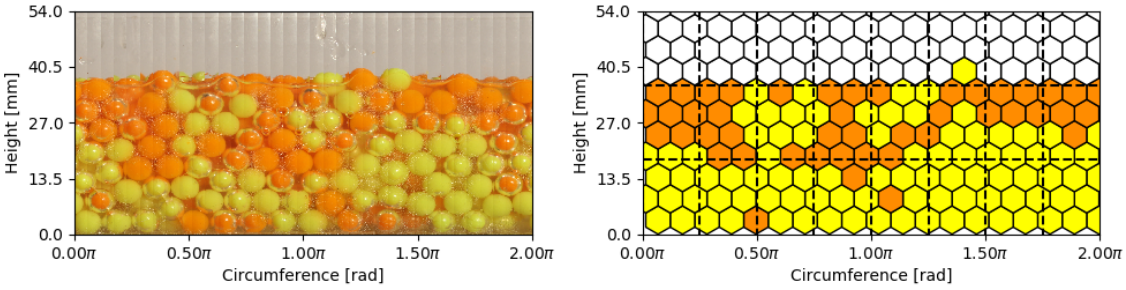
(d) Sample 4



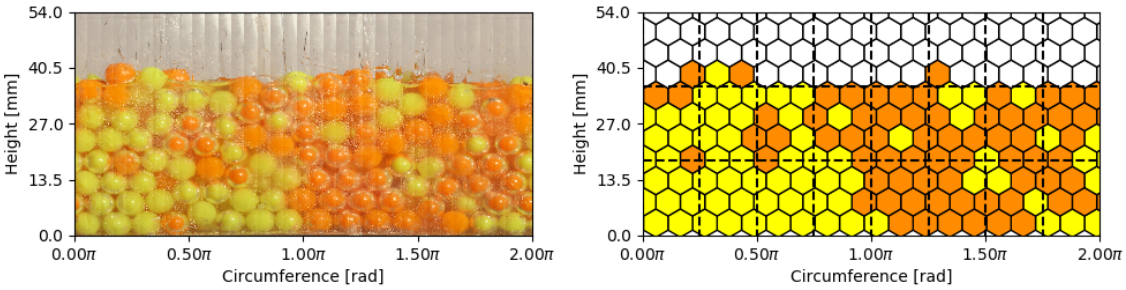
(e) Sample 5

Figure B.1: $\Pi_2 = 0.3$

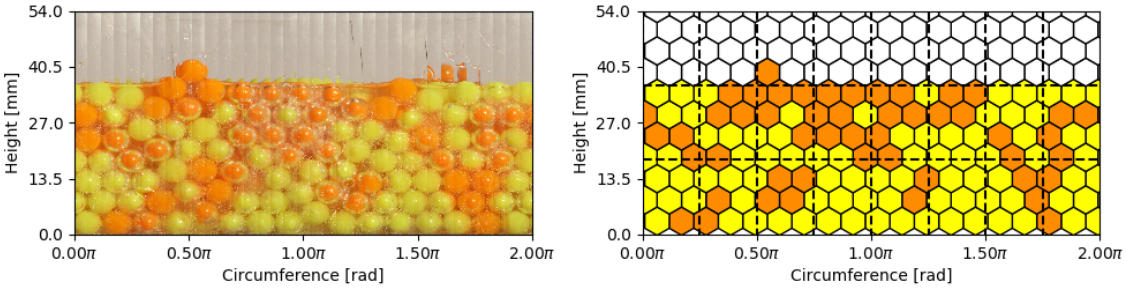
B.1.2 Acceleration - $\Pi_2 = 1.4$



(a) Sample 1

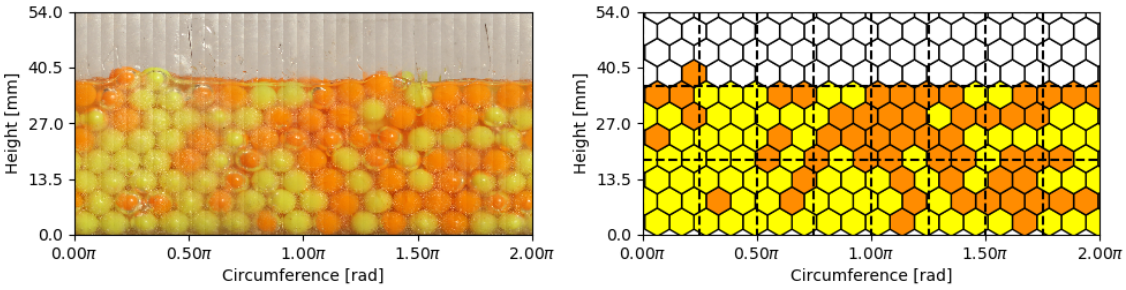


(b) Sample 2



(c) Sample 3

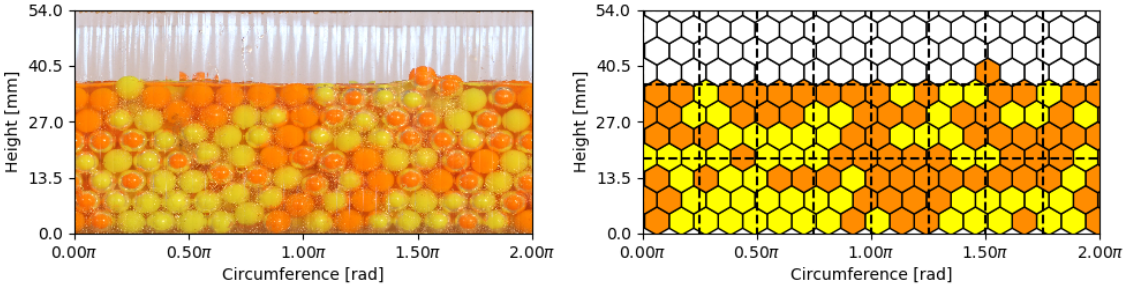
Figure B.2: $\Pi_2 = 1.4$



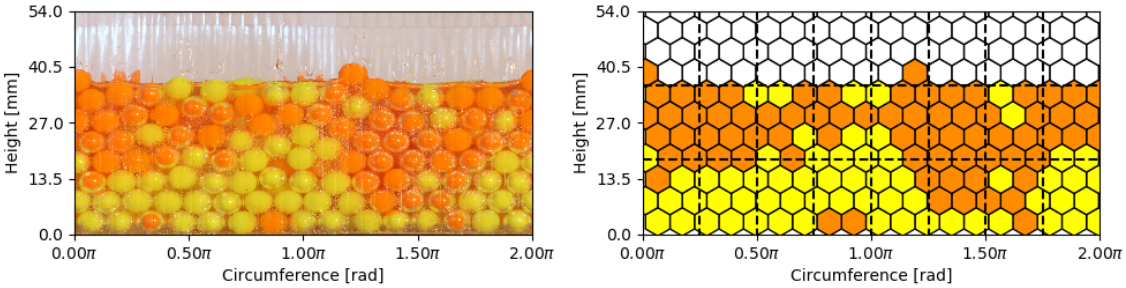
(d) Sample 4

Figure B.2: $\Pi_2 = 1.4$

B.1.3 Acceleration - $\Pi_2 = 1.6$

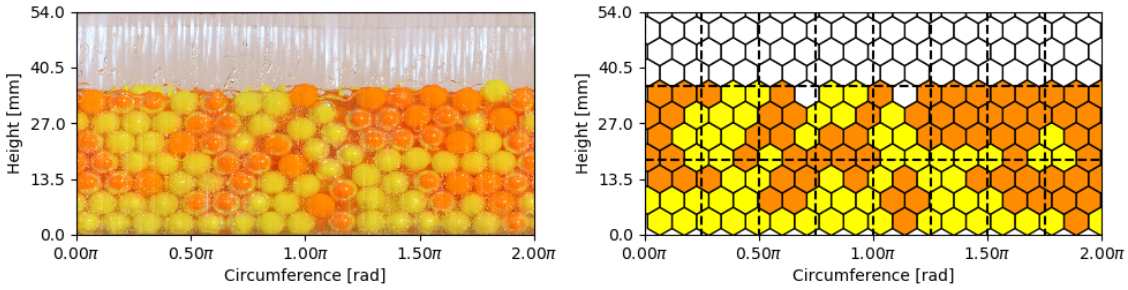


(a) Sample 1

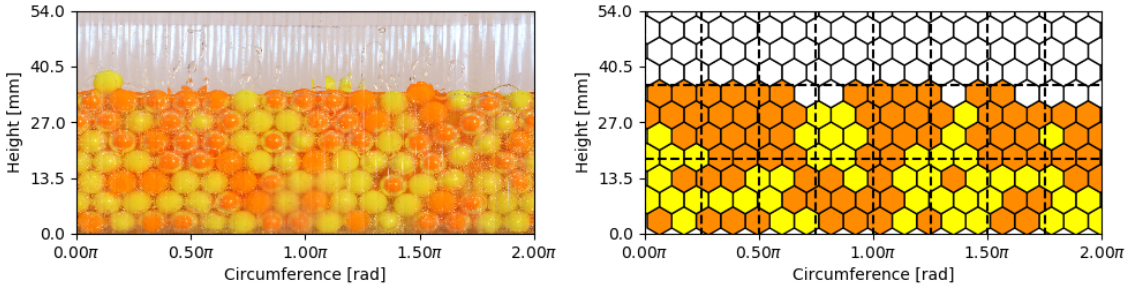


(b) Sample 2

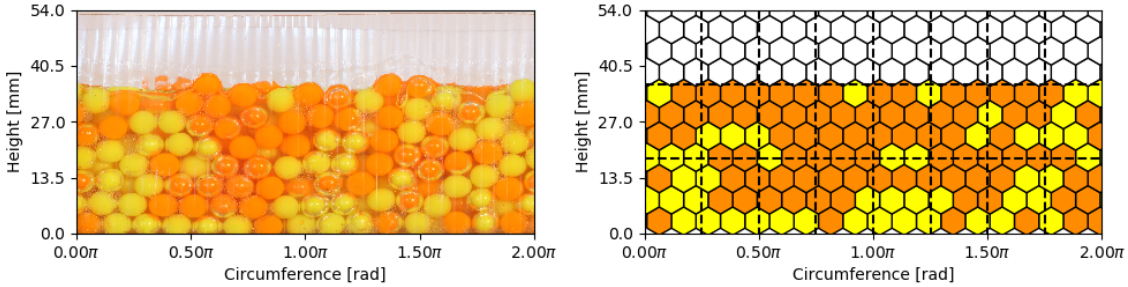
Figure B.3: $\Pi_2 = 1.6$



(c) Sample 3



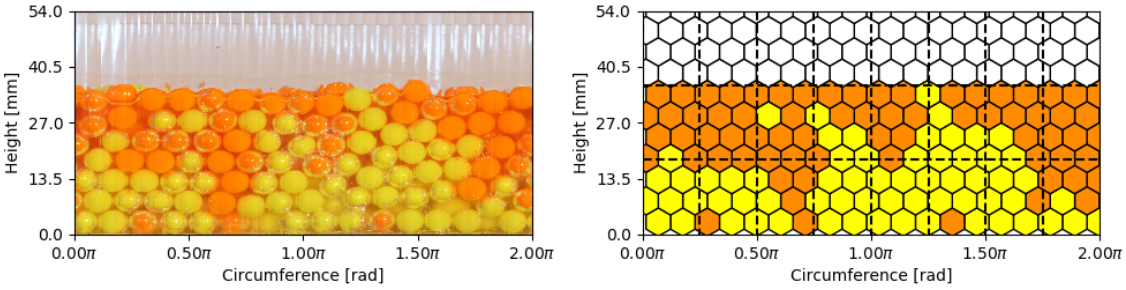
(d) Sample 4



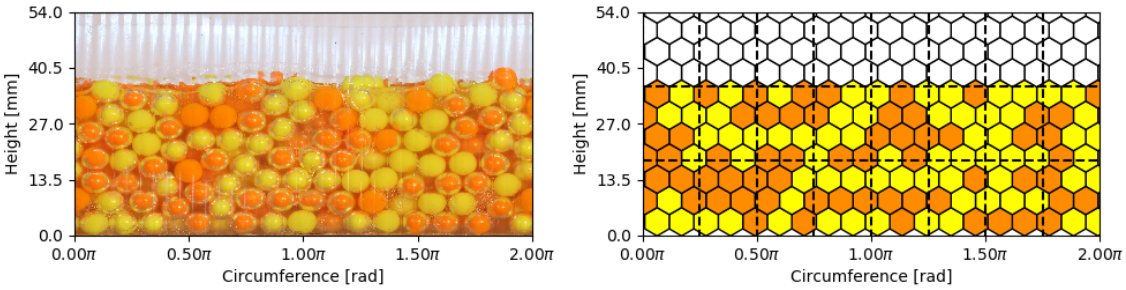
(e) Sample 5

Figure B.3: $\Pi_2 = 1.6$

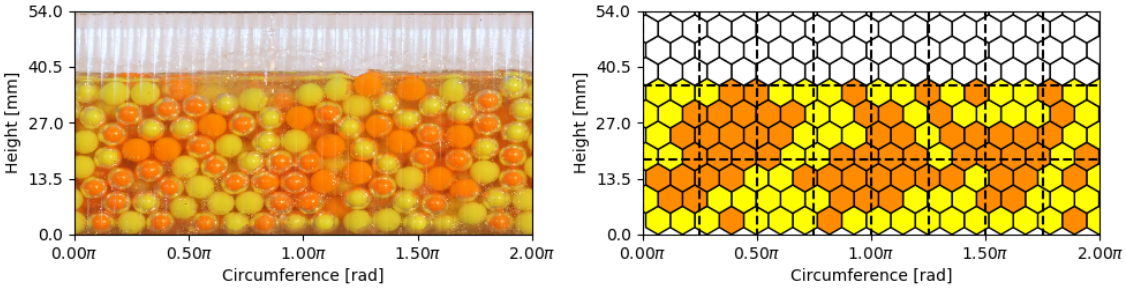
B.1.4 Acceleration - $\Pi_2 = 2.2$



(a) Sample 1

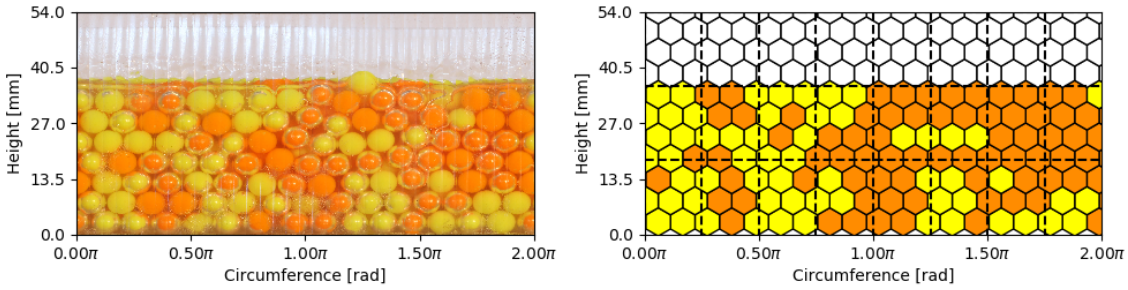


(b) Sample 2

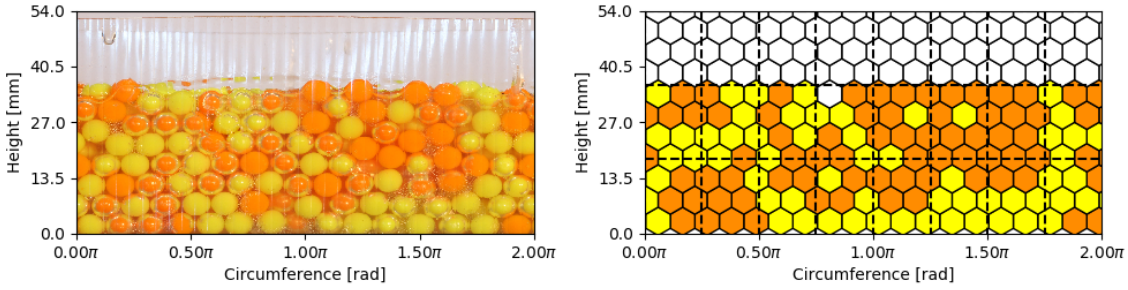


(c) Sample 3

Figure B.4: $\Pi_2 = 2.2$



(d) Sample 4

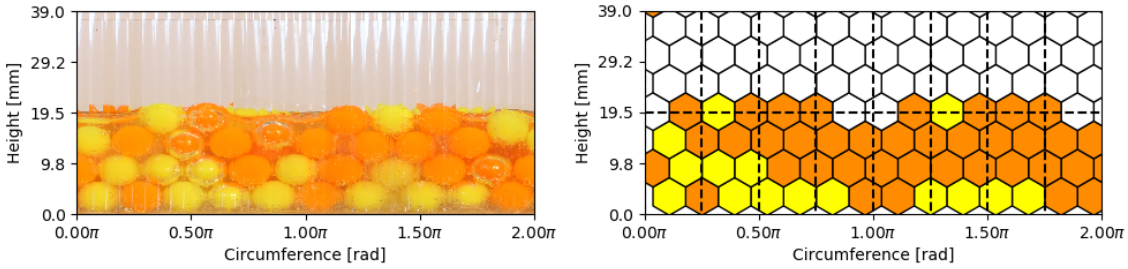


(e) $\Pi_2 = 2.2$ - Sample 5

Figure B.4: $\Pi_2 = 2.2$

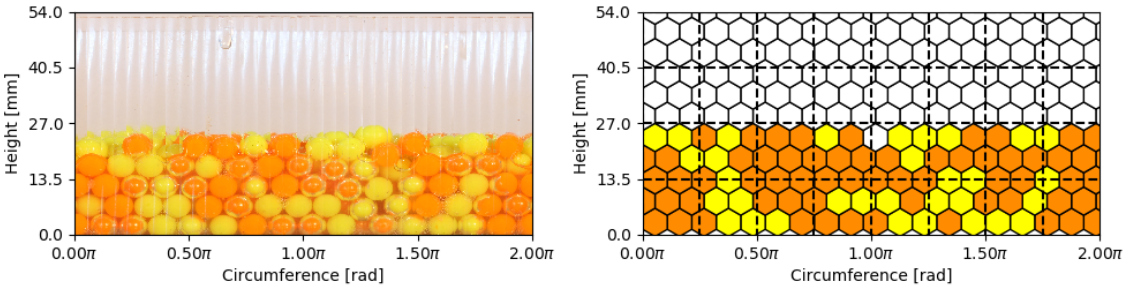
B.2 Particle Fill level Sweep

B.2.1 Particle Fill level - $\Pi_3 = 0.2$

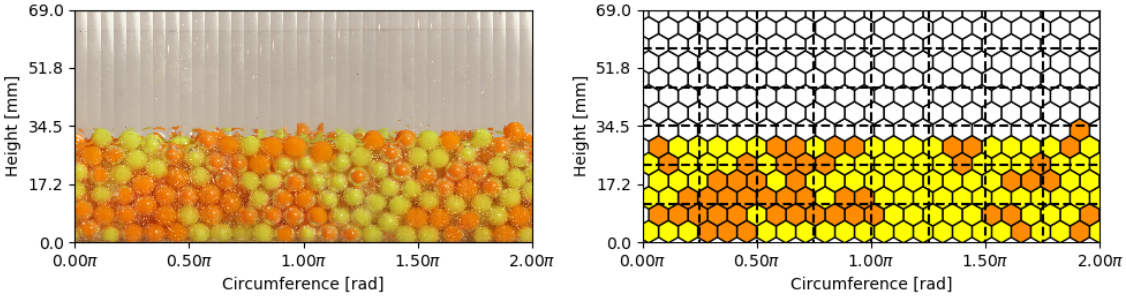


(a) Small cup

Figure B.5: $\Pi_3 = 0.2$



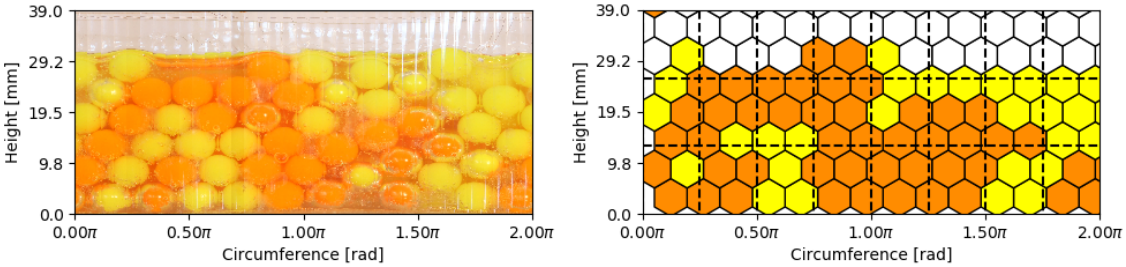
(b) Medium cup



(c) Large cup

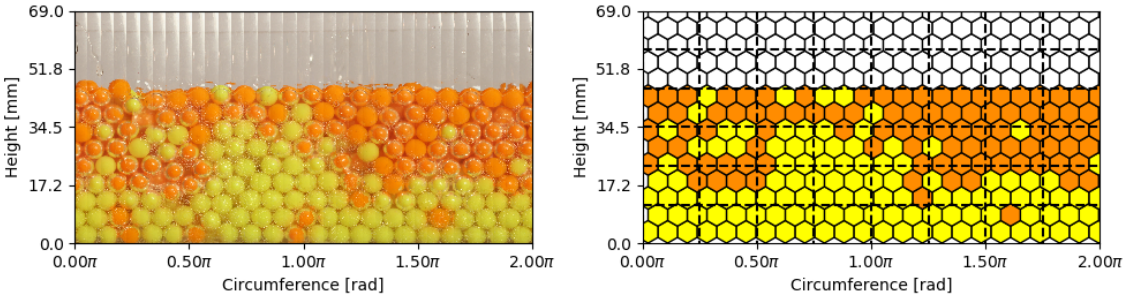
Figure B.5: $\Pi_3 = 0.2$

B.2.2 Particle Fill level - $\Pi_3 = 0.3$



(a) Small cup

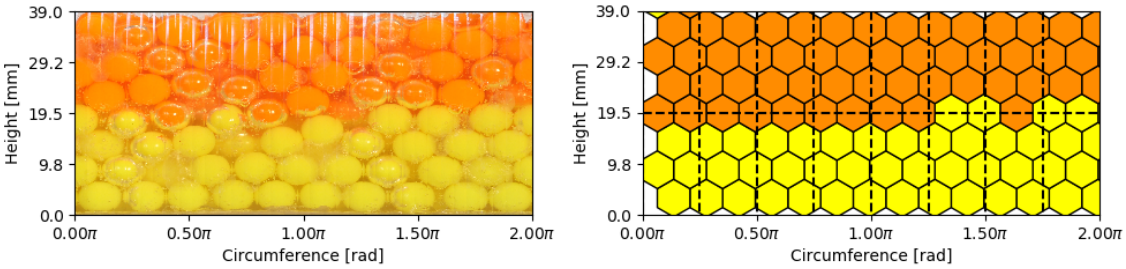
Figure B.6: $\Pi_3 = 0.3$



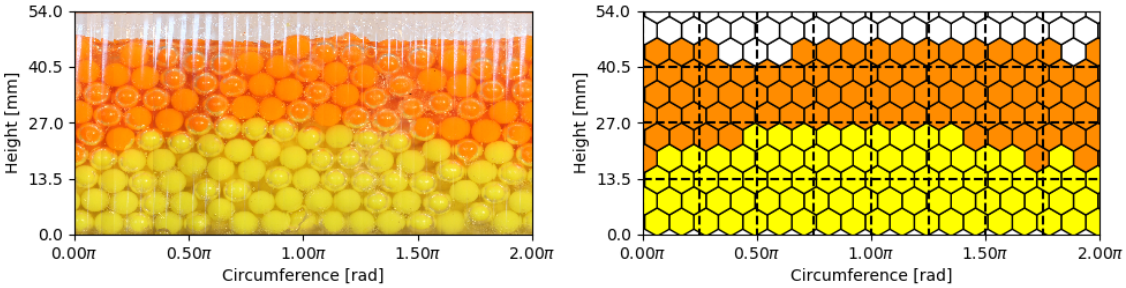
(b) Large cup

Figure B.6: $\Pi_3 = 0.3$

B.2.3 Particle Fill level - $\Pi_3 = 0.4$

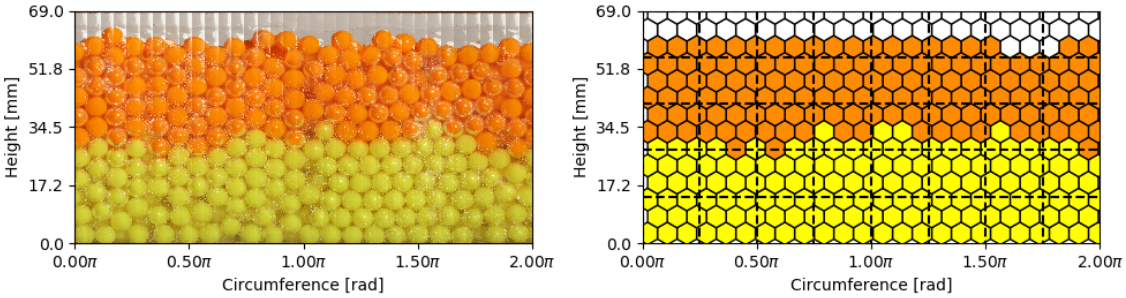


(a) Small cup



(b) Medium cup

Figure B.7: $\Pi_3 = 0.4$

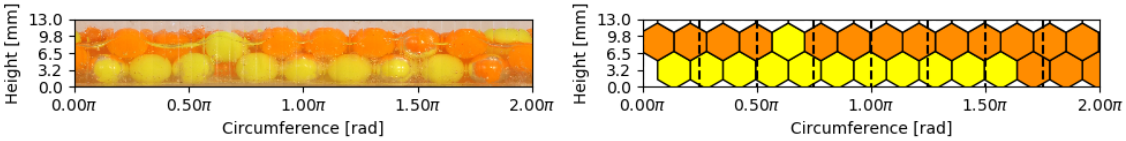


(c) Large cup

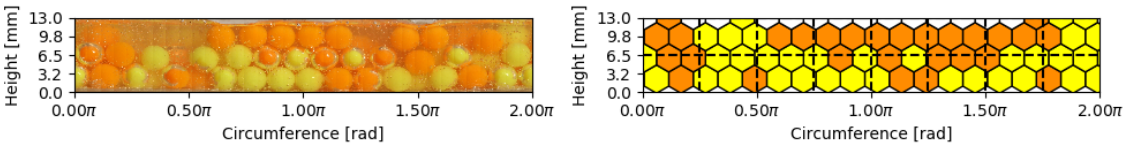
Figure B.7: $\Pi_3 = 0.4$

B.3 Volume Sweep

B.3.1 Volume ratio - $\Pi_1 = 0.5$

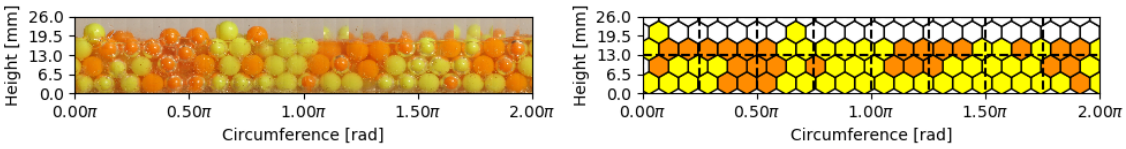


(a) Small cup



(b) Medium cup

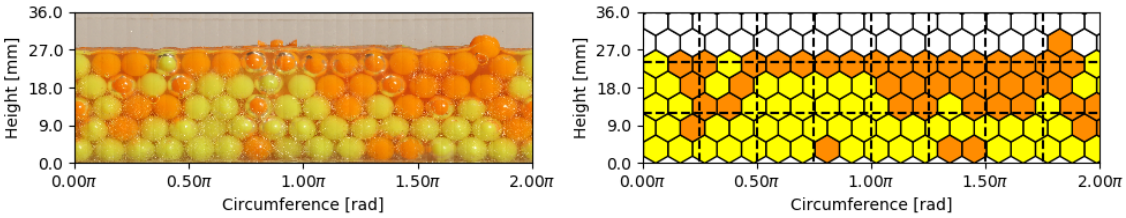
Figure B.8: $\Pi_1 = 0.5$



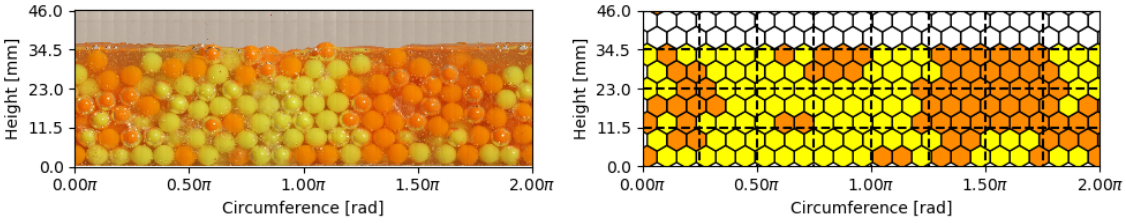
(c) Large cup

Figure B.8: $\Pi_1 = 0.5$

B.3.2 Volume ratio - $\Pi_1 = 1$



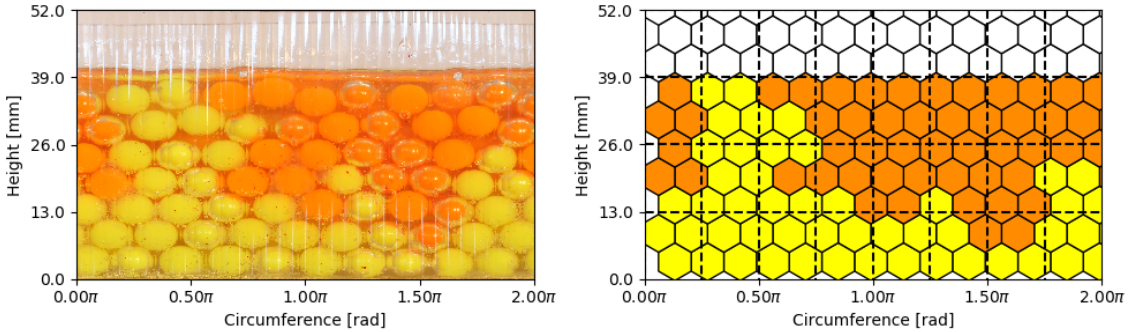
(a) Medium cup



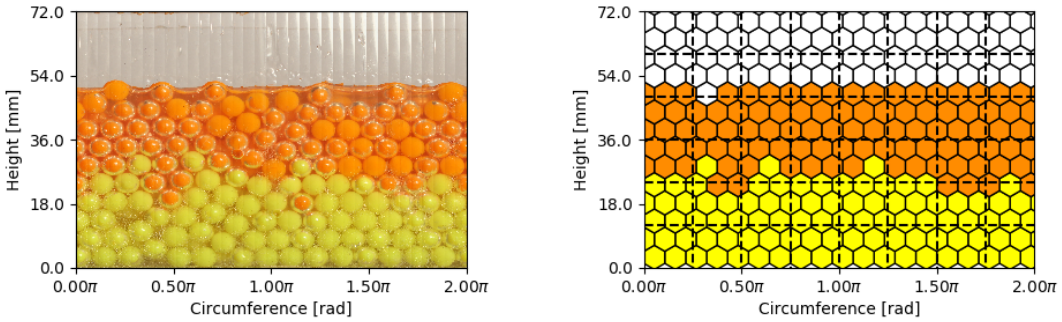
(b) Large cup

Figure B.9: $\Pi_1 = 1$

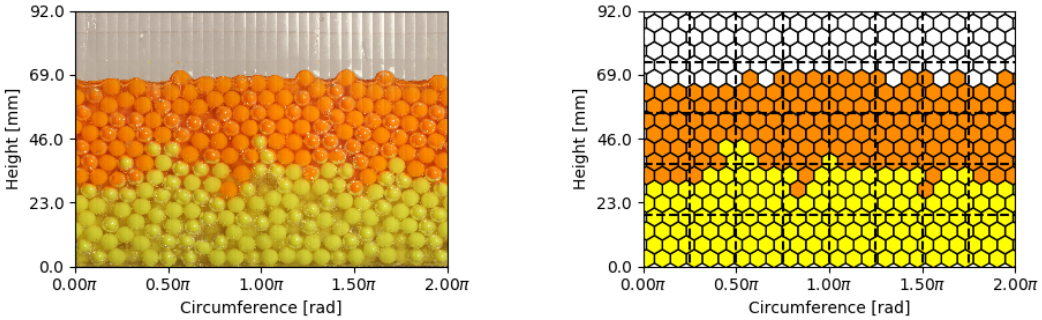
B.3.3 Volume ratio - $\Pi_1 = 2$



(a) Small cup



(b) Medium cup



(c) Large cup

Figure B.10: $\Pi_1 = 2$

Appendix C

Supporting Code

Listing C.1: Mixing Index calculation code

```
import numpy as np
import cells as c
import matplotlib.pyplot as plt
import plotparticle as pltP

'''
This module includes Mixing Index definitions, the following indexes are implemented here:
========
Average Height Method: The Average Height Method takes into account the average height of particle types. This method is very simple and easy to implement, but rudimentary and gives no information on mixing performance in other directions than the vertical direction. This method produces a semi unstable representation in the transitional stage of mixing as the particulate matter is dispersed across the volume.
--------
Nearest neighbour Method: The nearest neighbour approach considers the 12 (standard) closest particles to particle in question and counts the neighbouring particles. If more than half of the neighbouring particles is dissimilar to the particle in question the particle in question is then declared fully mixed.
