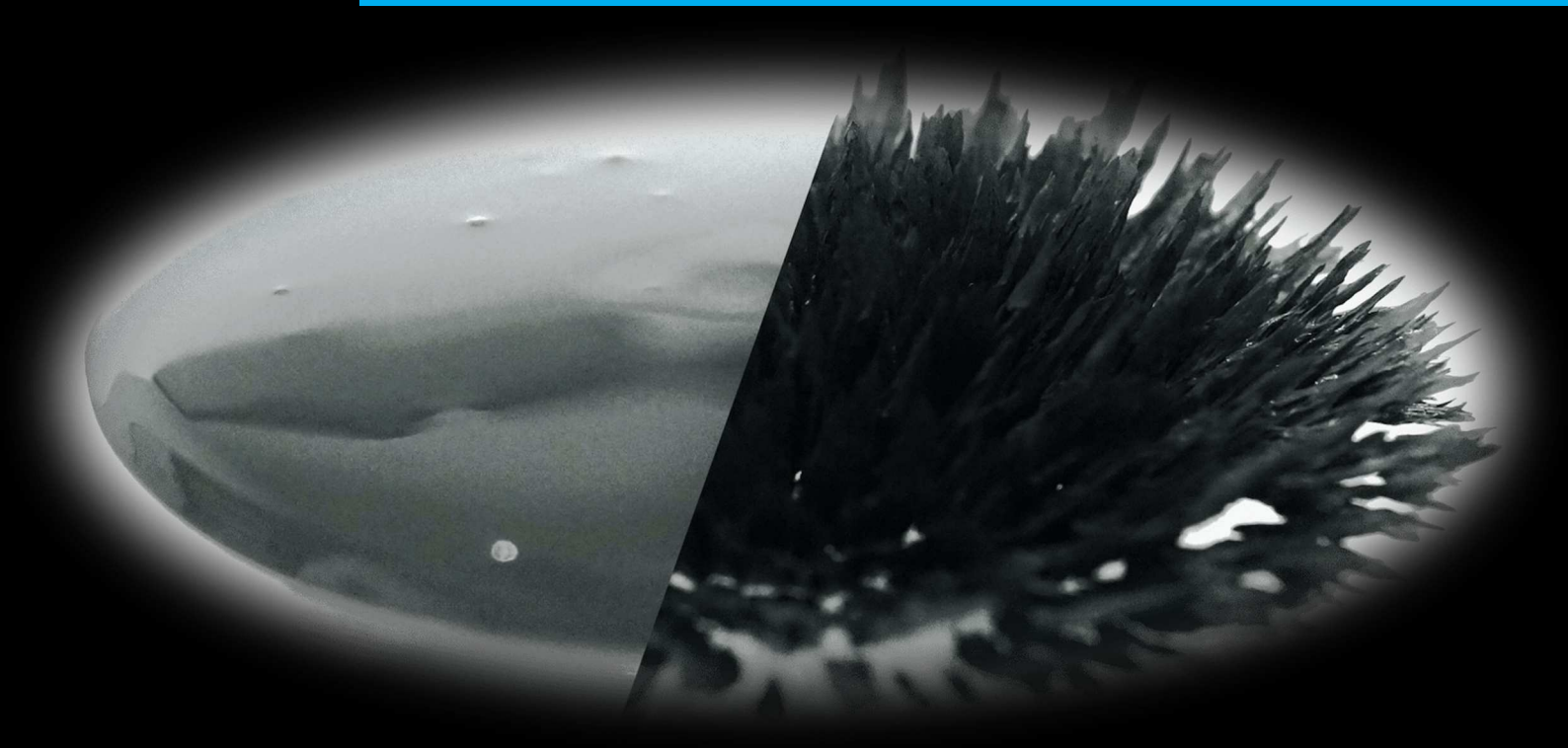
Department of Precision and Microsystems Engineering

The transition from structures to textures in magnetorheological fluids

T.L. van Kuik

Report no : 2022.079
Coach : Ir. G.H.G. van der Meer
Professor : Dr.ir. R.A.J. van Ostayen
Specialisation : Mechatronic System Design
Type of report : Master of Science Thesis
Date : 6 December 2022



TU Delft Delft University of Technology

Challenge the future

The transition from structures to textures in magnetorheological fluids

by

T.L. van Kuik

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Tuesday December 6, 2022 at 10:30 AM.

Student number:	4544986	
Project duration:	September, 2021 – December, 2022	
Thesis committee:	Dr. ir. R.A.J. van Ostayen,	TU Delft, supervisor
	Dr. ir. W.-P. Breugem,	TU Delft
	Dr. L. Botto	TU Delft
	Ir. G.H.G. van der Meer	TU Delft

This thesis is confidential and cannot be made public until December 6, 2024.

Cover: A magnetorheological fluid without a magnetic field applied and with a non-uniform magnetic field applied [1].

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Preface

Hereby I present to you the master thesis *The transition from structures to textures in magnetorheological fluids* to obtain the degree of Master of Science at the Delft University of Technology. This thesis will go over the research I have done into magnetorheological fluids (MRFs) from September, 2021 to December, 2022.

This thesis is focused on the discrete modeling of MRFs. Since the start of my Mechanical Engineering Bachelor at the Delft University of Technology I discovered that I have affinity for programming and modeling problems in and related to the Mechanical Engineering field. I was able to use these models in projects that helped me grow as an engineer. The thesis that I present required skills that I did not yet have or had learned during my Bachelors and Masters. I had some experience with COMSOL, MATLAB, SOLIDWORKS and Simulink, but my thesis required some new skills. First and foremost, to construct the models that are discussed in my thesis, I learned how to use Fortran, OpenMPI and computer clusters. Furthermore, during my thesis I decided to learn Linux to increase my efficiency.

I would like to thank my daily supervisor, G.H.G. van der Meer. The weekly discussions that we had helped me turn my thesis into a project I can be proud of.

I would also like to thank my supervisor R.A.J van Ostayen for the guidance during my thesis.

Another thanks goes out to W.-P. Breugem for taking the time to help me understand his work and modify it so that it could be used for my thesis.

I want to thank R.A.J van Ostayen, W.-P. Breugem, L. Botto and G.H.G van der Meer for making time to be a member of my graduation committee.

I'd like to give special thanks to my girlfriend J.M. Roest for supporting me during my studies. Your encouragements and reminders to take breaks helped me to stay fresh and keep working to a successful graduation. I will do my best to return the favor during your final years at the Delft University of Technology.

Another special thanks goes out to H.V. Schotman. During my thesis we have spent a lot of time together at the Delft University of Technology. Our discussions (and tea sessions) helped me greatly during this time.

Finally I would like to thank my friends and family. You have supported me throughout my journey at the Delft University of Technology. Thanks for showing interest in my studies, attending project design contests and just overall being there for me.

T.L. van Kuik
Delft, December 2022

Summary

Hydrodynamic bearings can be of use to reduce friction between two moving surfaces. These bearings utilize a thin film of pressurized fluid to support a load on the shaft. The load that the bearing can withstand depends on the pressure of the fluid film. This pressure can be increased by adding textures to the surface of the bearing. Due to the surfaces being separated, there is not a lot of friction between them which results in almost no wear during operation. At the starting and stopping, the film is not fully formed which results in contact and thus wear of the textures.

Using a magnetorheological fluid (MRF) in the hydrodynamic bearing provides a way to improve the lifetime of a texture. These fluids are suspensions of magnetizable particles in a carrier fluid. In the presence of a magnetic field, these particles will accumulate and form chain like structures that resist flow. Applying a non-uniform magnetic field generated by a permanent magnet, results in particles separating from the fluid, aggregating on the magnet, thus forming physical textures. Particles that are sheared off these textures will return into the fluid. The sheared off particles are replaced by other particles in the fluid, resulting in self-healing behavior.

Current knowledge of these textures states that the particles are pulled from the fluid due to the attractive force of the permanent magnets. It is not known how large these forces need to be to form the textures. This thesis will research how the textures are formed when the MRF is used in a non-uniform magnetic field and the effect they have on the behavior of the fluid.

Three discrete element models are constructed, which are inspired by the discrete element methods used for MRFs and particulate flows. A single core model is able ro simulate the MRF in uniform and non-uniform magnetic field. This model can only simulate small domains. The fixed- and mutual dipole models are used to determine the behavior of the MRF using this model. A parallel discrete element model is developed that can simulate domains larger than the magnet's dimension in the direction of flow. Using this model the formation of textures on the magnet can be simulated. Finally, a particulate flow model from Breugem was modified to simulate magnetizable particles. This model was used to verify the other two models and determine whether two-way coupling between the particles and the fluid is required to determine the steady state behavior of an MRF.

The first model showed that the textures are not formed due to the attractive force of the permanent magnets by itself, the model was not able to show what does influence the formation of textures. It only showed that accumulations happen in the simulation domain due to the deformation of the particle structures. Furthermore, the model was able to show differences in the behavior of the MRF in different non-uniform magnetic fields. Fields that are orientated into the direction of flow will result in higher shear stresses in the fluid. The parallel discrete element model was able to show that particles aggregate at the back of the magnet and that the size of the aggregation depends on the shear rate and the remanence of the magnets. For longer simulation times the model showed that the particles migrate from to the larger gradient in the magnetic field, as shown in figure 1. This simulation also showed that particles aggregate at the front of the magnet and amass after the magnet when a shear rate is applied, as shown in figure 2.

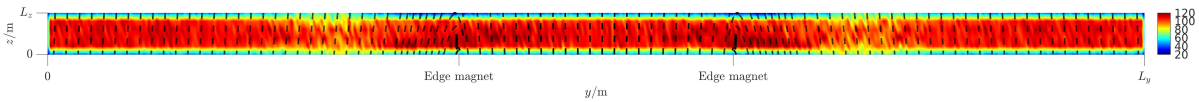


Figure 1: Density plot when $\dot{\gamma} = 0\,\mathrm{s}^{-1}$ for the increased t^{final} using bar magnet 1
Color bar: Number of particles per $20R \times 6R\,\mathrm{m}^2$
Dashed lines: Magnetic field

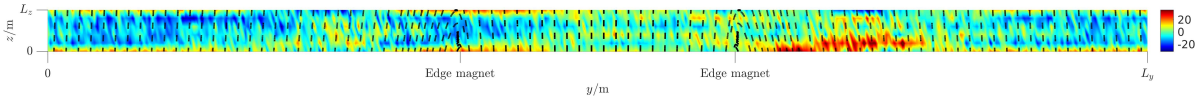


Figure 2: Density differences with respect to figure 1 where $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$ using bar magnet 1
Color bar: Difference in number of particles per $20R \times 6R$ m^2
Dashed lines: Magnetic field

The model showed that the shear stresses in the MRF mainly depend on the dimension of the magnet in the direction of the flow. Furthermore, for the simulated magnets and shear rates, it was determined that the viscoplastic models can still be used to approximate the behavior of the MRF, as shown in figure 3.

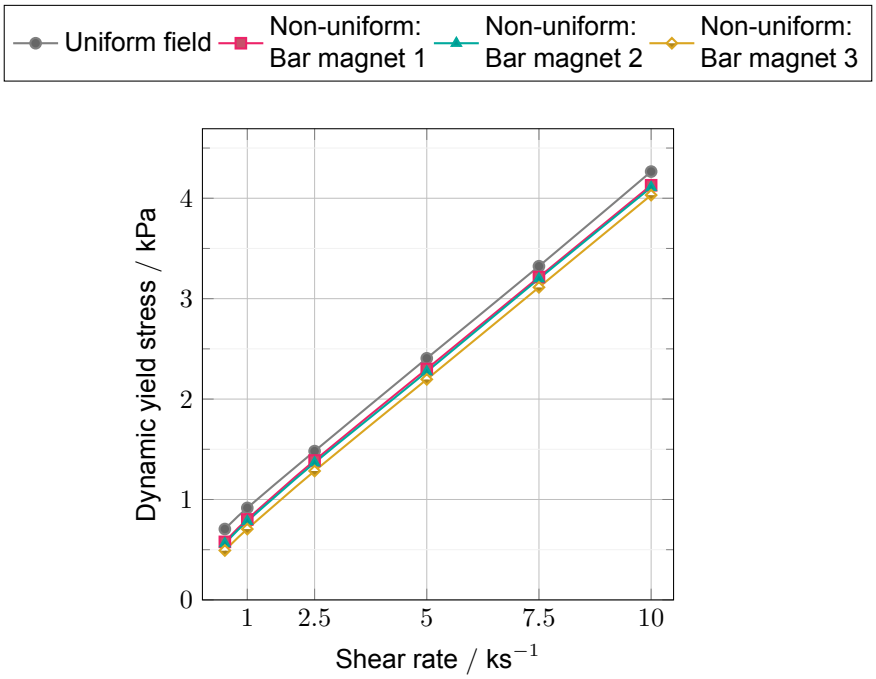


Figure 3: Rheological models for the simulated magnetic fields
Bar magnet 1: $5\,\mathrm{mm} \times 5\,\mathrm{mm} \times 2\,\mathrm{mm}$ (N52)
Bar magnet 2: $6\,\mathrm{mm} \times 5\,\mathrm{mm} \times 2\,\mathrm{mm}$ (N45)
Bar magnet 3: $8\,\mathrm{mm} \times 4\,\mathrm{mm} \times 2\,\mathrm{mm}$ (N45)

The two models were verified using Breugem's model. This model showed that the steady state of the MRF is postponed due to the two-way coupling. The two-way coupling does not affect the steady state shear stress of the MRF as shown in figure 4.

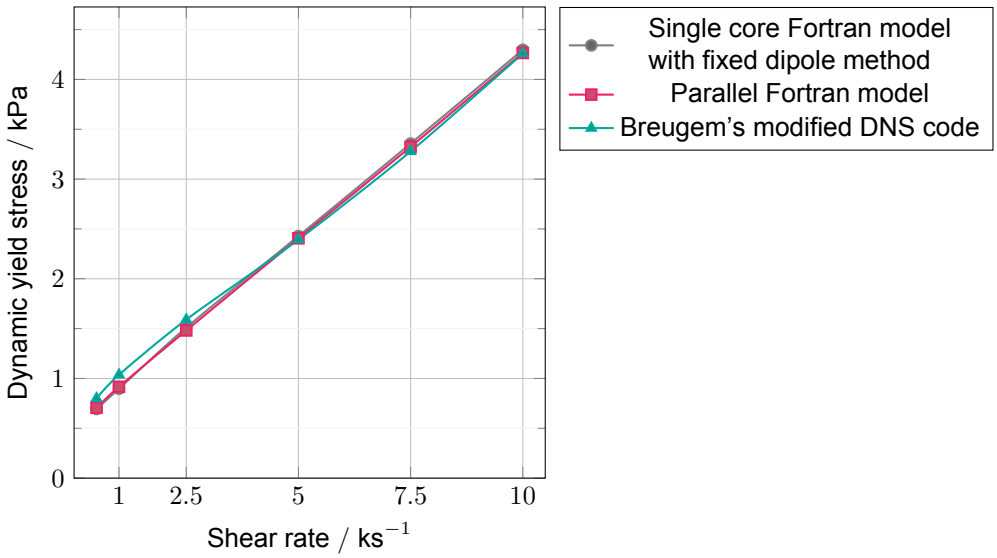


Figure 4: Verification of the models with Breugem's modified model

For future research it is recommended to use these models to construct viscoplastic models for the MRF in different non-uniform fields. The viscoplastic models could be of use for the design of hydrodynamic bearings. Next to that, the models could be used to determine the dimensions of the aggregations that form in non-uniform fields when a shear rate is applied.

Contents

Preface i

Summary ii

Nomenclature viii
Acronyms . . . viii
Symbols . . . viii

1 Introduction 1
1.1 Background . . . 1
1.2 Problem definition & research objective . . . 2
1.3 Structure of the thesis . . . 2

2 Method 3
2.1 Discrete element modeling . . . 3
2.1.1 Contact force . . . 4
2.1.2 Non-contact forces . . . 5
2.1.3 Particle fluid interaction . . . 8
2.1.4 External forces . . . 9
2.1.5 Dynamic evolution . . . 10
2.1.6 Integration scheme . . . 12
2.1.7 Boundary conditions . . . 13
2.1.8 Two-way coupled . . . 14
2.1.9 Slab profiles . . . 14
2.2 Single core Fortran models . . . 15
2.2.1 Details of the models . . . 15
2.2.2 Parameters . . . 16
2.2.3 Stability . . . 16
2.2.4 Number of particles . . . 17
2.2.5 Shear stress . . . 18
2.2.6 Verification . . . 18
2.3 Parallel Fortran model using OpenMPI . . . 20
2.3.1 Domain decomposition . . . 20
2.3.2 Details of the model . . . 22
2.3.3 Minimum number of cores . . . 23
2.3.4 Cores vs speed . . . 23
2.4 Breugem's DNS code with magnetic particles . . . 25
2.4.1 Modification . . . 25
2.4.2 Details of the model . . . 25

3 Results 26
3.1 Single core Fortran models . . . 26
3.1.1 Steady state . . . 26
3.1.2 Steady state structures . . . 27
3.1.3 Shear stresses . . . 30
3.1.4 Rheological models . . . 31
3.2 Parallel Fortran model using OpenMPI . . . 33
3.2.1 Steady state structures . . . 33
3.2.2 Shear stresses . . . 39
3.2.3 Rheological models . . . 39
3.3 Breugem's DNS code with magnetic particles . . . 40

3.3.1 Steady state structures . . . 40
3.3.2 Shear stress . . . 41
3.3.3 Rheological model . . . 41

4 Discussion **43**
4.1 Single core Fortran model . . . 43
4.1.1 Structure formation . . . 43
4.1.2 Rheological models . . . 44
4.2 Parallel Fortran model using OpenMPI . . . 45
4.2.1 Formation of planes . . . 45
4.2.2 Aggregation on the magnet . . . 45
4.2.3 Increased t^{final} . . . 46
4.2.4 Rheological models . . . 46
4.3 Breugem's DNS code with magnetic particles . . . 46

5 Conclusion **47**
5.1 Conclusions from literature review . . . 47
5.2 Verification of the models . . . 48
5.3 Aggregation of the particles . . . 48
5.4 Rheological models . . . 48
5.5 Recommendations . . . 49

References **50**

The transition from structures to textures in magnetorheological fluids: A literature review **54**

The transition from structures to textures in magnetorheological fluids **76**

A Lubrication model **87**

B Inclusion model **88**

C External magnetic field **90**
C.1 Original model . . . 90
C.2 Rotated model . . . 91
C.3 Verification . . . 91
C.4 Magnetic fields of the bar magnets . . . 95
C.4.1 Bar magnet 1 . . . 95
C.4.2 Bar magnet 2 . . . 95
C.4.3 Bar magnet 3 . . . 95

D MATLAB model **96**

E Single core Fortran code **97**
E.1 Source code . . . 97
E.1.1 Fixed dipole method in a uniform magnetic field . . . 97
E.1.2 Mutual dipole method . . . 104
E.1.3 Non-uniform magnetic field . . . 104
E.1.4 Compiling and running the code . . . 108
E.2 Number of particles . . . 109
E.3 Results . . . 110
E.3.1 Steady state structures . . . 110
E.3.2 Dynamic evolution . . . 126
E.3.3 Rheological model . . . 130
E.3.4 Slab volume fractions and velocities . . . 131

F Parallel Fortran model **135**
F.1 Source code . . . 135
F.1.1 Uniform magnetic field . . . 135
F.1.2 Non-uniform magnetic field . . . 155
F.1.3 Compiling and running the code . . . 156
F.2 Results . . . 158

F.2.1 Dynamic evolution . . . 158
F.2.2 Rheological models . . . 162
F.2.3 Slab volume fractions and velocities . . . 163
F.2.4 Density plots . . . 165

G Breugem's DNS code with magnetic particles 174
G.1 Results . . . 174
G.1.1 Steady state structures . . . 174
G.1.2 $\dot{\gamma} = 500\,\mathrm{s}^{-1}$ . . . 174
G.1.3 $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$ . . . 175
G.1.4 $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$ . . . 175
G.1.5 $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$ . . . 176
G.1.6 $\dot{\gamma} = 7500\,\mathrm{s}^{-1}$ . . . 176
G.1.7 $\dot{\gamma} = 10\,000\,\mathrm{s}^{-1}$ . . . 177
G.1.8 Shear stress response . . . 177
G.1.9 Rheological model . . . 178

H Recommendations 179
H.1 Integrating the mutual dipole method . . . 179
H.2 Determine the texture height . . . 179

Nomenclature

Acronyms

Acronym	Description
DHPC	Delft High Performance Computing Centre
DNS	Direct Numerical Simulation
Fortran	Formula translating system
IBM	Immersed Boundary Method
MRF	magnetorheological fluid
OpenMPI	Open Message Passing Interface
PBC	Periodic Boundary Condition

Symbols

Symbol	Description	Unit
Latin		
A	Surface	m^2
a	Particle a	1
$\boldsymbol{a}$	Acceleration vector	m
B	Magnetic flux density	T
$\boldsymbol{B}$	Magnetic flux density vector	T
b	Particle b	1
d	Wall d	1
E	Young's modulus	Pa
$\boldsymbol{e}$	Euclidian vector	1
F	Factors for the magnetic field of a bar magnet	1
$\boldsymbol{F}$	Force vector	N
H	Magnetic field strength	$\mathrm{A\,m}^{-1}$
$\boldsymbol{H}$	Magnetic field strength vector	$\mathrm{A\,m}^{-1}$
I	Moment of inertia	$\mathrm{kg\,m}^2$
$\boldsymbol{I}$	Identity matrix	1
i	Index notation i	1
j	Index notation j	1
K	Kinetic energy	J
k	Stiffness	$\mathrm{N\,m}^{-1}$
L	Length	m
l	Multipole degree of the spherical expansion of particle a	1
M	Mass	kg
$\boldsymbol{M}$	Magnetization vector field	$\mathrm{A\,m}^{-1}$
$\boldsymbol{m}$	Magnetic moment	$\mathrm{A\,m}^2$
m	Order of the spherical expansion	1
Mn	Mason number	1
N	Number of	1

p	Pressure	Pa
Pe	Peclet number	1
$P_l\left(\cos\theta_a^{\mathsf{p}}\right)$	Legendre polynomial	1
$P_l^m\left(\cos\theta_a^{\mathsf{p}}\right)$	Associated Legendre polynomial	1
$\boldsymbol{q}$	Magnetic moment	1
R	Particle radius	m
$\boldsymbol{R}$	Rotation matrix	1
r	Position	m
$\boldsymbol{r}$	Position vector	m
Re	Reynolds number	1
$\hat{\boldsymbol{r}}$	Unit position vector	1
s	Multipole degree of the spherical expansion of particle b	1
T	Temperature	°C
t	Time	s
V	Volume	$\mathrm{kg\,m^{-3}}$
v	Scalar velocity	$\mathrm{m\,s^{-1}}$
$\boldsymbol{v}$	Velocity vector	$\mathrm{m\,s^{-1}}$
$\bar{v}$	Mean velocity	m
X	Index notation of $\boldsymbol{X}$	1
$\boldsymbol{X}$	$2N^{\mathsf{m}} \times 2N^{\mathsf{m}}$ matrix for the inclusion model	1
x	x position	m
$\hat{\boldsymbol{x}}$	Unit vector in x-direction	1
y	y position	m
z	z position	m
$\hat{\boldsymbol{z}}$	Unit vector in z-direction	1
$\bar{z}$	Mean z-heigth	m
$\int I$	Integral factors for the magnetic field of a bar magnet	1

Greek

A	Coefficient A of the spherical expansion	1
B	Coefficient B of the spherical expansion	1
$\boldsymbol{B}$	Coefficient vector for the inclusion model	1
Γ	Index notation of $\boldsymbol{\Gamma}$	1
$\boldsymbol{\Gamma}$	$2N^{\mathsf{m}} \times 2N^{\mathsf{m}}$ matrix for the inclusion model	1
Δ	Index notation of $\boldsymbol{\Delta}$	1
$\boldsymbol{\Delta}$	$2N^{\mathsf{m}} \times 2N^{\mathsf{m}}$ matrix for the inclusion model	1
Δt	Time step	m
Φ	Magnetic scalar potential	A
$\boldsymbol{\nabla}$	Nabla operator	1
$\boldsymbol{\alpha}$	Angular acceleration	$\mathrm{rad\,s^{-2}}$
β	Contrast factor	1
$\dot{\gamma}$	Shear rate	$\mathrm{s^{-1}}$
δ	Kronecker delta	1
η	Dynamic viscosity	1
θ	Zenith angle	rad
κ_{B}	Boltzmann constant	1
λ	Emperical parameter for the modified velocity-Verlet algorithm	1
μ	Permeability	$\mathrm{N\,A^{-2}}$
ν	Poisson's ratio	1
ξ	Particle overlap	m
ρ	Mass density	$\mathrm{kg\,m^{-3}}$
σ	(Maxwell) stress	Pa
$\boldsymbol{\sigma}$	(Maxwell) stress tensor	Pa
τ	Shear stress	Pa
φ	Volume fraction	1
ϵ	Relative distance	1

ϕ	Azimuth angle	rad

Superscripts

c	Contact
dip	Dipole
eff	Effective
ext	External
f	Fluid
final	Final
fr	Relative fluid
int	Internal
k	Iteration k
m	Multipole(s)
mag	Magnetic
max	Maximum
mod	Modified
nc	Non-contact
nl	Normal lubrication
p	Particle(s)
peak	Peak
pm	Permenant magnet
pr	Relative particle
pred	Prediction
proc	Process
r	Relative
rem	Remanence
s	Sliding
sl	Sliding lubrication
top	Top wall
tot	Total
visc	Viscous
w	Wall(s)
$\parallel$	Parallel
$\perp$	Perpendicular

Subscripts

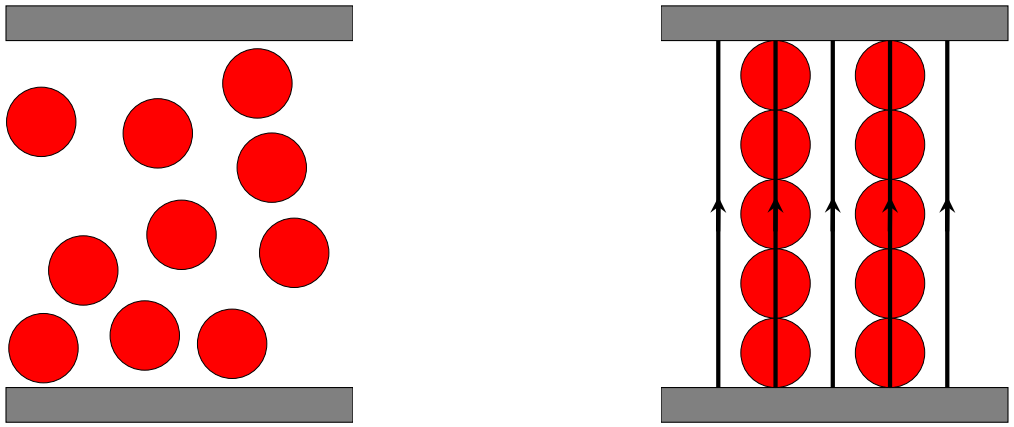
0	Free space
a	Particle a
ab	From particle b to/on a
ad	From wall d to/on a
b	Particle b
d	Wall d
i	Index notation i
j	Index notation j
k	Index notation k
l	Multipole degree of the spherical expansion of particle a
m	Order of the spherical expansion
s	Multipole degree of the spherical expansion of particle b
x	x-direction
y	y-direction
z	z-direction

1 Introduction

1.1. Background

Many mechanical systems in which a fluid is used, operate in a wide range of operating conditions. However, the fluids used are often only optimal for a small range of these conditions. For instance, take a car with passive dampers. Cars come across many types of road conditions, however passive dampers are designed for only one of these. On other road types, the car might not perform ideally. By making the properties of the fluid in the damper adaptable while driving, the car could perform optimally for many road types. A fluid that can change its rheological properties is thus desired.

Magnetorheological fluids (MRFs) are fluids that change their rheological properties in the presence of a magnetic field. Magnetorheological fluids are suspensions of magnetizable particles and, commonly non-magnetic, carrier fluids as shown in figure 1.1a [3], [4]. In the absence of a magnetic field, the suspension behaves as a non-Newtonian fluid [5]. Applying a uniform magnetic field results in the formation of chain-like structures within the MRF that form along the magnetic field as illustrated by figure 1.1b. These structures resist flow, thereby changing the yield stress and viscosity of the fluid. Only a few studies exist were a non-uniform magnetic field is applied to an MRF. The particles in the MRF still form structures that follow the magnetic field. These studies showed that the fluid behaves differently when a non-uniform field is applied [6], [7].



(a) MRF without an applied field

(b) MRF with an applied field

Figure 1.1: Overview of the MRF

Currently MRFs are mainly used for dampers, brakes, clutches [8], [9] and in the modern precision machining field for flexible fixtures or surface polishing [10]. Magnetorheological fluids could be of use in hydrodynamic bearings to increase the performance of the bearing. Hydrodynamic bearings consist of a rotating shaft, a supporting structure and a lubricant. When the shaft has no rotational velocity the shaft and the supporting structure are in contact. When the shaft rotates, the lubricant is squeezed and a fluid

film forms between the shaft and the supporting structure. At low speeds the fluid film has not formed fully and there is still contact between the shaft and the supporting structure, which results in friction and wear. To minimize the rotational velocity required to fully form a fluid film, surface textures can be applied to the bearing surfaces. These textures increase the fluid pressure, thereby improving the load capacity of the bearing. However, at low rotational speeds, the surfaces will still be in contact which results in wear of the textures [11], [12]. Replacing the lubricant by an MRF and applying a uniform magnetic field on the location of the textures would result in structures that resist flow, which increases the film pressure. Applying a non-uniform field, using a permanent magnet, results in the formation of physical textures, due to the particles separating from the fluid and accumulating on the permanent magnet. Due to the bearing being a closed system, any particles that are sheared of these textures eventually accumulate on the texture again, resulting in self-healing behavior [13]–[15].

1.2. Problem definition & research objective

The phenomenon that particles separate from the fluid and accumulate on the permanent magnet has recently been discovered. Therefore, not much is known about this behavior. A numerical model to predict the behavior of these self-healing textures does exist, however quite a few aspects of this phenomenon have not been investigated. One of which is the transition from the formation of structures to the formation of textures in a non-uniform magnetic field. A model that can simulate the behavior of an MRF in non-uniform magnetic fields would give insight into how the textures are formed and if the transition from structures to textures needs to be taken into account for the rheological behavior of the MRF. Therefore, the purpose of this thesis is to:

- Design a numerical model that gives insight into the formation of textures when a non-uniform field is applied to an MRF.
- This model should be able to determine the yield stress of the MRF so that a rheological model can be determined.
- Verify the model using literature and models that currently exist for MRFs and particulate flows.

In literature several models exist to determine the behavior of an MRF in a uniform magnetic field. Furthermore, relevant concepts from the particulate flows field are of interest for the model. These models and concepts are used as a basis to produce models that can determine the behavior of an MRF in a non-uniform magnetic field.

1.3. Structure of the thesis

This thesis starts with an introduction in which the problem and research objectives are defined. After the introduction, chapter 2 will go over the theory used to construct the models and will give a detailed description of each model. A more detailed overview of the existing literature can be found in: *The transition from structures to textures in magnetorheological fluids: A literature review*. Chapter 3 will then show the results of each model individually. The interpretations of these results will be discussed in: chapter 4. This thesis will finalize with conclusions in chapter 5 for each model and a overall conclusion. This chapter will also discuss recommendations for further research. In addition to this thesis, a paper will discuss the parallel model that was constructed. This paper is shown in: *The transition from structures to textures in magnetorheological fluids*.

2

Method

The MRF (magnetorheological fluid) is modeled in order to examine and predict its behavior. Two main methods exist in literature. The discrete method models the fluid and particles separately. This method is mainly used to examine the particle dynamics and structure formation within the MRF. The continuum approach provides a faster way to determine the behavior of an MRF, however only the macroscopic properties can be determined. For this research the MRF will be modeled using the discrete method with the parameters shown in table 2.2. A few different models are made to examine the behavior of the MRF. A model for very small domains was made in MATLAB during this project, however this model is of less interest and will therefore be discussed in appendix D. An overview of the properties of the models is given in table 2.1.

This chapter will start off with an overview of the discrete element method. Then, a single core Fortran (Formula translating system) model is discussed where two different models are used for the magnetic interaction forces between particles. Thirdly, a parallel Fortran model is described where OpenMPI (Open Message Passing Interface) is used to simulate larger domains [16]. Finally, the DNS (Direct Numerical Simulation) code of Breugem is modified to incorporate magnetic interaction forces between particles in order to determine whether two-way coupling is required.

2.1. Discrete element modeling

In a discrete model the continuous phase (carrier fluid), the discrete phase (particles), and the magnetic field are considered separately. Basic physical laws are used to model the three phases and their interactions. Numerical solving techniques are used to solve this kind of simulation. The main problem with the discrete approach is the high computational cost. However, because the models are based on basic physical laws, there are few assumptions and therefore the method can be more accurate than the continuum approach [17].

The behavior of suspensions can be modeled using Newton's equations of motion for the particles and the Navier-Stokes equations for the fluid. Next to this, boundary and initial conditions, the interaction

Model	Magnetic interaction method	Advantages	Limitation	Assumptions
Single core Fortran models	Fixed dipole	Low run times	Low number of particles, Small simulation domain	Particles are dipoles, Particles are only induced by the external magnetic field, Particles do not act on the fluid, No slip between particle/walls
	Mutual dipole	More accurate particle interaction	Low number of particles, Small simulation domain, High run times	Particles are dipoles, Particles do not act on the fluid, No slip between particle/walls
Parallel Fortran model	Fixed dipole	Low run times, Large number of particles, Large simulation domain	Requires a multi-core machine	Particles are dipoles, Particles are only induced by the external magnetic field, Particles do not act on the fluid, No slip between particle/walls
Breugem's DNS code with magnetic particles	Fixed dipole	2-way coupled, Slip between particles/walls	High run times, Requires a multi-core machine, Low number of particles, Small simulation domain	Particles are dipoles, Particles are only induced by the external magnetic field

Table 2.1: Overview of the models

Parameter	Value	Unit
μ_0	$4\pi \times 10^{-7}$	$\mathrm{N\,A^{-2}}$
λ	0.65	
φ	20%	
t^{final}	7.5	ms
L_z	75	µm

(a) General parameters

Parameter	Value	Unit
μ^{f}	μ_0	$\mathrm{N\,A^{-2}}$
ρ^{f}	1000	$\mathrm{kg\,m^{-3}}$
η^{f}	0.2	Pa s

(b) Fluid parameters

Parameter	Value	Unit
μ^{p}	$2000\mu_0$	$\mathrm{N\,A^{-2}}$
R	2.5	µm
ρ^{p}	7800	$\mathrm{kg\,m^{-3}}$
E	500	MPa
ν	0.3	

(c) Particle parameters

Table 2.2: Parameters

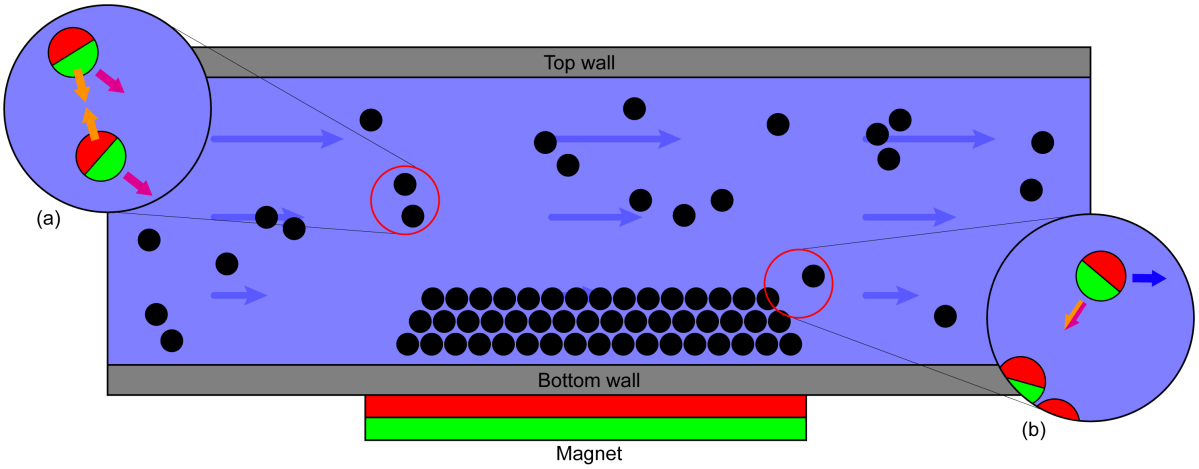


Figure 2.1: Overview of the discrete element model with the different forces acting on the particles
Blue = Stokes' drag force
Orange = Magnetic interaction force
Pink = Magnetic gradient force

between the continuous and discrete phase, and the interaction between particles should be taken into account. An overview of the discrete element model is shown in figure 2.1.

The equation of motion of a particle a with a mass M_a and an acceleration $\boldsymbol{a}_a$ is shown in equation (2.1).

$$M_a \frac{\mathrm{d}^2 \boldsymbol{r}_a}{\mathrm{d}t^2} = M_a \boldsymbol{a}_a = \boldsymbol{F}_a^{\text{c}} + \boldsymbol{F}_a^{\text{nc}} + \boldsymbol{F}_a^{\text{f}} + \boldsymbol{F}_a^{\text{ext}} \tag{2.1}$$

This section will start by discussing the contact force acting on a particle, $\boldsymbol{F}_a^{\text{c}}$. Then the non-contact forces, $\boldsymbol{F}_a^{\text{nc}}$, are discussed, after which the interaction between the fluid and the particles is described, $\boldsymbol{F}_a^{\text{f}}$. The external forces, $\boldsymbol{F}_a^{\text{ext}}$, are the final group of forces that will be discussed. The section will continue by discussing the indicators that will be monitored by the models. Then the integration scheme used in the models will be described and the boundary condition for the models will be presented. The method to take two-way coupling into account is then discussed. This section will end by describing a method to determine particle accumulations in the simulation domain.

2.1.1. Contact force

The contact force, $\boldsymbol{F}_a^{\text{c}}$, acting on a particle consists of the contact force acting from particle b on particle a and of contact forces acting on particle a due to a wall collision. The contact forces are modeled using a simplified Hertzian model [6], [17]–[26]. This model only takes the normal component of the collision force into account, slip is thus neglected. The contact force acting from particle b on particle a is determined using equation (2.2).

$$\boldsymbol{F}_{ab}^{\text{c}} = \left(\frac{\frac{2}{3}E}{1-\nu^2} \sqrt{R^{\text{eff}}} \xi_{ab}^{3/2} \right) \hat{\boldsymbol{r}}_{ab} \tag{2.2}$$

where $R^{\text{eff}} = R_a R_b / (R_a + R_b)$, $\xi_{ab} = \max\{R_a + R_b - \|\boldsymbol{r}_{ab}\|, 0\}$ and $\boldsymbol{r}_{ab} = (\boldsymbol{r}_a - \boldsymbol{r}_b)$.

It is important to notice that this force greatly increases when the distance between the center of the particles decreases. In order to make sure that the simulation is still stable, a stable time step Δt needs to be determined. The time step can be computed by sampling the duration of a contact t^{c}, given in equation (2.3). In literature several sampling factors are used in order to create a stable simulation. Shäfer, Dippel, and Wolf samples the collision over 100 time steps, Lagger, Bierwisch, Korvink, ***et al.*** samples over 50 time steps and Xuan, Li, Li, ***et al.*** samples over approximately 5 time steps [6], [27], [28].

$$t_{ab}^{\mathsf{c}} = 3.21 \left(\frac{\left(1-\nu^2\right) M_a M_b}{\frac{2}{3} E \sqrt{R^{\mathsf{eff}}} \left(M_a + M_b\right)} \right)^{2/5} \|\boldsymbol{v}_{ab}\|^{-1/5} \tag{2.3}$$

Walls might be placed on the boundaries of the simulation region. Therefore, when a particle collides with the wall they experience a contact force. The contact force acting from a wall d on particle a is determined by equation (2.4).

$$\boldsymbol{F}_{ad}^{\mathsf{c}} = \left(\frac{\frac{2}{3}E}{1-\nu^2} \sqrt{R} \xi_{ad}^{3/2} \right) \hat{\boldsymbol{r}}_{ad} \tag{2.4}$$

where $\xi_{ad} = \max\left\{R_a - \|\boldsymbol{r}_{ad}\|, 0\right\}$.

The total contact force acting on particle a is thus given by equation (2.5). It should be noted that the Young's modulus can be reduced so that the time step can be increased, while still accurately approximating the behavior of the MRF [21].

$$\boldsymbol{F}_a^{\mathsf{c}} = \sum_{b=1, b\neq a}^{N^{\mathsf{p}}} \boldsymbol{F}_{ab}^{\mathsf{c}} + \sum_{d=1}^{N^{\mathsf{w}}} \boldsymbol{F}_{ad}^{\mathsf{c}} \tag{2.5}$$

2.1.2. Non-contact forces

The non-contact force acting on a particle can be divided into two forces, the magnetic interaction force and the lubrication force. The lubrication force is not used in the Fortran models, as it is expected that it does not influence the steady state behavior of the MRF. The lubrication force should only affect the time when steady state is reached. The magnetic interaction force will be discussed below in more detail and further details of the lubrication force are given in appendix A.