--------
Neighbour distance Method: The neighbour distance method compares the distance change between two neighbouring particles - as well as the distance between randomly selected pairs.
--------
Lacey method: The Lacey mixing index is widely used and represents a mixing index based on the probabilistic behaviour of the particles.
--------
Mixing Entropy: The entropy of mixing determines the mixing index according to the Boltzmann mixing energy equation.
```

```
18 --------
19 Coordination number: Finds the Coordination Number mixing index of each
      particle in R.
20 --------
21 Particle Scale Index: Finds the Particle-scale index of particles in
      contact.
22 --------
23 Siiria Method: The Siiria method uses the distance change in the particle
       position.
24 --------
25 Proposed Methods
26 Chain Length: The chain length method concatenates various chains of
      neighbouring particles to form the neighbourhood.
27 --------
28 Growth Point: The Growth point method grows a neighbourhood from a
      particle sample, much like the MS Minesweeper game.
29 --------
30 '''
31
32 def AHM(P,**kwargs):
33         '''
34         The Average Height Method takes into account the average height
              of particle types. This method is very simple and easy to
              implement, but rudimentary and gives no information on mixing
              performance in other directions than the vertical direction.
              This method produces a semi unstable representation in the
              transitional stage of mixing as the particulate matter is
              dispersed across the volume.
35
36         Parameters
37         ----------
38         P: Particle position matrix.
39
40         **kwargs
41         --------
42         dir: Euclidean direction of interest ex. dir=0 for x-direction,
              dir=1 for y and dir=2 for z, standard setting is in the z-
              direction
43         track: Sample a specific particle type.
44
45         Returns
46         -------
47         Mixing index at input time-step.
48         '''
49         if 'dir' not in kwargs:
50                 d = 2
51         else:
52                 d = kwargs['dir']
```

```
53
54          if 'height_offset' not in kwargs:
55                  refH = 0
56          else:
57                  refH = kwargs['height_offset']
58
59          if 'track' not in kwargs:
60                  P1 = (P[:][d+1][P[:][0] == P[0][0]]) - refH
61                  P2 = (P[:][d+1][P[:][0] != P[0][0]]) - refH
62          else:
63                  trk = kwargs['track']
64                  P1 = (P[:][d+1][P[:][0] == trk]) - refH
65                  P2 = (P[:][d+1][P[:][0] != trk]) - refH
66
67          A = (1/len(P1))*sum(P1)
68          B = (1/(len(P1)+len(P2)))*(sum(P1)+sum(P2))
69          Z1 = A/B
70          return 2*(Z1-0.5)
71
72  def determine_neighbour(P,PIQ,n):
73          '''
74          This function determines the closest neighbouring particles (n)
               to a particle in question. This information is used in various
                mixing definitions that is also provided in this module.
75
76          Parameters
77          ----------
78          P: Particle positional list defined by [Particle Type vector,
               Particle X position, Particle Y position, Particle Z position]