The section will start by giving an overview of the magnetic interaction force. Then, two simplified methods to determine the force are described. Finally, in order to decrease the run times of the models, a cut-off distance is introduced.

Magnetic interaction force

The magnetic interaction force acting between two particles, illustrated by the orange arrows in figure 2.1 (a) and (b), can be solved by using the simplified Maxwell's equations. It is assumed that the fluid has the permeability μ_0 and that there are no free currents in the MRF. In this model particles are seen as multipoles, which means that each particle has multiple positive and negative poles. These assumptions result in the Maxwell's equations (2.6).

$$\boldsymbol{\nabla} \cdot \boldsymbol{B} = 0 \tag{2.6a}$$

$$\boldsymbol{\nabla} \times \boldsymbol{H} = 0 \tag{2.6b}$$

where $\boldsymbol{B}^{\mathsf{f}} = \mu^{\mathsf{f}} \boldsymbol{H}^{\mathsf{f}}$ in the fluid and $\boldsymbol{B}_a^{\mathsf{p}} = \mu^{\mathsf{p}} \boldsymbol{H}_a^{\mathsf{p}}$ in each particle. Equation (2.6b) can be used to form the scalar potentials $\boldsymbol{H}^{\mathsf{f}} = -\boldsymbol{\nabla}\Phi^{\mathsf{f}}$ inside the fluid and $\boldsymbol{H}_a^{\mathsf{p}} = -\boldsymbol{\nabla}\Phi_a^{\mathsf{p}}$ inside each particle a. Using these relations and the Maxwell's equations (2.6), the Laplace's equations (2.7) can be set up [23], [29].

$$\boldsymbol{\nabla}^2 \Phi^{\mathsf{f}} = 0 \tag{2.7a}$$

$$\boldsymbol{\nabla}^2 \Phi_a^{\mathsf{p}} = 0 \tag{2.7b}$$

Defining a coordinate system at the center of a particle a, where $\boldsymbol{r}_a^{\mathsf{p}}$ is the coordinate vector, requires the conditions $\hat{\boldsymbol{r}}_a^{\mathsf{p}} \cdot \left(\boldsymbol{H}^{\mathsf{f}} - \boldsymbol{H}^{\mathsf{p}}\right) = 0$ and $\hat{\boldsymbol{r}}_a^{\mathsf{p}} \times \left(\boldsymbol{B}^{\mathsf{f}} - \boldsymbol{B}^{\mathsf{p}}\right) = 0$ to satisfy the potential of equations (2.8a) and (2.8b) at $\|\boldsymbol{r}_a^{\mathsf{p}}\| = R$. At $\|\boldsymbol{r}_a^{\mathsf{p}}\| \to \infty$ equation (2.8c) should be satisfied.

$$\Phi_a^{\mathsf{p}} = \Phi^{\mathsf{f}} \tag{2.8a}$$

$$\mu^{\mathsf{p}} \frac{\partial \Phi_a^{\mathsf{p}}}{\partial \left\|\boldsymbol{r}_a^{\mathsf{p}}\right\|} = \mu^{\mathsf{f}} \frac{\partial \Phi^{\mathsf{f}}}{\partial \left\|\boldsymbol{r}_a^{\mathsf{p}}\right\|} \tag{2.8b}$$

$$\Phi^{\mathsf{f}} \to \boldsymbol{H}^{\mathsf{ext}} \cdot \boldsymbol{r}_a^{\mathsf{p}} \tag{2.8c}$$

Equations (2.8) can be solved for the potential, from which the magnetic field can be determined. The magnetic field is then used to construct the Maxwell stress tensor of equation (2.9). The force acting on particle a is finally determined by a surface integral of the Maxwell stress tensor given by equation (2.10) [29].

$$\sigma_{ij} = \mu_0 H_i H_j - \frac{\mu_0}{2} \delta_{ij} H_k H_k \tag{2.9}$$

$$\boldsymbol{F} = R^2 \int_0^{2\pi} \int_0^{\pi} \boldsymbol{\sigma} \cdot \hat{\boldsymbol{r}}_a^{\mathsf{p}} \sin\theta_a^{\mathsf{p}} \, \mathrm{d}\theta_a^{\mathsf{p}} \, \mathrm{d}\phi_a^{\mathsf{p}} \tag{2.10}$$

Equation (2.7a) can be written down as the sum of potential due to the external magnetic field and the magnetic field generated by all the particles. The magnetic field produced by the particles is determined using N^{p} spherical harmonic expansions located at the center of each particle. An example of a simplified model with two particles is given in appendix B. At the boundary of each particle, the boundary conditions of equations (2.8) need to be taken into account, however this results in complications of the spherical harmonic expansion. In order to solve the general problem, the expansion needs to be truncated while retaining N^{m} multipoles and combining the individual coordinate systems of all particles into one. This results in an impractical of N^{p}, $N^{\mathsf{p}} N^{\mathsf{m}} \times N^{\mathsf{p}} N^{\mathsf{m}}$ systems of linear equations from which the unknown coefficients of each expansion need to be determined. In order to reduce the complexion of the general solution several approximations can be made. These approximations result in the models discussed below.

Fixed dipole method The fixed dipole method is a simple and fast way to determine the force acting between two magnetically induced particles. The particles are considered to be point dipoles and multiple particle interaction is ignored. The particles are thus only induced by the external magnetic field that is applied to the MRF [17], [21]–[23], [28]–[36]. The magnetic dipole moment can be determined using equation (2.11).

$$\boldsymbol{m}_a = 3 V^{\mathsf{p}} \beta \boldsymbol{H}^{\mathsf{ext}}\left(\boldsymbol{r}_a\right) \tag{2.11}$$

where $\beta = \frac{\mu^{\mathsf{pr}} - \mu^{\mathsf{fr}}}{\mu^{\mathsf{pr}} + 2\mu^{\mathsf{fr}}}$, $\mu^{\mathsf{pr}} = \frac{\mu^{\mathsf{p}}}{\mu_0}$ and $\mu^{\mathsf{fr}} = \frac{\mu^{\mathsf{f}}}{\mu_0}$.

Due to the magnetic dipole moment, the particle generates a magnetic field. Seen from the particles coordinate system, this field can be determined using equation (2.12).

$$\boldsymbol{H}_a^{\mathsf{dip}}\left(\boldsymbol{r}_a^{\mathsf{p}}\right) = \frac{1}{4\pi} \left(\frac{3\left(\boldsymbol{m}_a \cdot \boldsymbol{r}_a^{\mathsf{p}}\right)}{\left\|\boldsymbol{r}_a^{\mathsf{p}}\right\|^5} - \frac{\boldsymbol{m}_a}{\left\|\boldsymbol{r}_a^{\mathsf{p}}\right\|^3} \right) \tag{2.12}$$

The interaction force acting on a point dipole due to the magnetic field produced by another dipole is given by equation (2.13).

$$\begin{aligned} \boldsymbol{F}_{ab}^{\mathsf{mag}} &= \mu_0 \boldsymbol{\nabla}_{\boldsymbol{r}_a} \left(\boldsymbol{m}_a \cdot \boldsymbol{H}_b^{\mathsf{dip}}\left(\boldsymbol{r}_{ab}\right) \right) \\ &= \frac{3\mu_0}{4\pi \|\boldsymbol{r}_{ab}\|^4} \Big(\left(\boldsymbol{m}_a \cdot \boldsymbol{m}_b\right) \hat{\boldsymbol{r}}_{ab} + \left(\boldsymbol{m}_b \cdot \hat{\boldsymbol{r}}_{ab}\right) \boldsymbol{m}_a + \\ &\qquad \left(\boldsymbol{m}_a \cdot \hat{\boldsymbol{r}}_{ab}\right) \boldsymbol{m}_b - 5 \left(\boldsymbol{m}_a \cdot \hat{\boldsymbol{r}}_{ab}\right) \left(\boldsymbol{m}_b \cdot \hat{\boldsymbol{r}}_{ab}\right) \hat{\boldsymbol{r}}_{ab} \Big) \end{aligned} \tag{2.13}$$

Mutual dipole method The mutual dipole method uses the same principles as the fixed dipole method discussed above. The particles are still considered to be point dipoles, therefore multi-poles are ignored. However, this model does take the multiple particle interaction into account, which means that the particles are induced by the external magnetic field and the magnetic field produced by other particles [6], [17], [18], [22], [23], [29], [37]. For this model, the magnetic moment of a particle is thus determined using equation (2.14).

$$\boldsymbol{m}_a = 3V^{\mathsf{p}}\beta\left(\boldsymbol{H}^{\mathsf{ext}}\left(\boldsymbol{r}_a\right) + \sum_{b=1,b\neq a}^{N^{\mathsf{p}}} \boldsymbol{H}^{\mathsf{dip}}\left(\boldsymbol{r}_{ab}\right)\right) \tag{2.14}$$

The magnetic moments of the particles depend on each other, therefore an iterative algorithm is used to determine the magnetic moment of each particle. This algorithm is given by equation (2.15). The first step, $\boldsymbol{m}_a^{(0)}$, is determined by the magnetic dipole moment of equation (2.11).

$$\begin{aligned}\boldsymbol{m}_a^{(k+1)} &= 3V^{\mathsf{p}}\beta\left(\boldsymbol{H}^{\mathsf{ext}}\left(\boldsymbol{r}_a\right) + \sum_{b=1}^{a-1}\boldsymbol{H}^{\mathsf{dip}}\left(\boldsymbol{r}_{ab},\boldsymbol{m}_b^{(k+1)}\right) + \sum_{b=a+1}^{N^{\mathsf{p}}}\boldsymbol{H}^{\mathsf{dip}}\left(\boldsymbol{r}_{ab},\boldsymbol{m}_b^{(k)}\right)\right)\\ k &= 0,1,2,\ldots\end{aligned} \tag{2.15}$$

The magnetic interaction force acting on particle a is then determined by using the final iteration of the magnetic moment in equation (2.13). No more than three iterations are required to have a relative error of less than 10^{-5} [6], [23].

Cut-off distance
In order to improve the computation speed of these models, the forces are not calculated over the whole simulation domain. Only particles that are within a certain distance of each other are taken into account. This is illustrated by figure 2.2 in 2D. In literature several cut-off distances are used for the magnetic interaction force, an overview of these cut-off distances is given in table 2.3. For this thesis the magnetic interaction force will only be taken into account when $\|\boldsymbol{r}_{ab}\| \leq 6R$. The magnetic force acting from particle b on a is thus given by equation (2.16)

Cut-off distance / R	Model	References
4	Fixed	[22], [23], [30]
6	Mutual	[6], [23], [34], [38]
6.4	Fixed	[21]

Table 2.3: Cut-off distances used in literature

$$\boldsymbol{F}_{ab}^{\mathsf{mag}} = \begin{cases}\mu_0\boldsymbol{\nabla}_{\boldsymbol{r}_a}\left(\boldsymbol{m}_a\cdot\boldsymbol{H}^{\mathsf{dip}}\left(\boldsymbol{r}_{ab}\right)\right), & \text{if} \quad \|\boldsymbol{r}_{ab}\| \leq 6R\\ 0, & \text{if} \quad \|\boldsymbol{r}_{ab}\| > 6R\end{cases} \tag{2.16}$$

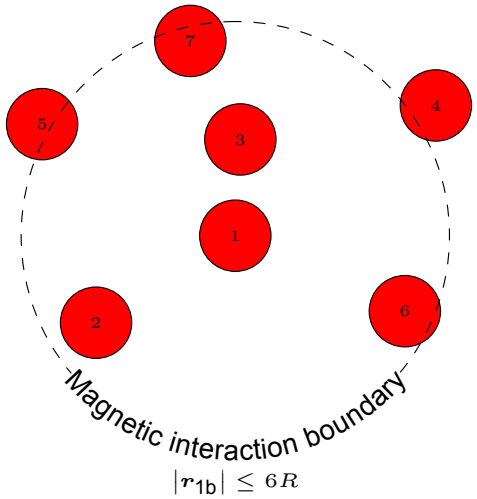


Figure 2.2: Cut-off distances as seen from particle 1 in 2D.
Particles $2, 3, 6, 7$ are taken into account for the magnetic interaction force. Particle $4, 5$ are not considered.

The total non-contact force acting on particle a is given by the summation of the magnetic interaction forces as shown in equation (2.17).

$$\boldsymbol{F}_a^{\mathsf{nc}} = \sum_{b=1, b\neq a}^{N^{\mathsf{p}}} \boldsymbol{F}_{ab}^{\mathsf{mag}} \tag{2.17}$$

2.1.3. Particle fluid interaction

Magnetorheological fluids are mainly used in configurations where the inertial forces are small compared with the viscous forces. The Reynolds number is thus generally very low, $\mathrm{Re} \ll 1$. The force acting on the particles by the fluid can therefore be approximated using Stokes' drag force of equation (2.18), illustrated by the blue arrows in figure 2.1 (b). It should be noted that this model does not consider that the particles influences the fluid velocity [6], [17]–[21], [23], [28], [32]–[34], [38]–[41].

$$\boldsymbol{F}_a^{\mathsf{f}} = 6\pi\eta^{\mathsf{f}} R \left(\boldsymbol{v}^{\mathsf{f}}(\boldsymbol{r}_a) - \boldsymbol{v}_a \right) \tag{2.18}$$

The MRF is confined between two plates in the z direction. The upper plate has a velocity of $\boldsymbol{v}^{\mathsf{top}} = v^{\mathsf{top}} \boldsymbol{e}_y$ and the bottom plate is stationary. A Couette flow, with a shear rate of $\dot{\gamma} = v^{\mathsf{top}}/L_z$, is thus generated between the two plates. An illustration of the flow profile is given in figure 2.3. The fluid velocity at a particles centroid is determined using equation (2.19).

$$\boldsymbol{v}^{\mathsf{f}}(\boldsymbol{r}_a) = \dot{\gamma} (\boldsymbol{r}_a)_z \, \boldsymbol{e}_y \tag{2.19}$$

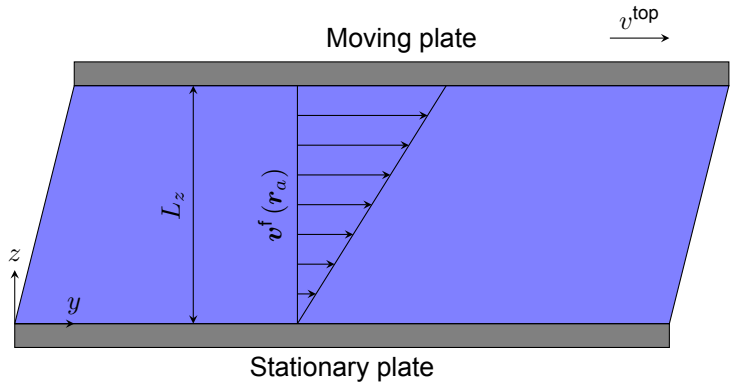


Figure 2.3: Couette flow between two plates

2.1.4. External forces

Next to the forces described above, there are some external force acting on the particles. Not all these forces are taken into account, because that would increase simulation times, but would not improve the accuracy of the models. Because the particles in the MRF are small, the Brownian forces will be discussed first. Then, the assumptions made for the anti-sedimentation modifications and gravity forces will be described. Finally, the model to determine te force acting from the external magnetic field on the particles will be presented.

Brownian motion

Brownian forces act on small particles in a fluid due to thermal fluctuations. The Peclet number, (Pe), given in equation (2.20), gives the ratio of the shear forces to the Brownian forces. The parameters used for the models given in this report, result in large values of Pe. The shear forces will thus dominate the Brownian forces, which means can be neglected [17], [20], [42]–[44].

$$\mathrm{Pe} = \frac{6\pi\eta^{\mathrm{f}}R^3\dot{\gamma}}{\kappa_{\mathrm{B}}T} \tag{2.20}$$

Sedimentation and gravity

The two phases of the MRF have a mismatch in density. This results in the sedimentation of particles which is unwanted in long term use of the fluid [45]. Several methods are used to overcome the settling behavior of MRFs. Additives can be added to the MRF to increase the stability of the fluid. These additives occupy the empty spaces between the magnetic particles. Another way to increase the stability of the fluid is by reducing the density mismatch between the two phases. This is done by coating the particles in a low density material or by increasing the density of the carrier fluid [4], [46], [47]. In order to incorporate these stabilization techniques into the model, it is assumed that the particles within the MRF are neutrally buoyant. Therefore, gravity does not affect the particles. It should be noted that these modifications influence the magnetorheological effect of the MRF which are not taken into account.

External magnetic force

The force acting on a magnetic dipole due to an external magnetic field, illustrated by the pink arrows in figure 2.1 (a) and (b), is given by equation (2.21). In a uniform magnetic field, this force is equal to zero because there is no gradient in the magnetic field [6].

$$\boldsymbol{F}_a^{\mathrm{mag,ext}} = \mu_0 \boldsymbol{\nabla}\left(\boldsymbol{m}_a \cdot \boldsymbol{H}^{\mathrm{ext}}\left(\boldsymbol{r}_a\right)\right) \tag{2.21}$$

In this report, permanent bar magnets are used. The model used to determine the external magnetic field at particle a, $\boldsymbol{H}^{\mathrm{ext}},(\boldsymbol{r}_a)$ assumes that the permanent magnet is a continuous distribution of dipoles with a volume $\mathrm{d}V$ and a magnetic dipole moment of $\mathrm{d}\boldsymbol{m} = \boldsymbol{m}^{\mathrm{dip}} = \frac{1}{\mu_0}\boldsymbol{B}^{\mathrm{rem}}\,\mathrm{d}V$. The potential and the magnetic field of each dipole with the origin located at its center are respectively given by equations (2.22).

$$\Phi^{\mathrm{dip}}\left(\boldsymbol{r}\right) = \frac{1}{4\pi}\frac{\boldsymbol{m}^{\mathrm{dip}}\cdot\hat{\boldsymbol{r}}}{\left\|\boldsymbol{r}\right\|^2} \tag{2.22a}$$

$$\boldsymbol{B}^{\mathrm{dip}}\left(\boldsymbol{r}\right) = \frac{\mu_0}{4\pi}\frac{3\hat{\boldsymbol{r}}\left(\hat{\boldsymbol{r}}\cdot\boldsymbol{m}^{\mathrm{dip}}\right)-\boldsymbol{m}^{\mathrm{dip}}}{\left\|\boldsymbol{r}\right\|^3} \tag{2.22b}$$

In order to determine the magnetic field produced by the permanent magnet, the magnetic field or the magnetic scalar potential contributions of all dipoles are integrated over the volume of the permanent magnet. The magnetic field of a homogeneously polarized bar magnet with dimensions $2L_x^{\mathrm{pm}} \times 2L_y^{\mathrm{pm}} \times 2L_z^{\mathrm{pm}}$ is thus given by equations (2.23).

$$\mathrm{d}\boldsymbol{B}^{\text{ext}}(\boldsymbol{r}_0) = \frac{\mu_0}{4\pi}\frac{3\hat{\boldsymbol{r}}\left(\hat{\boldsymbol{r}}\cdot\mathrm{d}\boldsymbol{m}\right)-\mathrm{d}\boldsymbol{m}}{\|\boldsymbol{r}\|^3} = \frac{1}{4\pi}\frac{3\hat{\boldsymbol{r}}\left(\hat{\boldsymbol{r}}\cdot\boldsymbol{B}^{\text{rem}}\right)-\boldsymbol{B}^{\text{rem}}}{\|\boldsymbol{r}\|^3}\,\mathrm{d}V \tag{2.23a}$$

$$\boldsymbol{B}^{\text{ext}}(\boldsymbol{r}_0) = \int_{-L_x^{\text{pm}}}^{L_x^{\text{pm}}}\int_{-L_y^{\text{pm}}}^{L_y^{\text{pm}}}\int_{-L_z^{\text{pm}}}^{L_z^{\text{pm}}}\mathrm{d}\boldsymbol{B}^{\text{ext}}(\boldsymbol{r}_0)\,\mathrm{d}x^{\text{dip}}\,\mathrm{d}y^{\text{dip}}\,\mathrm{d}z^{\text{dip}} \tag{2.23b}$$

$$\boldsymbol{H}^{\text{ext}}(\boldsymbol{r}_0) = \frac{\boldsymbol{B}^{\text{ext}}(\boldsymbol{r}_0)}{\mu_0} - \boldsymbol{M}^{\text{pm}}(\boldsymbol{r}_0) \tag{2.23c}$$

where $\boldsymbol{r} = \boldsymbol{r}_0 - \boldsymbol{r}^{\text{dip}}$, $\boldsymbol{r}_0$ is a position in free space, $\boldsymbol{r}^{\text{dip}}$ is the location of the dipole in the magnet, $\boldsymbol{M}^{\text{pm}}(\boldsymbol{r}_0) = \frac{\boldsymbol{B}^{\text{rem}}}{\mu_0}$, inside the magnet, $\boldsymbol{M}^{\text{pm}}(\boldsymbol{r}_0) = 0$ outside the magnet and the origin is located at the center of the magnet [48]–[50]. An overview of the model is shown in figure 2.4. The elaborated model, source code, verification with a COMSOL model and the magnetic fields of the magnets used in this report are given in appendix C. With the external magnetic field known, the force acting on a particle due to the field can be determined using equation (2.21).