                sorted row by row according to the particle identifier
79          PIQ: List of particles in question (0 >= PIQ >= len(P))
80          n: Number of neighbouring particles to determine
81
82          Returns
83          -------
84          List of nearest neighbours to each particle in question
               structured row by row for every PIQ with every row containing
               the neighbours indexes (sorted from closest to furthest).
85          '''
86          d = []
87          minimum = []
88          for I in range(0,len(PIQ)):
89                  i = (P[:][1] - P[:][1][PIQ[I]])**2
90                  j = (P[:][2] - P[:][2][PIQ[I]])**2
91                  k = (P[:][3] - P[:][3][PIQ[I]])**2
92                  d = np.sqrt(i+j+k)
93                  m = np.array(d).argsort()[:int(n[I])+2]
94                  minimum.append(m[1:-1].tolist())
```

```
95          return minimum
96
97  def NN(P,R,**kwargs):
98          '''
99          The nearest neighbour approach considers the 12 (standard)
               closest particles to particle in question and counts the
               neighbouring particles.     If more than half of the
               neighbouring particles are dissimilar to the particle in
               question, the particle in question is then declared fully
               mixed.
100
101         Parameters
102         ----------
103         P: Particle positional list defined by [Particle Type vector,
               Particle X position, Particle Y position, Particle Z position]
                sorted row by row according to the particle identifier
104         R: Is a list of particles that will be tracked and will determine
                the mixing. The list can be randomly generated integers to
               best describe the mixing at random locations. The more
               particles to track the better defined the mixing will be.
105
106         **kwargs
107         --------
108         nb: If the user would like to change the number of neighbouring
               particles to account one should define a keyword argument as
               follows: eg. 'nb'=10
109
110         Returns
111         -------
112         Mixing index at input timestep.
113         '''
114         A = []
115         N = len(R)
116         if 'nb' in kwargs:
117                 nnb = kwargs['nb']*np.ones(len(R))
118         else:
119                 nnb = 12*np.ones(len(R))
120         Mini = determine_neighbour(P,R,nnb)
121         for I in range(0,N):
122                 ndiff = 0
123                 PCol = P[:][0][R[I]]
124                 for J in range(0,int(nnb[I])):
125                         if PCol != P[:][0][Mini[I][J]]:
126                                 ndiff = ndiff+1
127                 A.append((2*ndiff)/nnb[I])
128         A = np.array(A)
129         return (1/N)*sum(A)
130
```

```
131 def ND(P,R,**kwargs):
132         '''
133         The neighbour distance method compares the distance change
               between two neighbouring particles, as well as the distance
               between randomly selected pairs.
134 
135         Parameters
136         ----------
137         P: Particle positional list defined by [Particle Type vector,
               Particle X position, Particle Y position, Particle Z position]
                sorted row by row according to the particle identifier.
138         R: Is a list of particles that will be tracked and will determine
                the mixing. The list can be randomly generated integers to
               best describe the mixing at random locations. The more
               particles to track the better defined the mixing will be.
139 
140         **kwargs
141         --------
142         neighbours: Neighbouring particles to the tracked particles
143         random_pairs: Random Particles associated with tracked pairs
144         p_dia: List of average particle diameter of each particle type,
               otherwise it is assumed to be zero (only taking into account
               the particle centroid)
145 
146         Returns
147         -------
148         1) Mixing index at input time-step.
149         2) Nearest Neighbouring particles (at time-step re-evaluated if
               not provided in **kwargs)
150         3) Random particles associated with tracked R particles
151         '''
152         A = []
153         B = []
154         N = len(R)
155         if 'neighbours' not in kwargs:
156         #IMPORTANT TO DETERMINE NEIGHBOURS ONLY ONCE AT THE START
157                 Mini = determine_neighbour(P,R,np.ones(N))
158         else:
159                 Mini = kwargs['neighbours']
160         if 'random_pairs' not in kwargs:
161         #ALSO DETERMINE RANDOM PAIRS
162                 RR = np.random.randint(0,len(P[0]),len(R))
163                 while RR is R:
164                         RR = np.random.randint(0,len(P[0]),len(R))
165         else:
166                 RR = kwargs['random_pairs']
167         if 'p_dia' not in kwargs:
168                 p_dia = [0,0]
```

```
169                 print('WARNING: No particle diameter information provided
                        and may produce inaccurate mixing results, add as ex.
                        p_dia = [0.003,0.003]')
170         else:
171                 p_dia = kwargs['p_dia']
172 
173         for I in range(0,N):
174                 i1 = (P[:][1][R[I]] - P[:][1][Mini[I][0]])**2
175                 j1 = (P[:][2][R[I]] - P[:][2][Mini[I][0]])**2
176                 k1 = (P[:][3][R[I]] - P[:][3][Mini[I][0]])**2
177                 a = np.sqrt(i1+j1+k1) - (p_dia[0]+p_dia[1])/2
178                 A.append(a)
179 
180                 i2 = (P[:][1][R[I]] - P[:][1][RR[I]])**2
181                 j2 = (P[:][2][R[I]] - P[:][2][RR[I]])**2
182                 k2 = (P[:][3][R[I]] - P[:][3][RR[I]])**2
183                 b = np.sqrt(i2+j2+k2) - (p_dia[0]+p_dia[1])/2
184                 B.append(b)
185         A = np.array(A)
186         B = np.array(B)
187         return sum(A)/sum(B), Mini, RR
188 
189 def cell(P,*args,**kwargs):
190         '''
191         Determines the particle fraction in each cell. The cells is
              volume 'rings' that each has identical volume measurements.
              The rings are taken from the core outwards.
192 
193         Parameters
194         ----------
195         P: Particle positional list defined by [Particle Type vector,
              Particle X position, Particle Y position, Particle Z position]
               sorted row by row according to the particle identifier
196         cD: Cup diameter [m]
197         H: Cup height [m]
198 
199         **kwargs
200         --------
201         N: Number of cells in height, standard is set at 10
202         W: Width of box or cup diameter
203         H: Height of cup or box
204 
205         Returns
206         -------
207         Cell Info of each cells, numbering starts at the core cell and
              moves out to outer cell, then moves one level up and repeats