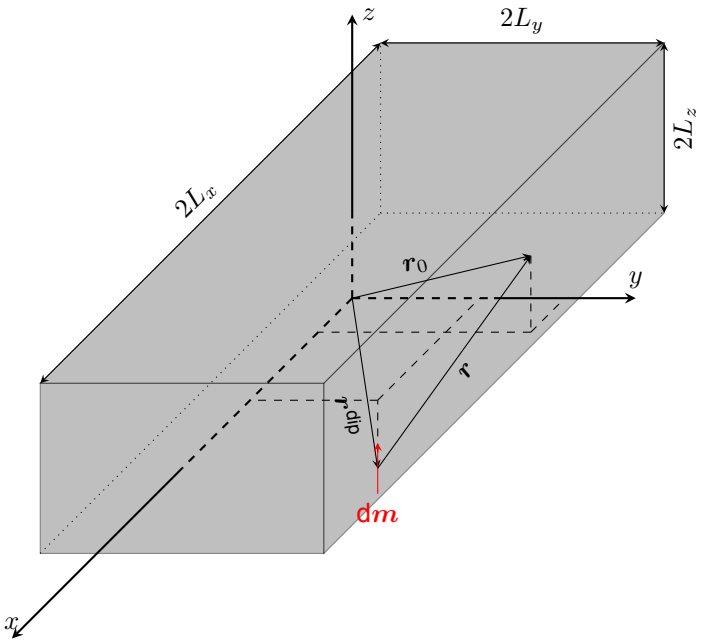


Figure 2.4: Schematic overview of the bar magnet model

2.1.5. Dynamic evolution

The dynamic evolution of the MRF is monitored by several indicators. The kinetic energy of the MRF is stored to check the stability of the simulation and to find out when steady state is reached. The shear stress within the MRF is saved in order to see the shear stress versus time response of the simulation and determine the dynamic yield stress of the fluid. Finally, the mean z height is stored to see if particles are moving towards the magnet.

The section will start by discussing the importance of the kinetic energy. Then, the model to determine the shear stress in the fluid will be given. Finally, the reason for saving the mean z is explained.

Kinetic energy

The kinetic energy of the fluid gives an indication of the phase of the MRF. In the first phase, the particles are able to move freely, which results in high kinetic energies. After that, particles will start to form chain-like structures, thereby decreasing the kinetic energy in the fluid. When the formation of these structures is complete, the MRF will reach a steady state where the aggregations are stable. At this phase the kinetic energy will reach a plateau [6], [23], [30]. An overview of the phases is given in figure 2.5. The time required to go from the initial state to 95% of the final state is called the response

time, which is located just before the end of region 2. This response time is used in industry to indicate the behavior of MRFs. The kinetic energy of the fluid is used to monitor the stability of the simulation as well. If the kinetic energy reaches very large values the simulation is unstable and the time step Δt needs to be decreased. The kinetic energy of the MRF is determined using equation (2.24a).
Next to this, the maximum kinetic energy of the particles is monitored. This indicator follows the same curve as the kinetic energy of the fluid, but at lower energy values. The maximum kinetic energy is determined using equation (2.24b) and allows for the calculation of a stable time step. The maximum kinetic energy of the peak at the end of the first phase can be used to determine the maximum relative velocity between particles, which, as discussed in section 2.1.1, influences the collision time between particles and thus the time step of the simulation [6]. The maximum relative speed between particles is determined using equation (2.24c).

$$K^{\text{tot}}(t) = \sum_{a=1}^{N^{\text{p}}} \frac{1}{2} M_a \|\boldsymbol{v}_a\|^2 \tag{2.24a}$$

$$K^{\text{max}}(t) = \max_{a \in N^{\text{p}}} \left(\frac{1}{2} M_a \|\boldsymbol{v}_a\|^2 \right) \tag{2.24b}$$

$$\|\boldsymbol{v}_{ab}\|^{\text{max}} = 2\|\boldsymbol{v}_a\|^{\text{max}} = 2\sqrt{\frac{2K^{\text{max}}\left(t = t^{\text{peak}}\right)}{M_a}} \tag{2.24c}$$

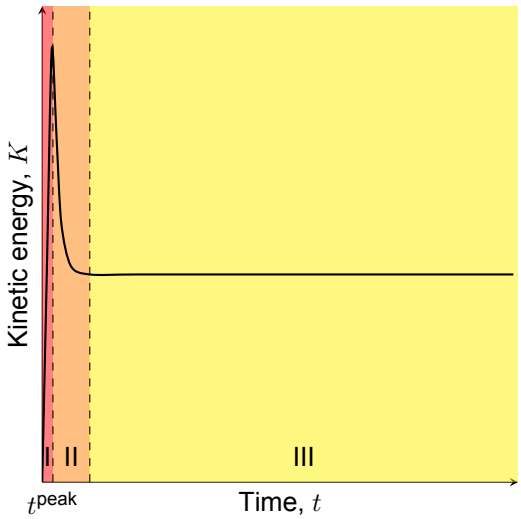


Figure 2.5: Dynamic evolution of the kinetic energy of the fluid MRF

Shear stress
The average stress of the particle structures in the MRF is commonly given by the internal stress tensor, $\boldsymbol{\sigma}^{\text{int}}$, or the external stress tensor, $\boldsymbol{\sigma}^{\text{ext}}$, from equations (2.25) [21]. When the MRF is in equilibrium, the internal stress should equal the external stress [28].

$$\boldsymbol{\sigma}^{\text{int}} = -\frac{1}{V} \sum_{a}^{N^{\text{p}}} \sum_{b=a+1}^{N^{\text{p}}} \boldsymbol{r}_{ab} \otimes \left(\boldsymbol{F}^{\text{c}}_{ab} + \boldsymbol{F}^{\text{mag}}_{ab} \right) \tag{2.25a}$$

$$\boldsymbol{\sigma}^{\text{ext}} = \frac{1}{V} \sum_{a}^{N^{\text{p}}} \boldsymbol{r}_a \otimes \left(\boldsymbol{F}^{\text{f}}_{a} + \boldsymbol{F}^{\text{ext}}_{a} \right) \tag{2.25b}$$

In this research, the shear stress of the fluid is of interest due to the applied shear rate. The shear stress of the MRF is given by the (z, y) component of the stress tensor. The internal and external shear stresses are determined using equations (2.26) [6], [35], [36], [51], [52]. Next to the contribution of the

shear stress of the particle structures, there is a viscous addition as well. The viscous shear stress in the fluid is determined using the Krieger-Daugherty model of equation (2.27) where $\varphi^{\text{max}} = 0.63$ [21], [53]. Finally, the total shear stress is determined by the summation of the external or internal shear stress and the viscous shear stress.

$$\tau^{\text{int}} = \boldsymbol{\sigma}^{\text{int}}_{(z,y)} = -\frac{1}{V}\sum_{a}^{N^{\text{p}}}\sum_{b=a+1}^{N^{\text{p}}} (\boldsymbol{r}_{ab})_z \left(\boldsymbol{F}^{\text{c}}_{ab} + \boldsymbol{F}^{\text{mag}}_{ab}\right)_y \tag{2.26a}$$

$$\tau^{\text{ext}} = \boldsymbol{\sigma}^{\text{ext}}_{(z,y)} = \frac{1}{V}\sum_{a}^{N^{\text{p}}} (\boldsymbol{r}_a)_z \left(\boldsymbol{F}^{\text{f}}_{a}\right)_y \tag{2.26b}$$

$$\tau^{\text{visc}} = \eta^{\text{f}}\dot{\gamma}\left(1 - \frac{\varphi}{\varphi^{\text{max}}}\right)^{-2.5\varphi^{\text{max}}} \tag{2.27}$$

The shear stress versus time response for an MRF in a uniform magnetic field can be distinguished into three regions, as shown in figure 2.6. Region I is the elastic region which shows the low-strain regime where the stress is proportional to the shear strain. For shear rates for which the viscous contribution can be neglected, the static yield stress is located at the peak. In region II the shear stress starts to decline to a steady state in region III. The steady state stress is known as the dynamic yield stress [35], [36].

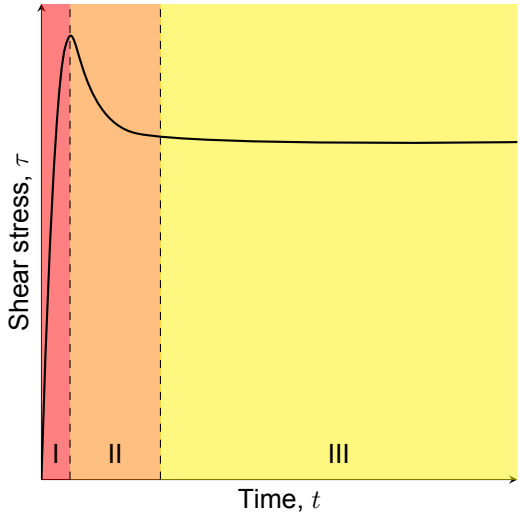


Figure 2.6: Shear stress regions

Mean z-height

For a uniform magnetic field it is expected that the distribution of particles remains even in the z-direction. In a non-uniform field, the gradient force of equation (2.21), pulls the particles towards the bar magnet, thereby disturbing the uniform distribution in z-direction. A decline in the mean z-height is thus expected over time until all the particle have settled to a steady state. The mean z-height is determined using equation (2.28).

$$\bar{z}\,(t) = \frac{\sum_{a=1}^{N^{\text{p}}} (\boldsymbol{r}_a)_z}{N^{\text{p}}} \tag{2.28}$$

2.1.6. Integration scheme

The equation of motion from equation (2.1) is solved using the modified velocity-Verlet algorithm. This is a simple, convenient, numerically stable method that is reversible in time. Furthermore, if the forces are conservative the method guarantees conservation of linear momentum [54]. The original velocity-Verlet algorithm assumes that the acceleration of a particle only depends on its position. However, due to the

Stokes drag force of equation (2.18) the acceleration of the particle depends on its velocity as well. The algorithm is therefore modified to make a prediction for the new velocity. This prediction is corrected in the final step of the algorithm. The algorithm is shown in equations (2.29) where the empirical parameter λ is chosen as 0.65 [6], [55], [56].

$$\boldsymbol{r}_a\left(t+\Delta t\right) = \boldsymbol{r}_a\left(t\right) + \Delta t \boldsymbol{v}_a\left(t\right) + \frac{1}{2}\left(\Delta t\right)^2 \boldsymbol{a}_a\left(t\right) \tag{2.29a}$$

$$\boldsymbol{v}_a^{\text{pred}}\left(t+\Delta t\right) = \boldsymbol{v}_a\left(t\right) + \lambda \Delta t \boldsymbol{a}_a\left(t\right) \tag{2.29b}$$

$$\boldsymbol{a}_a\left(t+\Delta t\right) = \boldsymbol{a}_a\Big(\boldsymbol{r}_a\left(t+\Delta t\right), \boldsymbol{v}_a^{\text{pred}}\left(t+\Delta t\right)\Big) \tag{2.29c}$$

$$\boldsymbol{v}_a\left(t+\Delta t\right) = \boldsymbol{v}_a\left(t\right) + \frac{1}{2}\Delta t\Big(\boldsymbol{a}_a\left(t\right) + \boldsymbol{a}_a\left(t+\Delta t\right)\Big) \tag{2.29d}$$

2.1.7. Boundary conditions

The MRF is confined in a simulation box of dimensions $L_x \times L_y \times L_z$. Conditions are placed on the boundaries of this box. On the bottom and top of the box at $z = 0$ and $z = L_z$ walls are placed. When particles come into contact with the wall, they experience the contact force given in equation (2.4). Therefore, the particles are confined in the z-direction. As discussed in section 2.1.3 the top wall has a velocity, $\boldsymbol{v}^{\text{top}}$, and the bottom wall is stationary which results in a Couette flow.
In the x-direction and y-direction PBC's (Periodic Boundary Condition) are placed. This is equivalent to each particle having an image of itself in neighboring image cells, as shown in figure 2.7, where the blue square is the simulated domain. By using PBCs a small part of a large domain can be simulated to approximate the large domain [30], [34]. In order to make sure that a particle only interacts with one image of any other particle, the minimum image convention is used. The position vector between two particles is therefore determined using equations (2.30) [6]. The minimum image convention requires the distance between the boundaries to be at least twice the cut-off distance, therefore $L_x \geq 2 \times 6R$ and $L_y \geq 2 \times 6R$. If this is not the case, it is possible that particles will interact multiple times with each other. Figure 2.7 shows that increasing the cut-off distance, or decreasing L_x and/or L_y, will lead to interactions with multiple images of particles.

$$\left(\boldsymbol{r}_{ab}\right)_x = \begin{cases} \left(\boldsymbol{r}_{ab}\right)_x + L_x, & \text{if} \quad \left(\boldsymbol{r}_{ab}\right)_x < -\frac{L_x}{2} \\ \left(\boldsymbol{r}_{ab}\right)_x - L_x, & \text{if} \quad \left(\boldsymbol{r}_{ab}\right)_x > \frac{L_x}{2} \end{cases} \tag{2.30a}$$

$$\left(\boldsymbol{r}_{ab}\right)_y = \begin{cases} \left(\boldsymbol{r}_{ab}\right)_y + L_y, & \text{if} \quad \left(\boldsymbol{r}_{ab}\right)_y < -\frac{L_y}{2} \\ \left(\boldsymbol{r}_{ab}\right)_y - L_y, & \text{if} \quad \left(\boldsymbol{r}_{ab}\right)_y > \frac{L_y}{2} \end{cases} \tag{2.30b}$$

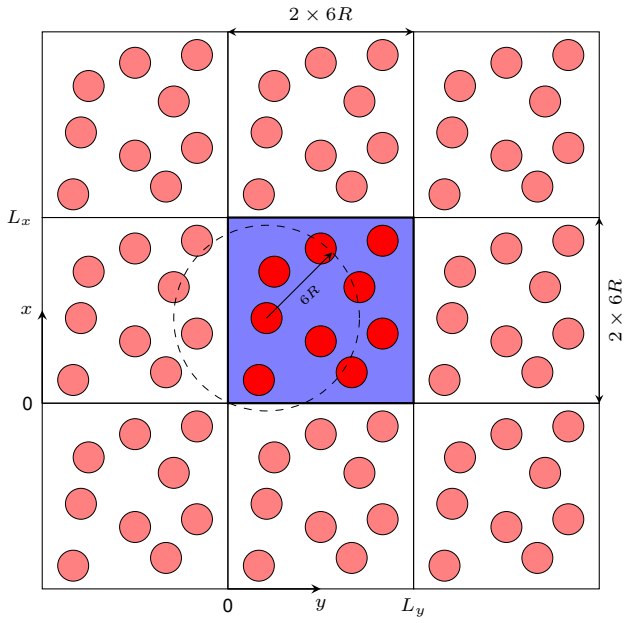


Figure 2.7: Periodic boundary conditions with minimum image convention seen from the yx-plane
Light blue: Simulation domain

2.1.8. Two-way coupled

In order to take the change in the velocity of the fluid due to the particles into consideration a different approach is used. This is called two-way coupling, because the fluid acts on the particle and the particle acts on the fluid. The fluid is described by the incompressible Navier-Stokes equations (2.31).

$$\boldsymbol{\nabla} \cdot \boldsymbol{v}^{\mathsf{f}} = 0 \tag{2.31a}$$

$$\frac{\partial \boldsymbol{v}^{\mathsf{f}}}{\partial t} + \boldsymbol{\nabla} \cdot \boldsymbol{v}^{\mathsf{f}} \boldsymbol{v}^{\mathsf{f}} = -\frac{1}{\rho^{\mathsf{f}}} \boldsymbol{\nabla} p^{\mathsf{mod}} + \frac{\eta^{\mathsf{f}}}{\rho^{\mathsf{f}}} \boldsymbol{\nabla}^2 \boldsymbol{v}^{\mathsf{f}} \tag{2.31b}$$

where p^{mod} is the pressure minus the hydrostatic contribution. Newton's equation (2.1) needs to be modified in order to couple it with the Navier-Stokes equations (2.31). Furthermore, the rotational dynamics are added, transforming it to the Newton-Euler equations (2.32).

$$M_a \boldsymbol{a}_a = \boldsymbol{F}_a^{\mathsf{c}} + \boldsymbol{F}_a^{\mathsf{nc}} + \boldsymbol{F}_a^{\mathsf{ext}} + \rho^{\mathsf{f}} \oint_{\partial V_a^{\mathsf{p}}} \boldsymbol{\sigma}^{\mathsf{f}} \cdot \hat{\boldsymbol{r}}_a^{\mathsf{p}} \, \mathrm{d}A \tag{2.32a}$$

$$I_a \boldsymbol{\alpha}_a = \rho^{\mathsf{f}} \oint_{\partial V_a^{\mathsf{p}}} (\boldsymbol{r}_a^{\mathsf{p}} - \boldsymbol{r}_a) \times \left(\boldsymbol{\sigma}^{\mathsf{f}} \cdot \hat{\boldsymbol{r}}_a^{\mathsf{p}} \right) \mathrm{d}A \tag{2.32b}$$

where $\boldsymbol{\sigma}^{\mathsf{f}} = -p\boldsymbol{I} + \eta^{\mathsf{f}} \left(\boldsymbol{\nabla} \boldsymbol{v}^{\mathsf{f}} + \boldsymbol{\nabla} \left(\boldsymbol{v}^{\mathsf{f}} \right)^{\top} \right)$

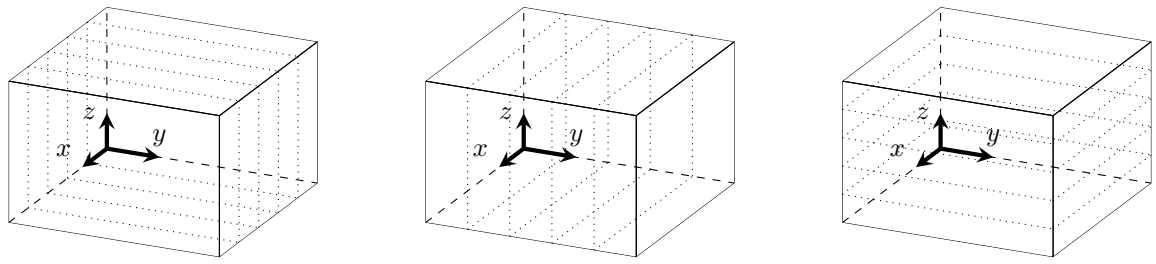
The pressure gradient in the Navier-Stokes equations (2.31) is solved using the IBM (Immersed Boundary Method) and a second-order Crank-Nicolson scheme. With this method, the fluid and the surface of the particles have separated grids. The IBM adds to the immediate vicinity of the particle's surface a force to the right-hand side of the momentum equation (2.31b), such that the velocity of the fluid equals the velocity of a particle segment. With the pressure gradient known, equations 2.31 and 2.32 can be solved using a three step Runga-Kutta algorithm [2].

2.1.9. Slab profiles

Slab profiles can be created in order to get more insight into the steady state behavior of the MRF. The simulation domain is divided into slabs in each direction as illustrated by figure 2.8. To generate a slab profile for the volume fraction, the volume of the particles within a slab is determined. These profiles can show whether particles are aggregating at locations in the simulation domain. Furthermore, the

difference between the mean fluid velocity in the slab and the mean velocity of the particles within the slabs is determined using equation (2.33). These profiles give insight into the shear stress of the MRF. For the volume fraction profiles the data is fitted with a Gaussian fit because of fluctuations that occurs within the profile due to the way particles are stacked. As shown in figure 2.9, the volume fraction in the slabs fluctuate due to the segments of the particles that are present in the slab. The relative mean velocity versus the z-height is fitted with a 3$^{\text{rd}}$ order polynomial. The relative mean velocity versus the x- and y-directions are smoothed out as well.

$$\bar{v}_y^{\mathsf{r}} = \bar{v}_y^{\mathsf{f}} - \bar{v}_y^{\mathsf{p}} \tag{2.33}$$



(a) Slab division in x-direction **(b)** Slab division in y-direction **(c)** Slab division in z-direction

Figure 2.8: Slab division of the simulation domain

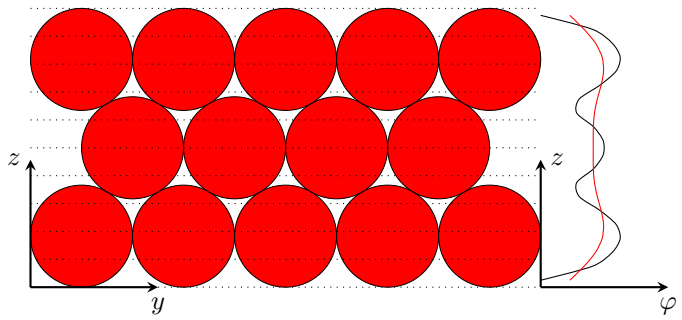


Figure 2.9: Fluctuations of the volume fraction slab profile

2.2. Single core Fortran models

In this section the single core Fortran models will be described. Fortran is used for these models, as it is a high-performance programming language. The single core Fortran models solve Newton's equation (2.1) using the integration scheme of section 2.1.6. Two Fortran models, that use different methods for the magnetic interaction force described in section 2.1.2, will be discussed. The fixed dipole method is used to produce a model with low run times and the mutual dipole method is used to produce a model with a more accurate representation of the magnetic interaction force. The single core Fortran models can be used to simulate the behavior of an MRF in a simulation domain of $L_x \times L_y \times L_z$. As explained in section 2.1.7, boundary conditions are added to the simulation box and a shear rate $\dot{\gamma}$ is applied to the fluid.

This section will first go over the details of the models. Then, the parameters for the simulations will be given. To ensure stability, the stable time step for the models is determined. Thirdly, the number of particles required to make the models independent of N^{p} is determined, after which the shear stresses of the models are checked for a stable system.

2.2.1. Details of the models

Initially the particles are randomly placed within the domain and initial conditions are applied. Each particle starts of with no velocity i.e., $\boldsymbol{v}_a = \boldsymbol{0}$. The particles are generated in such a way that they initially do not overlap with other particles or the walls. The initial positions and velocities are then used to determine the initial acceleration of the particles as described in section 2.1.

After initialization the particles' positions and velocities are updated every time step, Δt, using the modified velocity-Verlet algorithm described in section 2.1.6. Each step the dynamic evolution indicators of section 2.1.5 are stored. The positions and velocities of the particles are not saved every time step, because this would result in very large file sizes. Therefore, it is chosen to store the positions and velocities of the particles 60 times during the whole simulation. The pseudocode is given by algorithm 1 and the source code is given in appendix E.1.

Algorithm 1 Pseudocode for the single core Fortran models

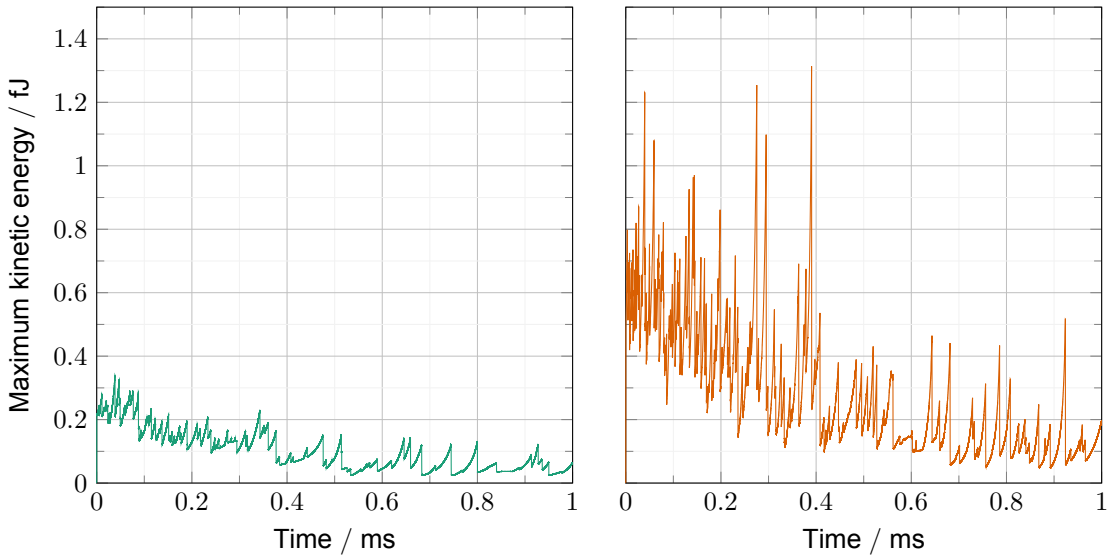
```
Load parameters
t = 0
Initialize particles
Determine and save the dynamic evolution
for each step untill t^final is reached do                ▷ Main loop
    Integration using the modified velocity-Verlet algorithm
    Determine and save the dynamic evolution
    if modulo of step/60 = 0 then
        Store the positions and velocities of the particles
    end if
    t = t + Δt                                            ▷ Update time
end for
```

2.2.2. Parameters

The models are used to simulate the MRF in a uniform and non-uniform magnetic field. The uniform magnetic field only has a component in the z-direction of $H^{\text{ext}} = 0.1\,\text{T}$. The non-uniform magnetic field is generated by a magnet that is homogeneously polarized along the z-direction, with dimensions of $5\,\text{mm} \times 5\,\text{mm} \times 2\,\text{mm}$, a N45 grade, where $B^{\text{rem}} = 1.45\,\text{T}$. The magnet will be placed $2\,\text{mm}$ underneath the simulation domain, just as in Lucieer's experiments [15]. L_x and L_y are determined by the volume fraction, a given number of particles and stating that $L_y = \frac{3}{2}L_x$. This way it is possible to determine how many particles are required to make the simulation independent of the number of particles. This will be discussed below.

2.2.3. Stability

In order to make sure that the simulation is stable, the MRF is initially simulated using a very small time step until the simulation has passed the first kinetic phase, discussed in section 2.1.5. These simulations are done in a uniform magnetic field using the parameters stated in table 2.2, where $\dot{\gamma} = 0\,\text{s}^{-1}$, $N^{\text{p}} = 1000$, $\Delta t = 1\,\text{ns}$ and $t^{\text{final}} = 1\,\text{ms}$. The maximum kinetic energy of these simulations is shown in figure 2.10. The data shows that $K^{\text{max}} < 1.5\,\text{fJ}$, therefore, using equation (2.24c), $\|\boldsymbol{v}_{ab}\|^{\text{max}} < 2\sqrt{\frac{2(1.5\times10^{-15})}{M_a}} \approx 0.15\,\text{m}\,\text{s}^{-1}$. Equation (2.3) than gives that $(t^{\text{c}})^{\text{max}} > 2.45 \times 10^{-7}\,\text{s}$, from which can be concluded that the simulation is stable when $\Delta t \lessapprox 5 \times 10^{-8}\,\text{s}$ with the sampling used by Xuan, Li, Li, ***et al.*** [6]. It should be noted that for both models, the same time step is used, even though the model that uses the fixed dipole method could be simulated with a bigger time step. The model using the fixed dipole method would be stable when $\Delta t \lessapprox 5.5 \times 10^{-8}\,\text{s}$.

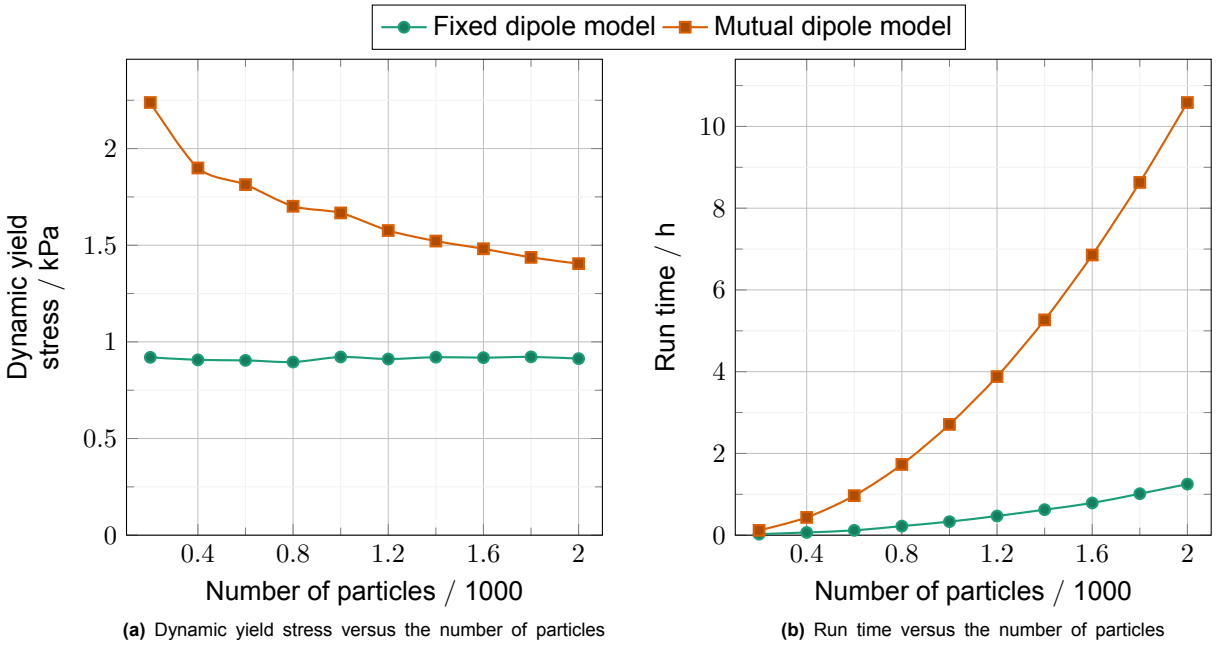


(a) Fixed dipole model **(b)** Mutual dipole model

Figure 2.10: Maximum kinetic energy of the single core Fortran models

2.2.4. Number of particles

In order to ensure accurate results from the simulations, the MRF is simulated using different numbers of particles. N^{p} is varied from 200 to 2000 particles in steps of 200. The MRF is simulated in a uniform magnetic field and with a shear rate of $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$, so that the shear stress can be used to verify when the simulation is independent of N^{p}. The overall results are given in appendix E.2. In order to determine when the simulation is independent of the number of particles, the dynamic shear stress is plotted against N^{p}. The dynamic yield stress is determined by the average of the shear stress after the plateau has been fully reached, which for these simulation is after $t \geq 5\,\mathrm{ms}$. The behavior of the MRF is simulated three times in order to make sure that the differences between each simulation, i.e. the error, is acceptable. The processed results are given in figure 2.11. These results show that the model using the fixed dipole method for the magnetic interaction force does not depend on N^{p} as much as the model using the mutual dipole method does. Furthermore, it can be concluded that the dynamic yield stress for the mutual dipole method has not fully converged yet. Due to the duration of simulation of the MRF using the mutual dipole method, it is chosen to not simulate more than 2000 particles, as this would take upwards of 10 hours. Even though the simulation using the mutual dipole method is not fully independent of N^{p} yet, it is chosen to use 1600 particles for the single core Fortran models to make sure that the MRF can be simulated within a feasible time frame.



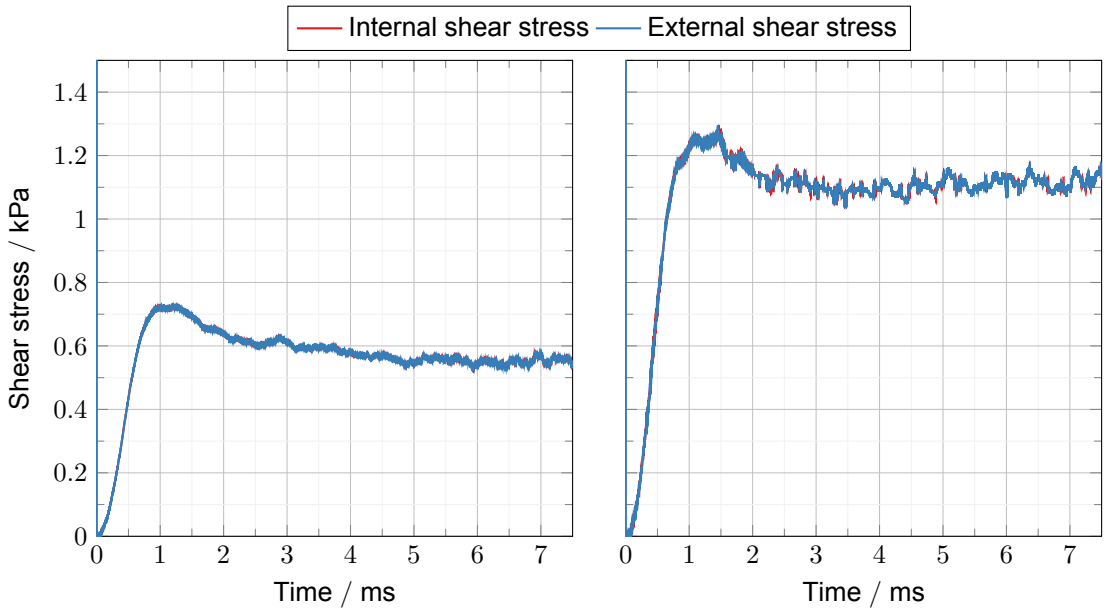
(a) Dynamic yield stress versus the number of particles

(b) Run time versus the number of particles

Figure 2.11: Results of the single core Fortran models with varying N^{p}

2.2.5. Shear stress

Section 2.1.5 states that the internal and external shear stresses of the MRF should be equal in a stable system. This is verified by the simulations used for determining the number of particles required for the simulation. Figure 2.12 shows that the internal and external shear stress of the simulations where $N^{\mathrm{p}} = 1600$, agree.



(a) Fixed dipole model

(b) Mutual dipole model

Figure 2.12: Internal and external shear stress of the MRF where $N^{\mathrm{p}} = 1600$

2.2.6. Verification

The model is verified using the results from Xuan, Li, Li, *et al.* [6]. Their model makes use of the mutual dipole method, therefore only this single core Fortran model can be compared with their results.

Figure 2.13 shows the rheological models for the single core Fortran model, Xuan, Li, Li, *et al.*'s model and Xuan, Li, Li, *et al.*'s experimental results [6].

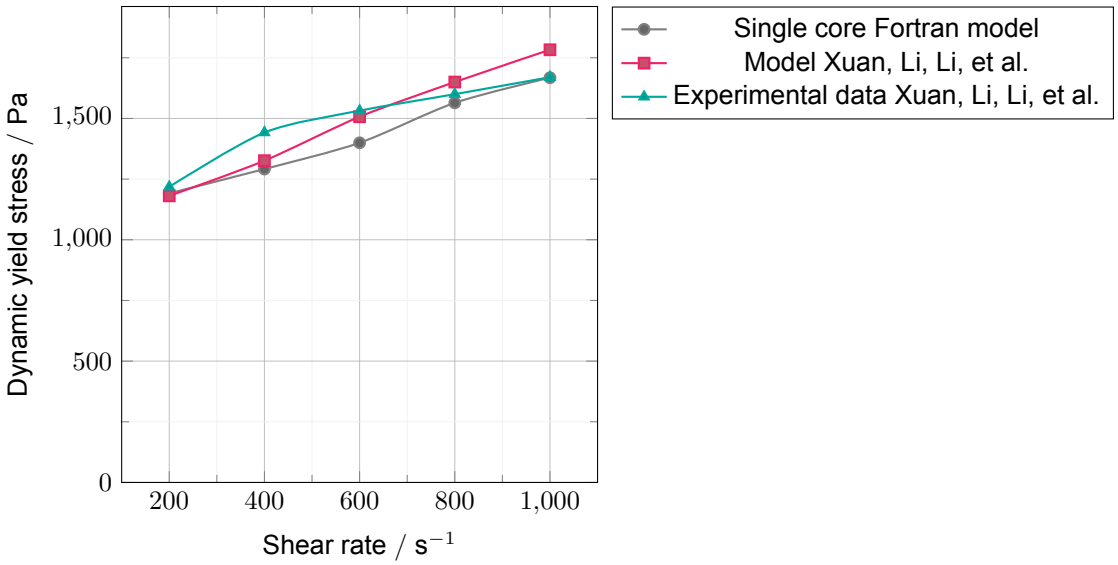


Figure 2.13: Verification of the dynamic yield stresses [6]

The plots shows that model are comparable in the resulting dynamic yield stresses. The small difference may be caused by different definitions of the steady state region. For the single core Fortran model this steady state region was chosen at $t = 5\,\text{ms}$. Both models follow the same trend as the experimental results, however the dynamic yield stresses are somewhat lower at lower shear rates. At lower shear rates the shear stress between the particles dominate. In the models the particles are assumed to be dipoles, which results in lowered interaction forces between the particles and thus lowered dynamic yield stresses. At higher shear rates, the viscous shear stress will dominate, therefore the differences between the model and the experimental data will reduce. The dynamic evolution of the shear stress for the single core Fortran model are given in figure 2.14.

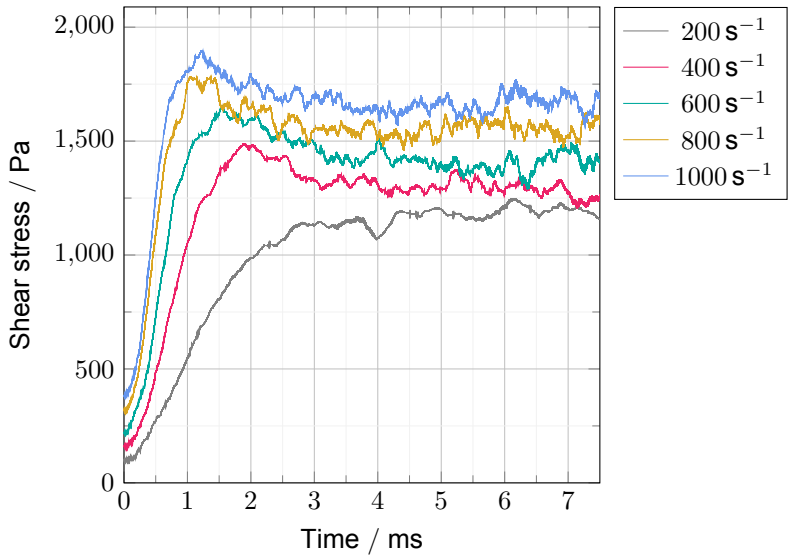
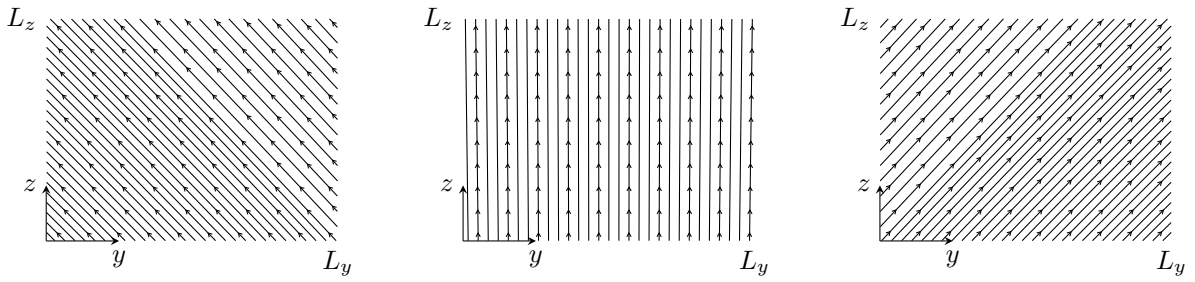


Figure 2.14: Dynamic evolution of the shear stress used for verification of the single core Fortran model

Now that the models are stable, independent of N^{p} and verified using literature, they can be simulated in uniform and non-uniform magnetic fields. The MRF will be simulated using shear rates of 0, 500, 1000, 2500, 5000, 7500 and $10\,000\,\text{s}^{-1}$. Because the models are not able to simulate a large enough domain



(a) Front **(b)** Center **(c)** Back

Figure 2.15: Magnetic field streamlines at the 3 locations of the magnet

for the bar magnet in a feasible time frame, the MRF is simulated at the front, center and back of the magnet. The magnetic fields at these locations are shown in figure 2.15. The results of the simulations will be discussed in section 3.1.

2.3. Parallel Fortran model using OpenMPI

The parallel Fortran model is based on the single core Fortran models, discussed in section 2.2. This model is modified in a way that the MRF can be simulated using multiple cores. Dividing the model over multiple cores means that more particles can be simulated in a feasible time frame. It thus allows for simulating a larger part of the magnetic field produced by a bar magnet. The downside of this model is that it uses the fixed dipole method, discussed in section 2.1.2, for the magnetic interaction force. This magnetic interaction model is used because the mutual dipole method requires a summation over all particles which would greatly increase the communication time between processes. Furthermore, a multi-core machine is needed for this model. OpenMPI is used for the communication between the cores/processes, for which an introduction is given by Fainchtein [57]. The model is simulated on the DHPC (Delft High Performance Computing Centre) [58].

Firstly, the decomposition of the domain and the communications between the processes will be described. Then the details of the model will be given and the minimum required cores are checked. To make sure that resources are not misused, the number of processes required to simulate an MRF within a feasible time frame will be determined. Finally, the parameters for the simulations will be given.