              the process.
208         '''
```

```
209         if 'N' not in kwargs:
210                 N = 10
211         else:
212                 N = kwargs['N']
213 
214         if 'W' not in kwargs:
215                 W = float(input('Please provide the solver with the
                        correct width or diameter: '))
216         else:
217                 W = kwargs['W']
218 
219         if 'H' not in kwargs:
220                 H = float(input('Please provide the solver with the
                        correct height of the volume: '))
221         else:
222                 H = kwargs['H']
223 
224         P1 = []
225         P2 = []
226 
227         P1.append(P[0][P[:][0] == P[0][0]])
228         P1.append(P[1][P[:][0] == P[0][0]])
229         P1.append(P[2][P[:][0] == P[0][0]])
230         P1.append(P[3][P[:][0] == P[0][0]])
231 
232         P2.append(P[0][P[:][0] != P[0][0]])
233         P2.append(P[1][P[:][0] != P[0][0]])
234         P2.append(P[2][P[:][0] != P[0][0]])
235         P2.append(P[3][P[:][0] != P[0][0]])
236 
237         if 'concentric' in args:
238                 cells = c.concentric(P1,P2,N,W,H)
239         elif 'radial' in args:
240                 if 'N_rad' not in kwargs:
241                         Nr = 1
242                 else:
243                         Nr = kwargs['N_rad']
244 
245                 if 'N_slice' not in kwargs:
246                         Nc = float(input('Please provide the solver with
                                the amount of radial slices: '))
247                 else:
248                         Nc = kwargs['N_slice']
249 
250                 cells = c.radial(P1,P2,N,Nr,Nc,W,H)
251         elif 'box' in args:
252                 if 't' not in kwargs:
253                         t = float(input('Please provide the solver with
```

```
                                the correct thickness of the box: '))
254                 else:
255                         t = kwargs['t']
256
257                 if 'Nz' not in kwargs:
258                         Nz = float(input('Please provide the solver with
                               the amount of thickness cells of the box: '))
259                 else:
260                         Nz = kwargs['Nz']
261
262                 cells = c.box(P1,P2,N,Nz,W,H,t)
263         else:
264                 print('NO VALID CELL TYPE SELECTED! Aborting')
265
266         return cells
267
268 def lacey(cells):
269         '''
270         The Lacey mixing index is widely used and represents a mixing
              index based on the probabilistic behaviour of the particles.
271
272         Parameters
273         ----------
274         cells: Provide the cell info provided by cell function in this
              module
275
276         Returns
277         -------
278         Mixing index value
279         '''
280         xi = []
281         n = []
282         xm = cells[0]['xm']
283         N = len(cells)
284         for I in range(0,len(cells)):
285                 xi.append(cells[I]['xi'])
286                 n.append(cells[I]['n'])
287         xi = np.array(xi)
288         n = np.mean(n)
289         S2 = (1/(N-1))*np.sum((np.array(xi)-xm)**2)
290         S02 = xm*(1-xm)
291         SR2 = (S02)/n
292         return (S2-S02)/(SR2-S02)
293
294 def mix_entropy(cells):
295         '''
296         The entropy of mixing determines the mixing index according to
              the Boltzmann mixing energy equation
```

```
297
298         Parameters
299         ----------