2.3.1. Domain decomposition

The total simulation domain of the single core Fortran code is divided into columns. Each column represents a process using one core. This is called domain decomposition, of which an overview is given in figure 2.16. The particles that are present in a column are mastered by that process. Therefore, the behavior of these particles is only determined in this core, which greatly reduces the simulation times.

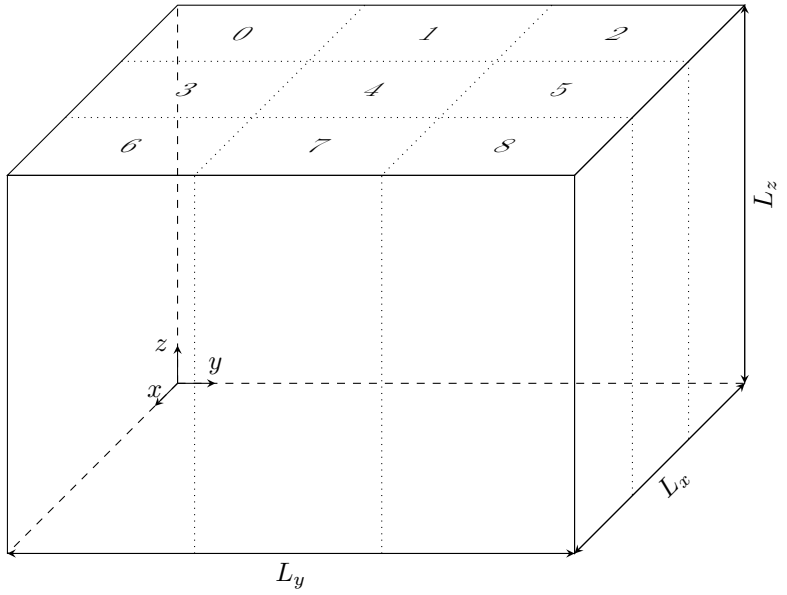


Figure 2.16: Domain decomposition where each processes controls a column of the simulation domain

The MRF is only properly simulated if the processes are able to communicate with each other. For this model each process only communicates with its direct neighbors to decrease the time lost on communication. Each process thus communicates with 8 other processes, taking PBCs into account. To make sure that all particles within the cut-off distance are included, the dimension of a process must be at least equal to the cut-off distance i.e., $L_x^{\text{proc}} \geq 6R$ and $L_y^{\text{proc}} \geq 6R$. An overview is given by a top view projection in figure 2.17 where the light blue square is the simulation domain, the dark blue cell is a process within the domain and the marked squares are the neighboring 8 processes. The figure shows why the dimension of a process needs to be at least the cut-off distance. It is chosen to divide the domain into columns because this way the simulation domain can be divided into more processes. If the domain was divided another way, less processes that still meet the minimum required process dimension could be used. It should be noted, that the minimum dimension for the simulation domain, discussed in section 2.1.7, should still be taken into account.

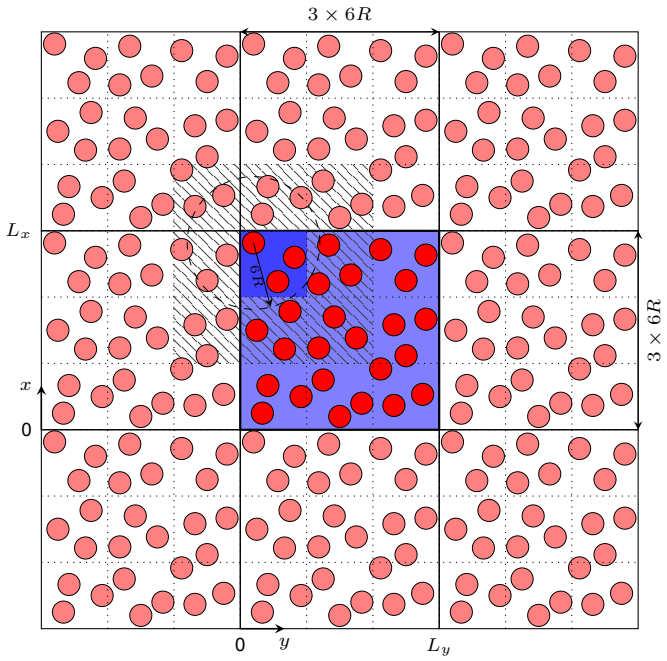


Figure 2.17: Communication between processes including PBCs seen from the yx-plane
Light blue: Simulation domain
Dark blue: Current process
Marked: Neighbors of the current process

2.3.2. Details of the model

The model starts by initializing the OpenMPI communication and dividing the domain into columns. Then particles are randomly generated within the whole simulation domain and they are assigned to the process the particles are currently in. For each particle within a process, the initial acceleration is determined by the forces acting on this particle by particles in the current process and its neighbors.

After the initialization, the model uses the same process explained in section 2.2.1 to simulate the behavior of the MRF. The only difference is that each time the position of the particles is updated, it is checked whether particles have moved into a different process. If this is the case, the particle's data is transferred between the old and the new process. The pseudocode is given by algorithm 2 and the source code is given in appendix F.1.

Algorithm 2 Pseudocode for parallel Fortran model using OpenMPI

```
Load parameters
t = 0
Initialize the processes
Divide the simulation domain                          ▷ Domain decomposition
Initialize and assign particles
Determine and save the dynamic evolution
for each step until t^final is reached do             ▷ Main loop
    Integration using the modified velocity-Verlet algorithm
    Determine and save the dynamic evolution
    if modulo of step/60 = 0 then
        Store the positions and velocities of the particles
    end if
    t = t + Δt                                        ▷ Update time
end for
```

2.3.3. Minimum number of cores

In order to make sure that the parallel Fortran model is accurate, the model is verified with the single core Fortran model that uses the fixed dipole method. The simulation used to check the stability of the single core Fortran models, are done using the parallel Fortran model as well, but for this model the domain is divided into a number of processes. The domain will be divided in to 1×1, 2×2, 3×3 and 3×5 processes to determine the minimum processes required in a direction. It is not possible to verify the model using more processes, because then the minimum process dimension requirements discussed in section 2.3.1 would not hold. The parameters for these simulations are the same as in section 2.2.3. The moving mean of the maximum kinetic energies of these simulations are shown in figure 2.18. The plot shows that a minimum of 3×3 processes is needed in order to compare the model to the single core Fortran model. Therefore, the same stable time step can be used as determined in section 2.2.3.

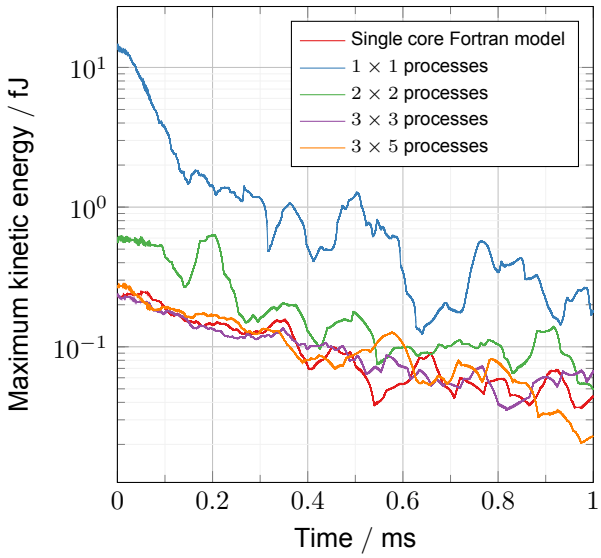
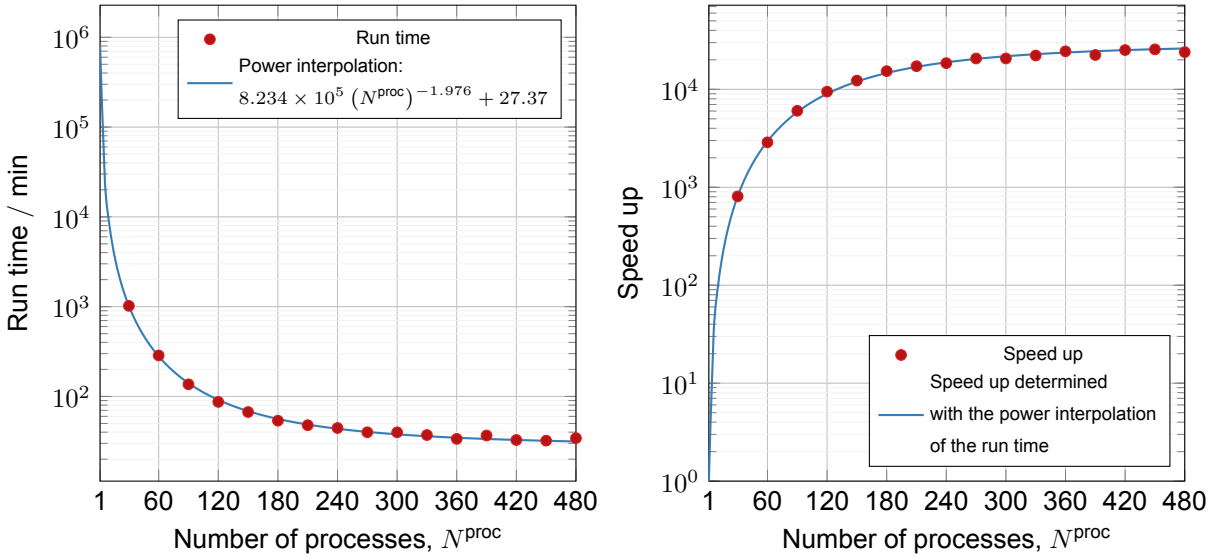


Figure 2.18: Moving average of the maximum kinetic energy of the parallel Fortran models using different domain divisions

2.3.4. Cores vs speed

If the domain is divided into many columns, large amounts of cores are needed to simulate the MRF. To make sure that DHPC's resources are not misused, the number of processes that are required to simulate the MRF within a feasible time frame is determined. This is achieved by varying N^{proc} from 3×10 processes to 3×160 processes in steps of 30. The same parameters are used as discussed in section 2.3.3, but with a simulation domain of $L_x = 3 \times 6R$ and $L_y = 8000R$. 206265 particles will thus be simulated. Due to applied shear rate, the flow direction is of the most interest for simulating an MRF, therefore the dimension in y-direction is larger. The run times of these simulations are shown in figure 2.19. By interpolating the run times of the multi-core simulations, the run time for a single core is estimated to be 8.234×10^5 min. The speed up for each simulation is then determined by dividing the run time for a single core, by the run time of a multi-process simulation. Figure 2.19 shows that the single core Fortran model can be sped up approximately 2.5×10^4 times when dividing the domain into 480 processes. Furthermore, the data shows that the speed up versus the number of cores has not converged, from which can be concluded that the simulation can be even faster. Due to a lack of resources, it was not possible to determine the run time with more than 480 cores. There is not a big difference in run times for simulating the MRF using 240 cores or 480 cores. It is thus chosen to use 240 processes for the remaining simulations to save resources.

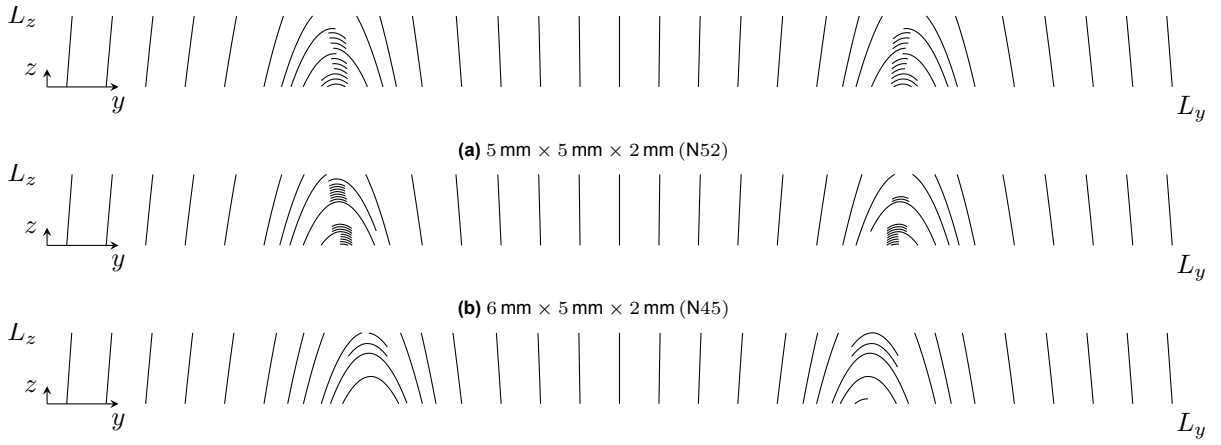


(a) Run time, versus the number of processes, N^{proc}

(b) Speed up of the model vs the number of processes, N^{proc}

Figure 2.19: Run time versus cores

Now that the model is stable and the domain division is set, the model can be simulated in a uniform magnetic field and non-uniform magnetic fields. The MRF will be simulated for the same shear rates used in section 2.2. Three different bar magnets that were used by Lucieer are simulated using this model. The first is a $5\,\text{mm} \times 5\,\text{mm} \times 2\,\text{mm}\,(\text{N}52)$ bar magnet, the second a $6\,\text{mm} \times 5\,\text{mm} \times 2\,\text{mm}\,(\text{N}45)$ bar magnet and the third a $7\,\text{mm} \times 6\,\text{mm} \times 2\,\text{mm}\,(\text{N}45)$ bar magnet. Just as in Lucieer's experiments, the magnets are placed $2\,\text{mm}$ under the simulation domain [15]. The magnetic fields generated by these magnets in the simulation domain are shown in figure 2.20 and appendix C. The results of these simulations will be given in section 3.2.



(a) $5\,\text{mm} \times 5\,\text{mm} \times 2\,\text{mm}\,(\text{N}52)$

(b) $6\,\text{mm} \times 5\,\text{mm} \times 2\,\text{mm}\,(\text{N}45)$

(c) $8\,\text{mm} \times 4\,\text{mm} \times 2\,\text{mm}\,(\text{N}45)$

Figure 2.20: Magnetic fields

2.4. Breugem's DNS code with magnetic particles

To check whether two-way coupling is required for the models discussed in sections 2.2 and 2.3, a parallel DNS (Direct Numerical Simulation) model that is used to simulate particulate flows is modified so it can simulate the behavior of an MRF [2]. This model will only be used to simulate the MRF in a uniform magnetic field, as this will be enough to check whether two-way coupling is required. The same parameters as described in section 2.2 will therefore be used.

This section will start by proposing modifications to Breugem's model [2] after which some details of the modified model will be discussed.

2.4.1. Modification

The model that will be modified is discussed in more detail by Breugem [2]. The model uses the equations from section 2.1.8 to simulate the behavior of particulate flows by incorporating IBM forces, soft-sphere collisions [2] and lubrication corrections [59]. Next to this, in order to simulate an MRF, the model should take the magnetic interaction force into account. Therefore, the model is modified to include the fixed dipole method. This method is used for the same reasons discussed in section 2.3. The magnetic interaction force is made dimensionless using the Mason Number of equation (2.34), which defines the ratio between the hydrodynamic and magnetic stresses in the system [17], [21].

$$\mathrm{Mn} = \frac{8\eta^{\mathrm{f}}\dot{\gamma}}{\mu_0\beta^2 H_0^2} \tag{2.34}$$

2.4.2. Details of the model

The model is structured the same way as discussed in section 2.3.2, but instead of solving Newton's equation (2.1) using the modified-velocity Verlet method, the Navier-Stokes equations (2.31) and Newton-Euler equations (2.32) are solved by a third-order, three-step Runga-Kutta integration scheme. The pressure gradient from the Navier-Stokes equations (2.31) is solved using a second-order Crank-Nicolson scheme. Furthermore, the particles are initialized with an initial velocity that corresponds to the velocity of the fluid at their centroid.

As stated before, the modified model is only simulated in a uniform magnetic field with the same shear rates as the other models, in order to check whether two-way coupling is required. The results of this simulation are given in section 3.3.

3

Results

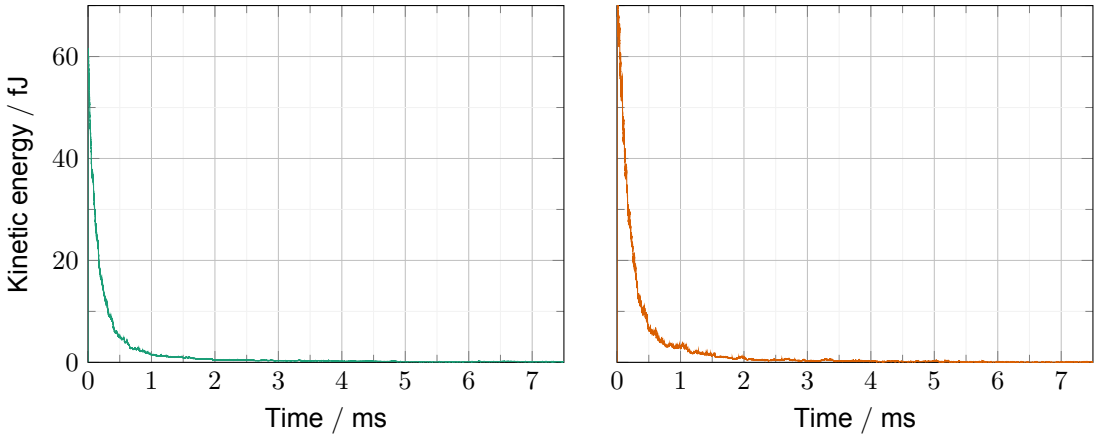
In this chapter the results of the models discussed in chapter 2 will be given. Firstly, the results of the single core Fortran models will be shown, after which the results of the parallel Fortran model will be given. This chapter will end with the results of Breugem's DNS code with magnetic particles.

3.1. Single core Fortran models

This section will give an overview of the results of the models discussed in section 2.2, with the parameters described in section 2.2.2. The section will show the results of the uniform field and the non-uniform field that are discussed at the end of section 2.2. Only the most relevant results will be discussed in detail, results that are not discussed can be found in appendix E.3. At first, the kinetic energy for the simulations is given to check if steady state is reached, after which the resulting formation of structures in the models will be given. Then, the shear stresses over time of both models will be shown from which the dynamic yield stress is determined. These dynamic yield stresses are then used to produce a rheological model for the MRF in uniform and non-uniform magnetic fields.

3.1.1. Steady state

The simulations need to have reached steady state in order to determined the dynamic yield stress of the MRF. The kinetic energy can be used to determine whether the steady state has been reached. When steady state is reached mainly depends on the formation of structures in the MRF. The kinetic energies of the two models where $\dot{\gamma} = 0\,\mathrm{s}^{-1}$ are shown in figure 3.1. These plots show that the simulations have reached a steady state. The simulations where no shear rate applied to the MRF take the longest to reach steady state. Whether steady state is reached for the other shear rates has also been checked. Therefore, the simulations where $\dot{\gamma} \neq 0\,\mathrm{s}^{-1}$ can be used to determine the dynamic yield stress of the MRF.



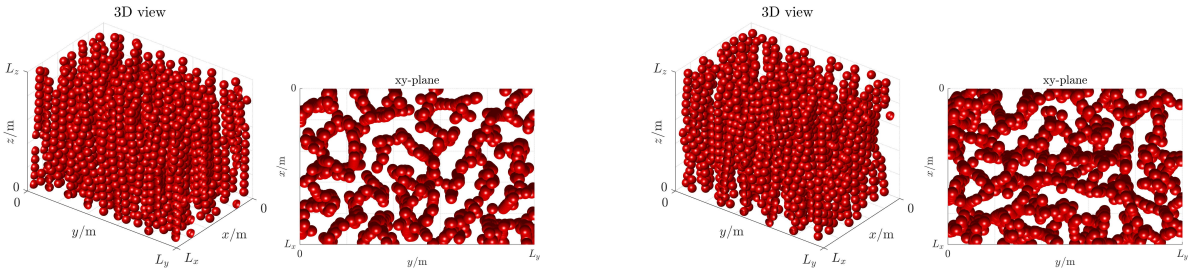
(a) Fixed dipole model **(b)** Mutual dipole model

Figure 3.1: Kinetic energy of the single core Fortran models where $\dot{\gamma} = 0\,\mathrm{s}^{-1}$

3.1.2. Steady state structures

As stated before in chapter 1, the particles within the MRF will form structures when a magnetic field is applied. These structures will form along the field lines to form chains. An overview of the steady state structures of all the simulations is given in appendix E.3.1.