300         cells: Provide the cell info provided by cell function in this
                module
301
302         Returns
303         -------
304         1) Mixing index value
305         2) Mixing index per cell
306         3) Global entropy
307         4) Cell entropy
308         '''
309         xi = []
310         xii = np.zeros(len(cells))
311         n = []
312         xm = cells[0]['xm']
313         for I in range(0,len(cells)):
314                 xi.append(cells[I]['xi'])
315                 n.append(cells[I]['n'])
316                 if n[I] > 0:
317                         xii[I] = 1-xi[I]
318         N = sum(n)
319         xi = np.array(xi)
320         xii = np.array(xii)
321         xii[xii == 0] = 1E-12
322         n = np.array(n)
323
324         e = -np.array((xi*np.log(xi)) + (xii*np.log(xii)))
325         Em = -((xm*np.log(xm)) + ((1-xm)*np.log(1-xm)))
326
327         E = (1/N)*np.nansum(e*n)
328         # e = (e*n)
329
330         M = E/Em
331         m = e/Em
332         return M, m, E, e
333
334 def coor_num(P,R,p_dia):
335         '''
336         Finds the Coordination Number mixing index of each particle in R
337
338         Parameters
339         ----------
340         P: Particle positional list defined by [Particle Type vector,
               Particle X position, Particle Y position, Particle Z position]
                sorted row by row according to the particle identifier
341         R: Is a list of particles that will be tracked and will determine
```

```
            the mixing. The list can be randomly generated integers to
            best describe the mixing at random locations. The more
            particles to track the better defined the mixing will be.
        p_dia: Average particle diameter of each particle in R.

        Returns
        -------
        1) Mixing Index
        2) Contact on each particle
        '''
        nnb = []
        cont = []
        for I in range(0,len(R)):
                i = (P[:][1] - P[:][1][R[I]])**2
                j = (P[:][2] - P[:][2][R[I]])**2
                k = (P[:][3] - P[:][3][R[I]])**2
                D = np.sqrt(i+j+k) - p_dia[I]
                nnb.append(sum(D < 0.1*p_dia[I]))
                cont.append(D < 0.1*p_dia[I])
        return NN(P,R,nb=nnb), cont


def psi(P,p_dia,**kwargs):
        '''
        Finds the Particle-scale index of particles in contact

        Parameters
        ----------
        P: Particle positional list defined by [Particle Type vector,
            Particle X position, Particle Y position, Particle Z position]
             sorted row by row according to the particle identifier
        p_dia: Average particle diameter of each particle in R.

        **kwargs
        --------
        cont: Uses the contacts solved in the coor_num function, if not
            provided the function samples a three layer cut from all the
            particles in P eg. cutsize = 0.3 (first 10%, middle 10%, last
            10%).
        cutsize: The sample fraction of the total set size to resolve
            contact from. Default is 0.30 (30%).

        Returns
        -------
        Mixing Index
        '''
        CnB = []
        Cn = []
```

```
381         if 'cont' not in kwargs:
382                 cont = []
383                 if 'cutsize' not in kwargs:
384                         cutsize = 0.1
385                 else:
386                         cutsize = kwargs['cutsize']/3
387 
388                 cut = np.floor(cutsize*len(P[0][:]))
389                 indeces = []
390                 indeces = np.linspace(0,cut,cut,dtype=int).tolist()
391                 midbegin = np.floor(len(P[0][:])/2 - cut/2)
392                 indeces = indeces + np.linspace(midbegin,midbegin+cut,cut
                       ,dtype=int).tolist()
393                 endbegin = len(P[0][:])-cut-1
394                 indeces = indeces + np.linspace(endbegin,endbegin+cut,cut
                       ,dtype=int).tolist()
395 
396                 for I in range(0,len(indeces)):
397                         i = (P[1][:] - P[1][indeces[I]])**2
398                         j = (P[2][:] - P[2][indeces[I]])**2
399                         k = (P[3][:] - P[3][indeces[I]])**2
400                         D = np.sqrt(i+j+k) - p_dia[I]
401                         cont.append(D < 0.1*p_dia[I])
402         else:
403                 cont = kwargs['cont']
404 
405         for I in range(0,len(cont)):
406                 CnB.append(sum(P[0][cont[I]] != P[0][0]))
407                 Cn.append(sum(cont[I][:]))
408         pI = np.array(CnB)/(np.array(Cn) + 1)
409         St2 = (1/len(pI))*np.sum((pI-np.mean(pI))**2)
410         p = sum(P[0] != P[0][0])/len(P[0])
411         n = np.mean(Cn) + 1
412         S02 = p*(1-p)
413         SR2 = S02/n
414         return (St2 - S02)/(SR2 -S02)
415 
416 def siiria(P,r,g,**kwargs):
417         '''
418         The Siiria method uses the distance change in the particle
               position.
419 
420         Parameters
421         ----------
422         P: Particle positional list defined by [Particle Type vector,
               Particle X position, Particle Y position, Particle Z position]
                sorted row by row according to the particle identifier
423         r: The scalable constant between 0 and 1.
```

```
        g: Another scalable constant to calibrate the length dimension.

        **kwargs
        --------
        N: Amount of particle pairs to solve. Default N=20
        pairs: Can choose pairs manually and insert here. The pairs
           should only be selected once, so input this parameter in the
           next time-steps. The pair's matrix is a square matrix with the
            particle in question on the diagonal.
        xij: Add the initial conditions towards the first pairs selected.

        Returns
        -------
        1) Mixing Index
        2) Initial conditions
        3) Pairs Matrix
        '''
        I = 0
        if 'N' not in kwargs:
                N = len(P[0])
        else:
                N = kwargs['N']
        if 'pairs' not in kwargs:
                pairs = np.random.randint(0,len(P[0]),size=(N,N))
        else:
                pairs = kwargs['pairs']
        if 'xij' not in kwargs:
                xij = np.zeros([N,N])
                for I in range(0,N):
                        for J in range(0,N):
                                i = (P[1][pairs[I][I]] - P[1][pairs[I][J
                                   ]])**2
                                j = (P[2][pairs[I][I]] - P[2][pairs[I][J
                                   ]])**2
                                k = (P[3][pairs[I][I]] - P[3][pairs[I][J
                                   ]])**2
                                xij[I][J] = np.sqrt(i+j+k)
        else:
                xij = kwargs['xij']
        if 'S0' not in kwargs:
                S0 = 1
        else:
                S0 = kwargs['S0']

        xim = np.zeros([N,N])
        Mij = np.zeros([N,N])
        M = []