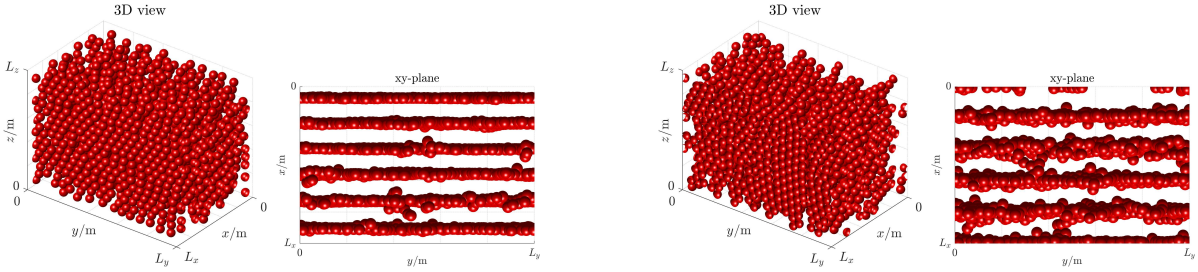
Figure 3.2 shows the steady state structures of the single core Fortran models when $\dot{\gamma} = 0\,\mathrm{s}^{-1}$ in a uniform magnetic field. As expected the particles form chain-like structures along the magnetic field lines in the z-direction. Because the magnetic field is oriented this way, a top view shows how the clusters are formed and intertwined.



(a) Fixed dipole method **(b)** Mutual dipole method

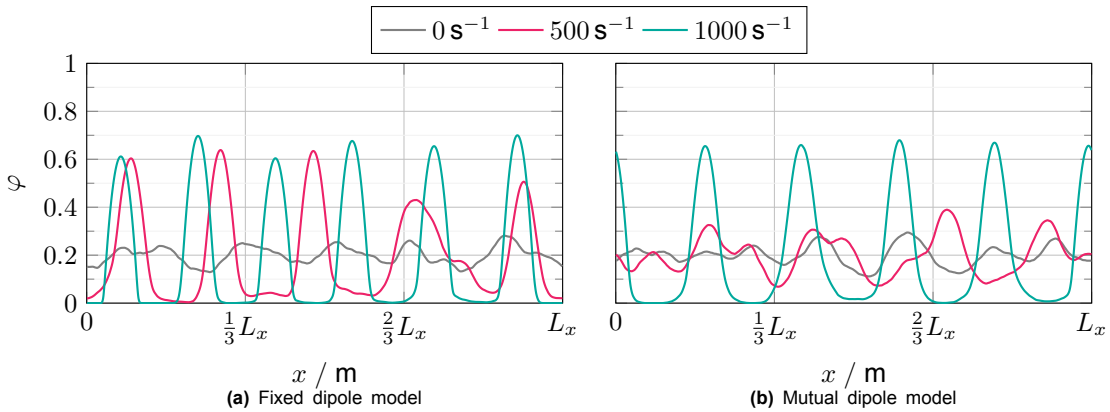
Figure 3.2: Steady state structures in a uniform magnetic field where $\dot{\gamma} = 0\,\mathrm{s}^{-1}$

An increase in the shear rate will cause the chains to tilt in the steady state phase and form planes, shown in figure 3.3. These planes become gradually less intertwined with an increasing shear rate. For large shear rates, the particles form evenly spaced planes which span from the bottom wall to the top wall, a thickness of one particle and with maximum packing. The formation of these planes can also be seen in figure 3.4, which shows the relation between the volume fraction x-direction.



(a) Fixed dipole method **(b)** Mutual dipole method

Figure 3.3: Steady state structures in a uniform magnetic field where $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$



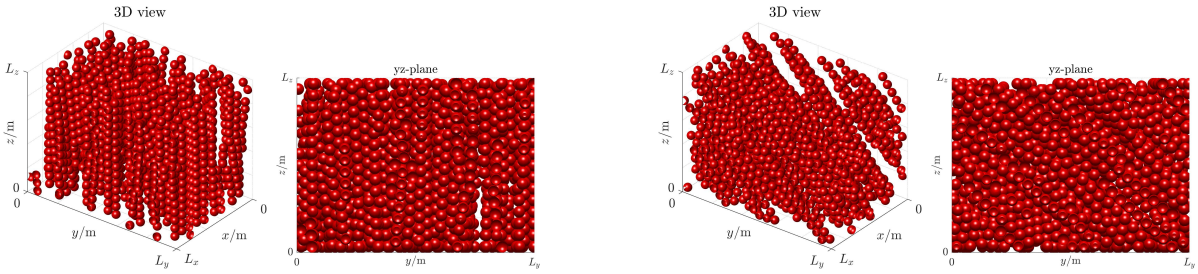
(a) Fixed dipole model **(b)** Mutual dipole model

Figure 3.4: Slab volume fraction profiles in x-direction in a uniform field

Even though for both models the particles are aligned with the magnetic field, there is a clear difference in the formation of the planes with an increasing shear rate. The figures of the model where the mutual dipole method is used, show that the planes are still intertwined with each other for higher shear rates. After a shear rate of $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$, not much difference in the steady state structures can be seen. The figures from appendices E.3.1 and E.3.4 show that the same holds for the MRF in a non-uniform field. The particles form chains along the magnetic fields and for higher shear rates they form planes of tightly packed particles.

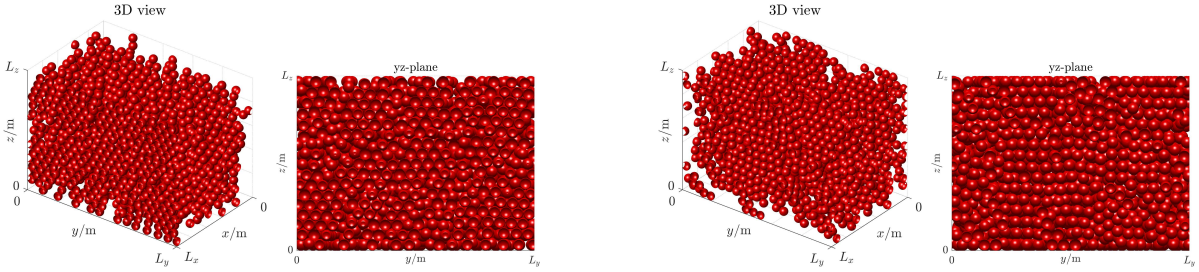
Aggregation on the bar magnet

Figures 3.5 and 3.6 show the steady state structures of an MRF in two different fields for two shear rates. These figures show that the particles form structures along the magnetic field lines, but it is hard to see whether particles are aggregating on the bar magnet in a non-uniform field. Therefore, the sedimentation of particles due to the external magnetic field is determined using the mean z-height and the relation between the volume fraction and the film height. The figures show that the particle chains deform when a shear rate is applied to the MRF.



(a) Center of the magnet **(b)** Front of the bar magnet

Figure 3.5: Steady state structures in two non-uniform magnetic fields where $\dot{\gamma} = 0\,\mathrm{s}^{-1}$

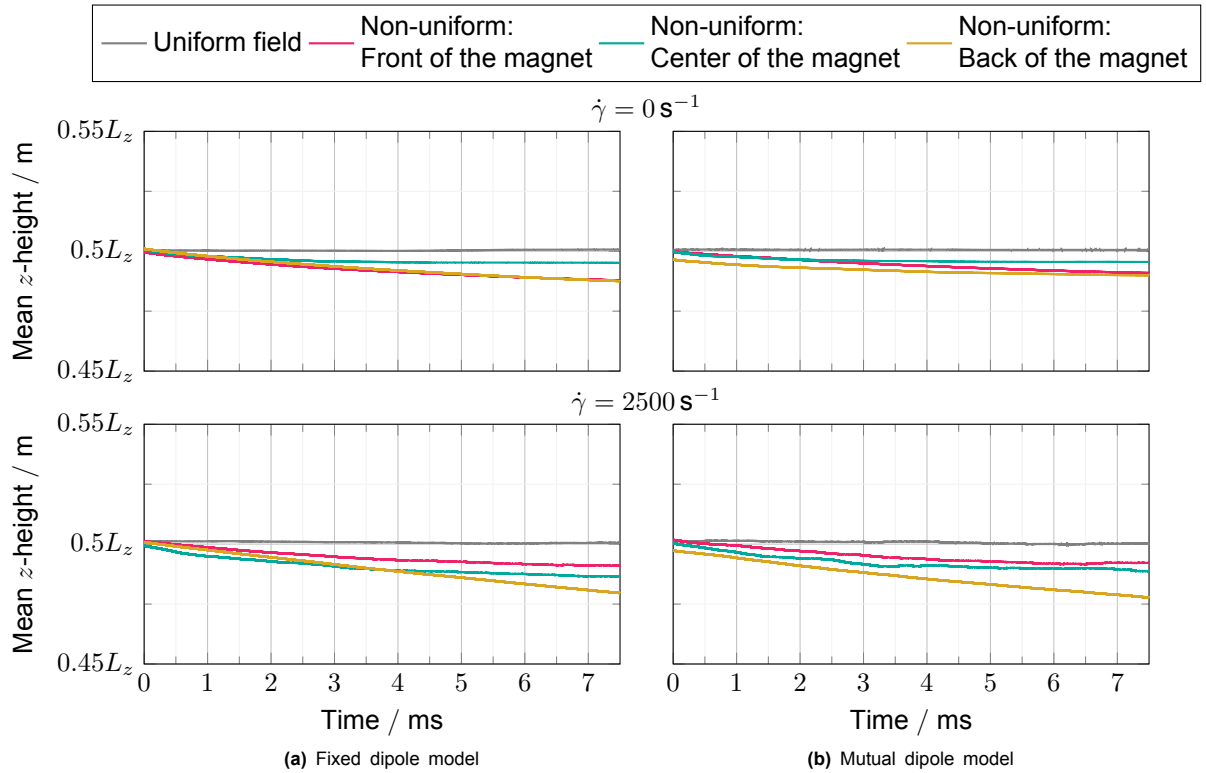


(a) Center of the magnet **(b)** Front of the bar magnet

Figure 3.6: Steady state structures in two non-uniform magnetic fields where $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$

The mean z-height gives an indication of the sedimentation that occurs due to the attractive force of the bar magnet on the particles. $\bar{z}$ is tracked over time and plotted in figure 3.7 for the uniform magnetic field and the three locations of the bar magnet, where $\dot{\gamma} = 0\,\mathrm{s}^{-1}$ and $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$. Figure 3.7 shows that the mean z-height does not vary over time in a uniform magnetic field. In the non-uniform fields there is a decline in $\bar{z}$ over time.

For increasing shear rates the mean z-heights for the non-uniform magnetic fields change. At the front of the magnet the mean z-height remains approximately the same at steady state and at the center and back of the magnet the mean z-height decreases. It should be noted that $\bar{z}$ is not converged yet.



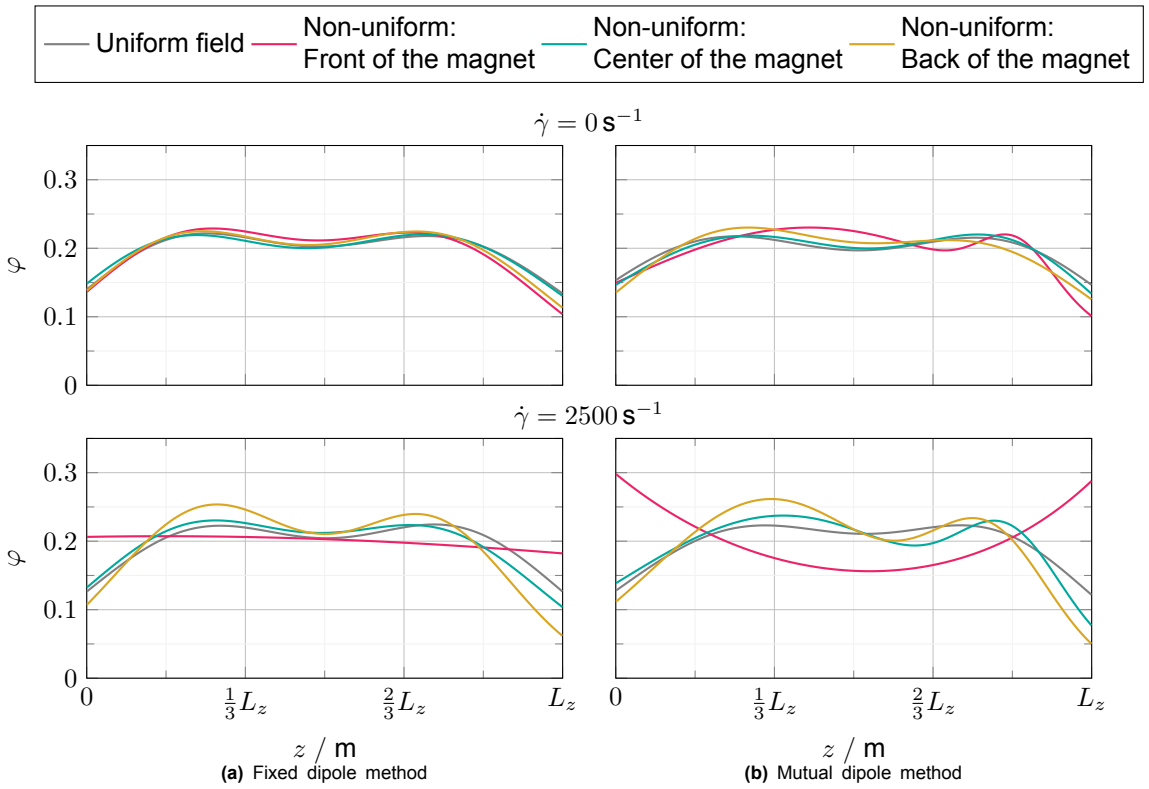
(a) Fixed dipole model **(b)** Mutual dipole model

Figure 3.7: The mean z-height for the different fields

The volume fraction versus the film height is another indicator whether particles are moving towards the magnet. The volume fractions in z-direction of the MRF at steady state are shown in figure 3.8 for the different magnetic fields where $\dot{\gamma} = 0\,\mathrm{s}^{-1}$ and $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$. For the model that uses the fixed dipole method only small differences in the density can be seen for the different magnetic fields when $\dot{\gamma} = 0\,\mathrm{s}^{-1}$. These differences are most visible at the top of the simulation domain, where the volume fraction decreases in the non-uniform fields. The mutual dipole method shows larger differences in volume fraction over the film height for the different magnetic fields. At the front of the magnet, the volume fraction at the top wall decreases while increasing in the center. At the center of the magnet,

the volume fraction increases at the bottom half of the simulation domain and decreases at the top half. This also holds at the back of the magnet, but the increases and decreases are larger.

At the higher shear rate of $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$, larger differences in the density start to show. At the front of the magnet, the density at the vicinity of the walls increases, while the density in the center of the film height decreases. At the center of the magnet, the density in the lower part of the domain increases, while the density decreases at the top of the simulation domain. The same happens at the back of the magnet, but the differences are larger.The model that uses the mutual dipole method shows larger increases and decreases than the model that used the fixed dipole method, especially at the front of the magnet.



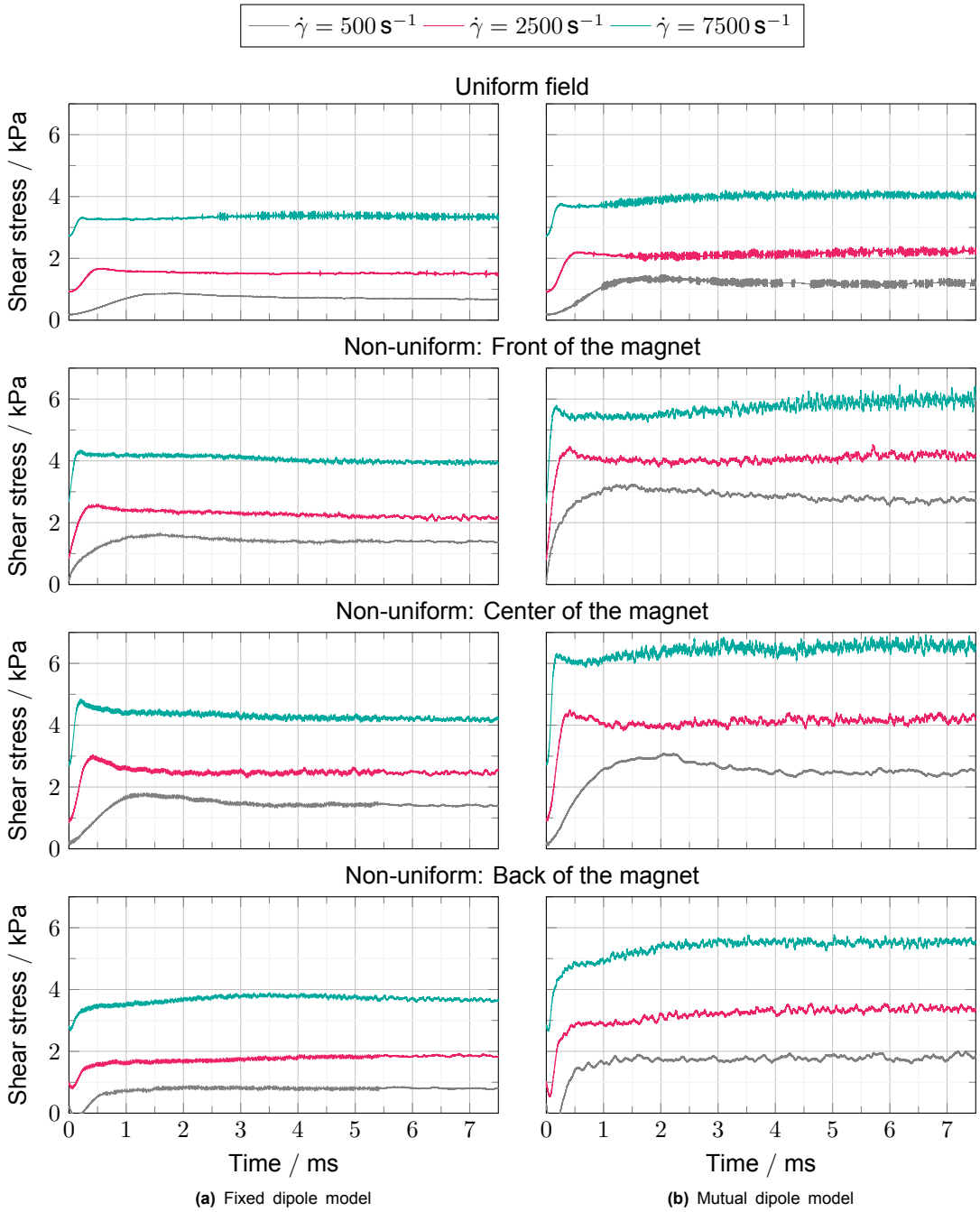
(a) Fixed dipole method **(b)** Mutual dipole method

Figure 3.8: Slab volume fraction in z-direction for the different fields

3.1.3. Shear stresses

The shear stresses of the fluid under different shear rates in different fields are shown in figure 3.9. In the uniform magnetic field at shear rates below $2500\,\mathrm{s}^{-1}$, the regions that were discussed in section 2.1.5 are clearly visible. At higher shear rates the shear stress will follow a different curve after the static yield stress is reached. Instead of declining to a plateau, the shear stress increases to the dynamic yield stress. The dynamic yield stresses are thus larger than the static yield stresses at higher shear rates. At the front and center of the magnet, the shear stress responses using the fixed dipole method, show the regions described in section 2.1.5. The model that uses the mutual dipole method shows at high shear rates that the dynamic yield stress will be larger than the static yield stress. At the back of the magnet, the results show that the dynamic yield stress is always larger than the static yield stress.

It holds for all models that both the static and dynamic yield stresses increase with an increasing shear rate, which is also shown in the rheological models below. Furthermore, at higher shear rates the shear stress reaches the plateau, and thus a steady state, faster.