```

```
466         for I in range(0,N):
467                 for M in range(0,N):
468                         i = (P[1][pairs[I][I]] - P[1][pairs[I][M]])**2
469                         j = (P[2][pairs[I][I]] - P[2][pairs[I][M]])**2
470                         k = (P[3][pairs[I][I]] - P[3][pairs[I][M]])**2
471                         xim[I][M] = np.sqrt(i+j+k)
472                         Mij[I][M] = (r**np.abs(g*xij[I][M]))*(1-r**(g*np.
                               abs(xij[I][M] - xim[I][M])))
473         MIJ = np.array(Mij)
474         MIJ = np.sum(MIJ, axis=0)
475         M = (np.sum(MIJ)/(N**2))/S0
476         return M, xij, pairs
477
478 def chain_len(P,R,**kwargs):
479         '''
480         The Chain Length method concatenates various chains of
               neighbouring particles to form the neighbourhood.
481
482         Parameters
483         ----------
484         P: Particle positional list defined by [Particle Type vector,
               Particle X position, Particle Y position, Particle Z position]
                sorted row by row according to the particle identifier
485         R: Is a list of particles that will be tracked and will determine
                the mixing. The list can be randomly generated integers to
               best describe the mixing at random locations. The more
               particles to track the better defined the mixing will be.
486
487         **kwargs
488         --------
489         num_neig: The contact number that will be used to check
               neighbouring particles.
490         step: Step length to search for neighbouring particles, default
               is 110% of first step taken.
491
492         Returns
493         -------
494         1) Mixing Index
495         2) Particles in the neighbourhood
496         3) Chain length fractions
497         '''
498         if 'num_neig' not in kwargs:
499                 num_neig = 12
500         else:
501                 num_neig = kwargs['num_neig']
502
503         def cal_len(old,new):
504                 i = (P[1][new] - P[1][old])**2
```

```
505                 j = (P[2][new] - P[2][old])**2
506                 k = (P[3][new] - P[3][old])**2
507                 return np.sqrt(i+j+k)
508
509         if 'step' not in kwargs:
510                 N = determine_neighbour(P,[R[0]],[1])
511                 step = 1.1*cal_len(N[0],R[0])
512         else:
513                 step = kwargs['step']
514
515         def path(N,track):
516                 maxi = []
517                 for I in range(0,len(N[0])):
518                         new = determine_neighbour(P,[N[0][I]],[num_neig])
519                         A = []
520                         for J in range(0,len(new[0])):
521                                 A.append((new[0][J] not in N[0]) and (new
                                    [0][J] not in track))
522                         maxi.append(sum(A))
523
524                 NewN = []
525                 for J in range(0,len(maxi)):
526                         NewN.append(N[0][maxi.index(max(maxi))])
527                         maxi[maxi.index(max(maxi))] = 0
528                 return [NewN]
529         tracks = []
530         S = []
531         for I in range(0,len(R)):
532                 track = []
533                 track.append(R[I])
534                 cnt = 0
535                 DONE = False
536                 while DONE is False:
537                         N = determine_neighbour(P,[track[cnt]],[num_neig
                            ])
538                         np.random.shuffle(N[0])
539                         N = path(N,track)
540                         MADE = False
541                         J = 0
542                         while MADE is False:
543                                 if (int(P[0][track[cnt]]) is int(P[0][N
                                    [0][J]])) and\
544                                  (step > cal_len(track[cnt],N[0][J])) and
                                     \
545                                  (N[0][J] not in track):
546                                         cnt += 1
547                                         for K in range(0,len(tracks)):
548                                                 if N[0][J] in tracks[K]:
```

```
                                                        track.extend(
                                                           tracks[K])
                                                        track.remove(N
                                                           [0][J])
                                                        del(tracks[K])
                                                        break
                                            track.append(N[0][J])
                                            MADE = True
                                    elif J is len(N[0])-1:
                                            DONE = True
                                            MADE = True
                                    J += 1
                        tracks.append(track)
                for T in range(0,len(tracks)):
                        num = np.sum(P[0][:] == P[0][tracks[T][0]])
                        S.append((num-len(tracks[T]))/num)
                M = np.mean(S)
                return M, tracks, S

def growth_point(P,R,**kwargs):
        '''
        Growth Point: The Growth point method grows a neighbourhood from
           a particle sample, much like the MS Minesweeper game.

        Parameters
        ----------
        P: Particle positional list defined by [Particle Type vector,
           Particle X position, Particle Y position, Particle Z position]
            sorted row by row according to the particle identifier
        R: Is a list of particles that will be tracked and will determine
            the mixing. The list can be randomly generated integers to
           best describe the mixing at random locations. The more
           particles to track the better defined the mixing will be.

        **kwargs
        --------
        num_neig: The contact number that will be used to check
           neighbouring particles.
        step: Step length to search for neighbouring particles, default
           is 110% of first step taken.

        Returns
        -------
        1) Mixing Index
        2) Particles fraction per neighbourhood
        3) Particles added at each iteration
        4) Average number of particles added per sample point
        '''
```

```
587
588         if 'num_neig' not in kwargs:
589                 num_neig = 12
590         else:
591                 num_neig = kwargs['num_neig']
592         def cal_len(old,new):
593                 i = (P[1][new] - P[1][old])**2
594                 j = (P[2][new] - P[2][old])**2
595                 k = (P[3][new] - P[3][old])**2
596                 return np.sqrt(i+j+k)
597         if 'step' not in kwargs:
598                 N = determine_neighbour(P,[R[0]],[1])
599                 step = 1.1*cal_len(N[0],R[0])
600         else:
601                 step = kwargs['step']
602         LARGE = []
603         M = []
604         ITR = []
605         CHKR = []
606         for I in range(0,len(R)):
607                 DONE = False
608                 spawn = []
609                 spawn.append(R[I])
610                 N = np.array(determine_neighbour(P,[R[I]],[num_neig]))
611                 A = P[0][N[0][:]] == int(P[0][R[I]])
612                 B = cal_len(R[I],N[0]) < step
613                 C = np.invert(np.in1d(N[0], spawn))
614                 D = A & B & C
615                 new = N[0][D]
616                 spawn.extend(new)
617                 itr = 1
618                 CHK = []
619                 CHK.append(len(new))
620                 while DONE is False:
621                         chk = []
622                         for J in range(0,len(new)):
623                                 N = np.array(determine_neighbour(P,[new[J
                                        ]],[num_neig]))
624                                 A = P[0][N[0][:]] == int(P[0][R[I]])
625                                 B = cal_len(new[J],N[0]) < step
626                                 C = np.invert(np.in1d(N[0], spawn))
627                                 D = A & B & C
628                                 each = N[0][D]
629                                 chk.append(len(each))
630                                 spawn.extend(each)
631                         if (not chk) or (int(np.max(chk)) is 0):
632                                 DONE = True
633                         else:
```

```
                                    new = spawn[-np.sum(chk):]
                            if 'see' in kwargs:
                                    plt.clf()
                                    W = kwargs['see'][0]
                                    H = kwargs['see'][1]
                                    p_dia = kwargs['see'][2]
                                    n = kwargs['see'][3]
                                    pltP.plotter(P,W,H,p_dia,n)
                                    plt.plot(P[1][R[I]],P[3][R[I]],'xg')
                                    plt.plot(P[1][spawn[1:]],P[3][spawn[1:]],
                                       'xk')
                                    plt.pause(0.005)
                            itr += 1
                            CHK.append(len(new))
                    #plt.plot(CHK)
                    #plt.pause(0.01)
                    CHKR.append(np.mean(CHK))
                    ITR.append(itr)
                    LARGE.append(spawn)
                    num = np.sum(P[0][:] == P[0][R[I]])
                    M.append((num-len(spawn))/num)
            MIX = np.mean(M)
            return MIX, M, LARGE, CHKR
```