(a) Fixed dipole model **(b)** Mutual dipole model

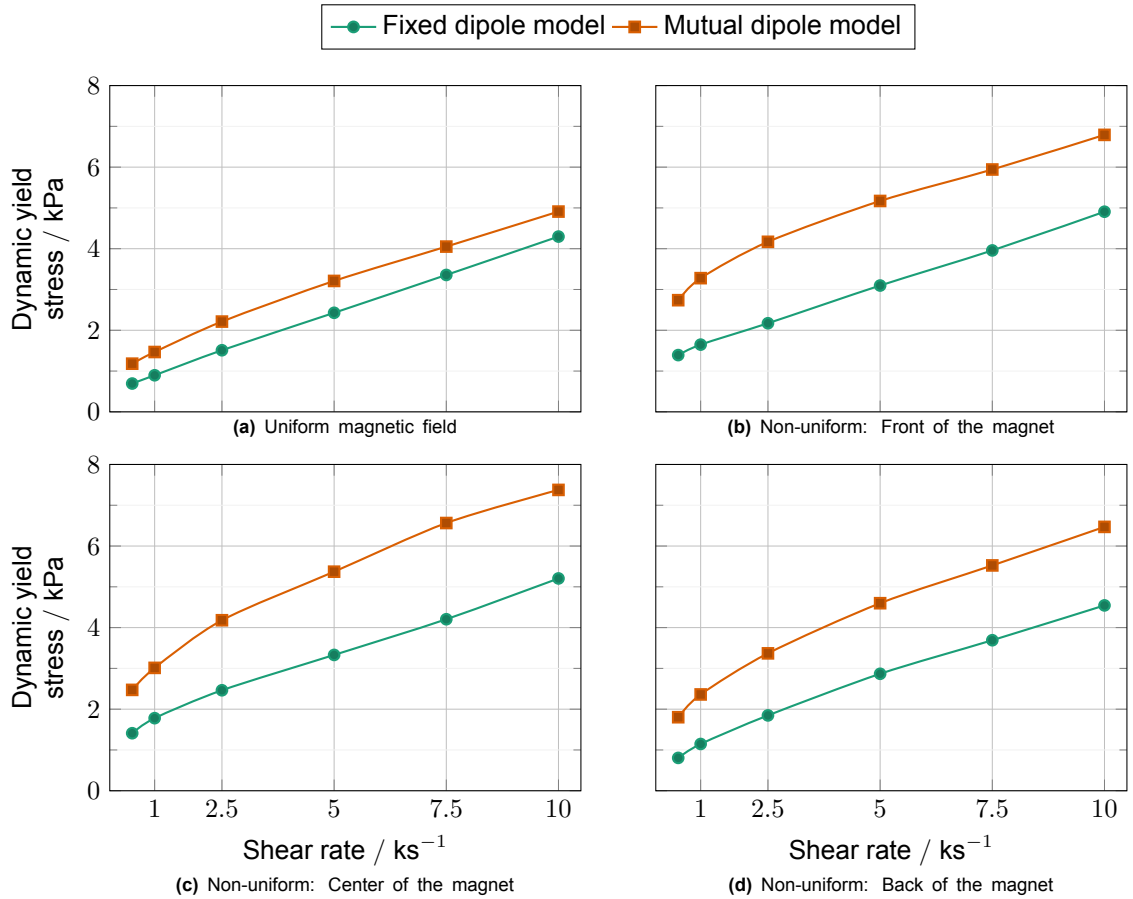
Figure 3.9: Shear stress over time in different magnetic fields for different shear rates

3.1.4. Rheological models

Rheological models for the single core Fortran models under the different magnetic fields are constructed using the dynamic yield stresses found in section 3.1.3. The models are shown in figure 3.10, where the dynamic yield stress versus shear rate is plotted. All models show an increase of the dynamic yield stress for increasing shear rates. At lower shear rates, the rate of increase is larger than at higher shear rates. This is more apparent when the mutual dipole method is used. The slope of the curves decrease at shear rates higher than $2500\,\mathrm{s}^{-1}$. Furthermore, using the mutual dipole method results in larger dynamic yield stresses.

The rheological models for the non-uniform field show different dynamic yield stresses. At the center of the magnet, the dynamic yield stresses are the highest. The dynamic yield stresses at the front of the magnet are almost as large as at the center of the magnet. At the back of the magnet, the dynamic

yield stresses are the smallest.



(a) Uniform magnetic field

(b) Non-uniform: Front of the magnet

(c) Non-uniform: Center of the magnet

(d) Non-uniform: Back of the magnet

Figure 3.10: Rheological models

3.2. Parallel Fortran model using OpenMPI

This section will give an overview of the results of the model discussed in section 2.3. The MRF is simulated in a uniform magnetic field and three non-uniform magnetic fields produced by three different bar magnets that were given at the end of section 2.3. Only the most relevant results will be discussed in detail. Results that are not discussed can be found in appendix F.2. First, the steady state structures will be discussed using the volume fraction slab profiles, the mean z height and the density plots. Then the dynamic yield stress will be determined using the shear stress over time plots. These yield stresses are used to construct rheological models.

3.2.1. Steady state structures

The parallel Fortran model is used to simulate large domains with a large numbers of particles. The model is unable to show the steady state structures within the MRF due to the proportions of the simulation domain. Therefore, the steady state structures of the MRF are inspected using the slab volume fraction profiles in all directions, the mean z height of the particles and density plots. Firstly, the results of the formation of planes will be shown, then the results that show the aggregations in the simulation domain will be given. Finally, the results of the steady state structures for an increased t^{final} will be shown.

Formation of planes

The volume fraction profiles in x-direction are shown in figure 3.11. Just as in the single core Fortran models, the volume fraction versus the x-direction shows that an increase of the shear rate will cause the particles to form planes parallel to the fluid flow direction. In this model, there are no empty spaces between the sheets. When there is no flow in the system, the volume fraction over the x-direction remains constant. The volume fraction profiles show differences between the fields. The volume fraction in a uniform field increases faster with an increasing shear rate than the non-uniform fields. The first bar magnet has the largest increases for an increasing shear rate, while the increases for the other two bar magnets are approximately the same.

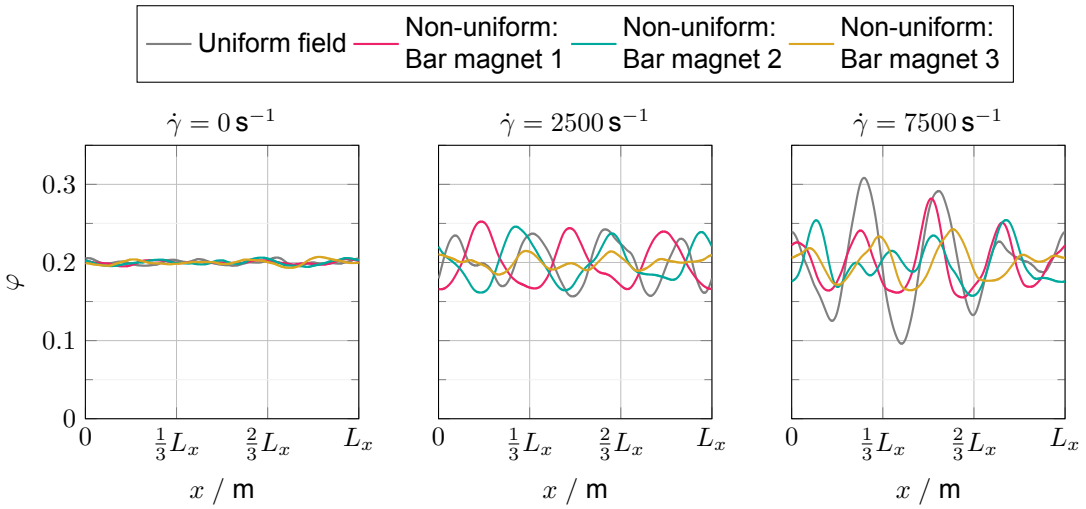


Figure 3.11: Slab volume fraction profiles in x-direction for different magnetic fields

Using density plots of the yx-plane in figures 3.12 and 3.13, the formation of planes is made visible. Only the density for the uniform magnetic field and bar magnet 1 will be given, the other density plots are shown in appendix F.2.4. These plots show that the particles form intertwined sheets parallel to the flow direction at increasing shear rates. For the non-uniform field, these planes only form on top of the magnet.

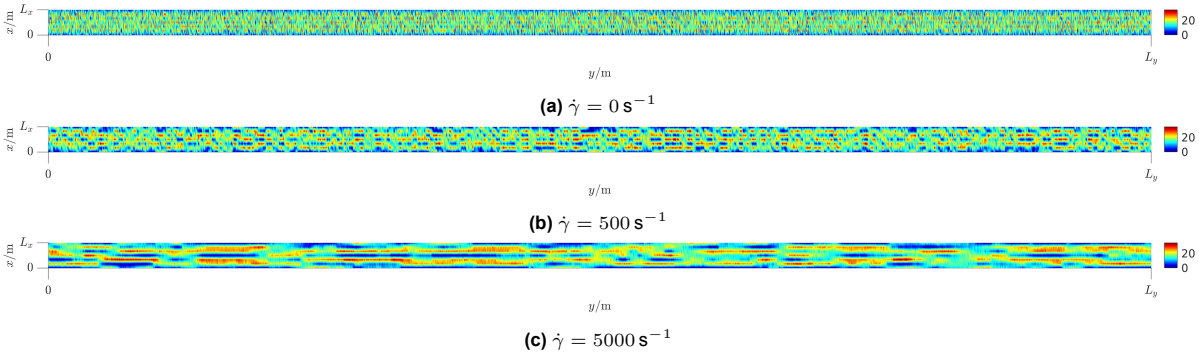


Figure 3.12: Density plot of the yx-plane for a uniform field
Color bar: Number of particles per $4R \times 3R$ m^2

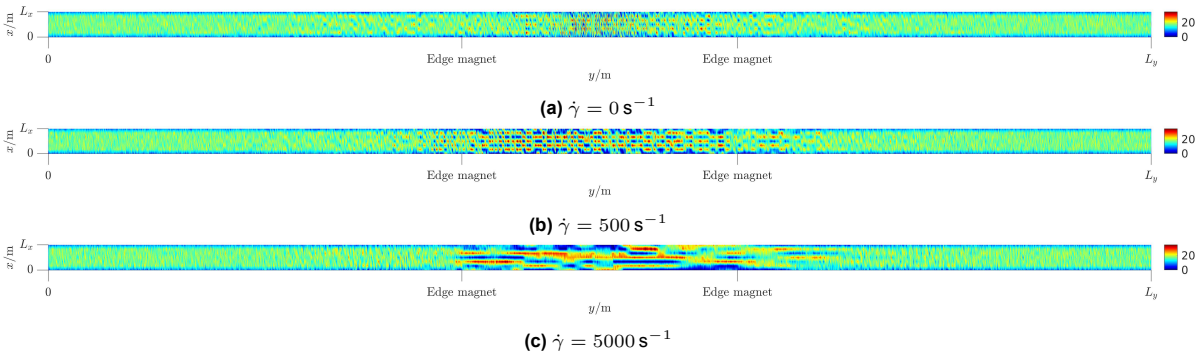


Figure 3.13: Density plot of the yx-plane using bar magnet 1
Color bar: Number of particles per $4R \times 3R$ m^2

Aggregation on the magnet

The volume fraction versus the y-direction of figure 3.14 shows that the density of particles is constant when $\dot{\gamma} = 0\,\mathrm{s}^{-1}$. An increasing shear rate will result in a fluctuation in the density at the back half of the magnet. These fluctuations do not appear in the uniform magnetic field. At the back half of the magnet the volume fraction increases, after which the volume fraction decreases. The plots show that the fluctuations increase with increasing shear rates. The increases of the volume fraction are larger for the first bar magnet and approximately the same for the second and third bar magnets.

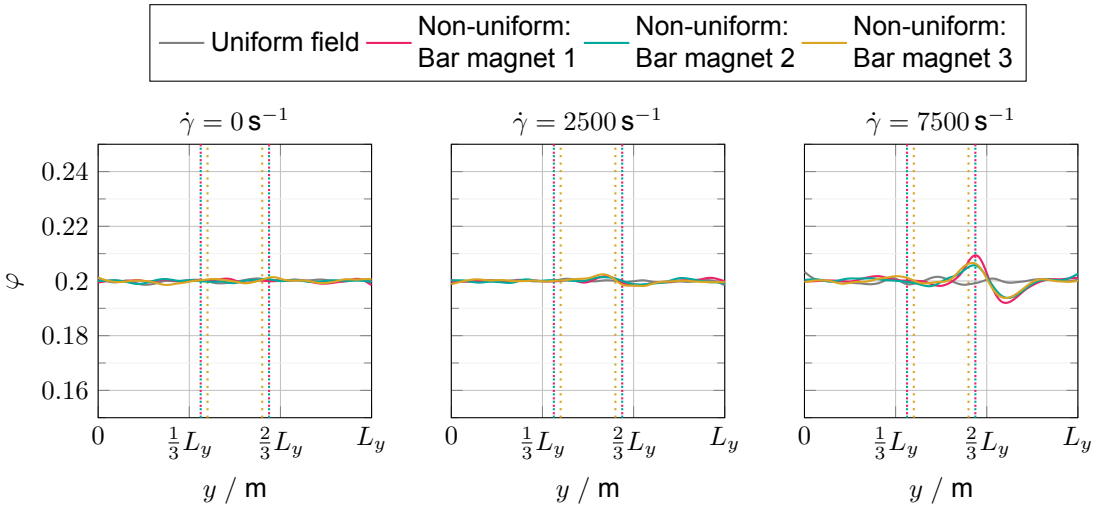


Figure 3.14: Slab volume fraction profiles in y-direction for different magnetic fields
Dotted lines: Edges of the bar magnets

The mean relative velocity in each slab in the y-direction is shown in figure 3.15 for different shear rates. The plot shows a spike in the mean relative velocity at the back of the magnet. This spike increases for higher shear rates and does not appear when no flow is present. The plots show that the spikes are the highest for the first bar magnet. The spikes are smallest for the second and third bar magnets For higher shear rates, the mean relative velocity increases at the front of the magnet as well. The increases do not appear in the uniform magnetic field.

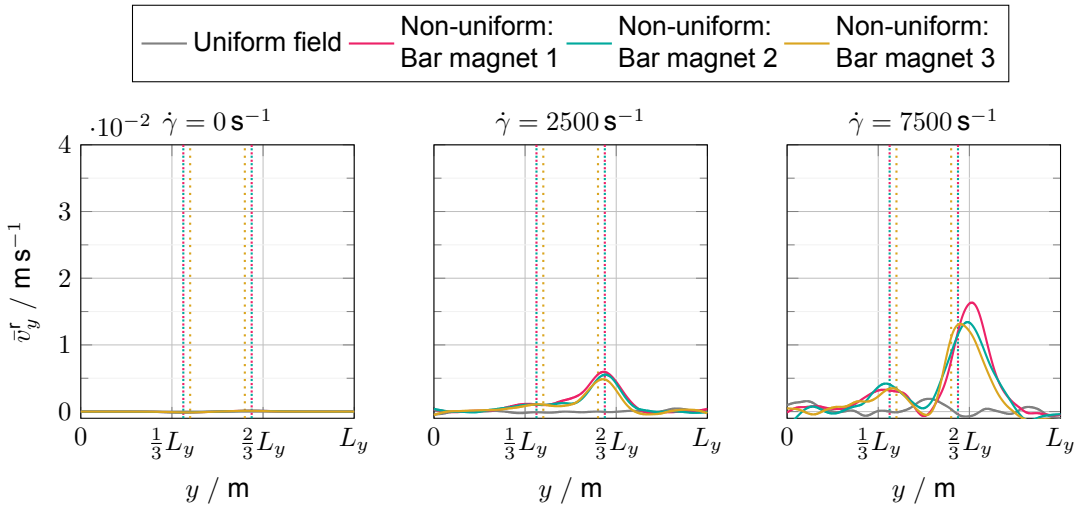


Figure 3.15: Relative slab velocities for the different magnetic fields
Dotted lines: Edges of the bar magnets

The volume profiles in the z-direction for the different magnetic fields are given in figure 3.16. The volume fractions between the magnetic fields do not differ much when $\dot{\gamma} = 0\,\mathrm{s}^{-1}$. There is a minute increase of the volume fraction at the bottom wall and a decrease at the top wall for each bar magnet. For higher shear rates, differences in the volume fraction start to appear. For each bar magnet, an increase of the volume fraction can be seen at the bottom wall. Furthermore, the plots show a small increase of the volume fraction at the top wall. At the center of the film height, an overall decrease of the volume fraction can be seen. The differences between the uniform field and the non-uniform fields increase with increasing shear rates. The relation between the volume fraction and z-height is approximately the same for all bar magnets.

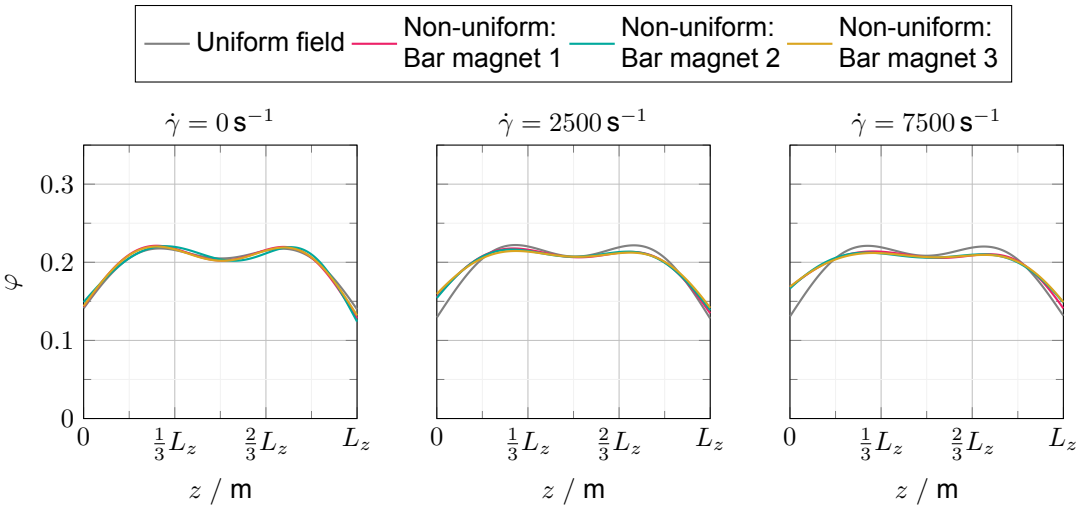


Figure 3.16: Slab volume fraction profiles in z-direction for different magnetic fields

Density plots of the yz-plane are made to determine whether particles are forming a texture at the back of the magnet. These plots are used to show differences in densities with respect to the density when $\dot{\gamma} = 0\,\mathrm{s}^{-1}$. Figure 3.17 shows the density plot for the uniform field when no shear is applied to the MRF. The plot shows large densities at the center of the film height and the number of particles is small at the top wall and bottom wall. When a shear rate is applied, density in the center of the film height increases, as shown in figure 3.18.

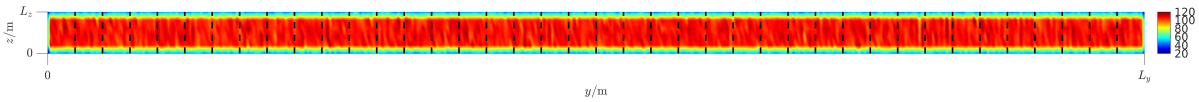
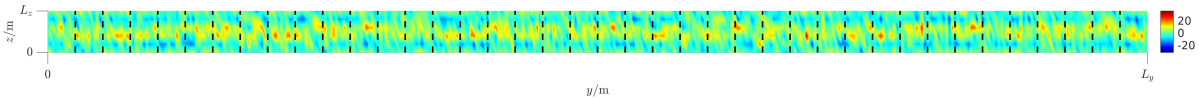
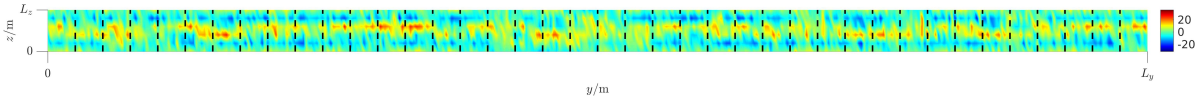


Figure 3.17: Density plot when $\dot{\gamma} = 0\,\mathrm{s}^{-1}$ in a uniform field
Color bar: Number of particles per $20R \times 6R$ m^2
Dashed lines: Magnetic field



(a) $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$
Color bar: Difference in number of particles per $20R \times 6R$ m^2
Dashed lines: Magnetic field



(b) $\dot{\gamma} = 7500\,\mathrm{s}^{-1}$
Color bar: Difference in number of particles per $20R \times 6R$ m^2
Dashed lines: Magnetic field

Figure 3.18: Density differences with respect to figure 3.17

Figure 3.19 shows the density plot for bar magnet 1 when there is no shear applied. Just as in the uniform field, the plot shows a band of large densities at the center of the film height, however there are some differences. A smaller number of particles is present in front of the magnet, where there is a change in the gradient of the magnetic field. At the back of the magnet, the same decrease of the density is visible where the gradient of the field changes. Applying a shear rate to the fluid results in changes of the distribution of particles. Figure 3.20 shows the increases and decreases of the number of particles with respect to the density plot where no shear rate is applied. For lower shear rates, the plot shows that particles have moved downwards from the top wall at the back of the magnet. At higher shear rates there are increases of the density along the whole film height. At the front of the magnet, a reduction of the number of particles in the center of the film height and an increase of the density at the walls are visible. Furthermore, the density at the top wall increases at the center of the magnet for higher shear rates.

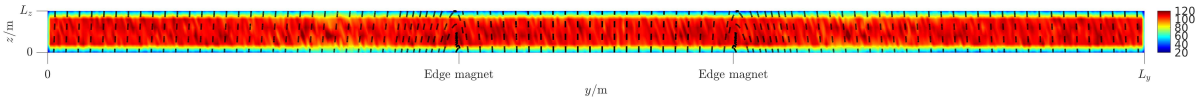
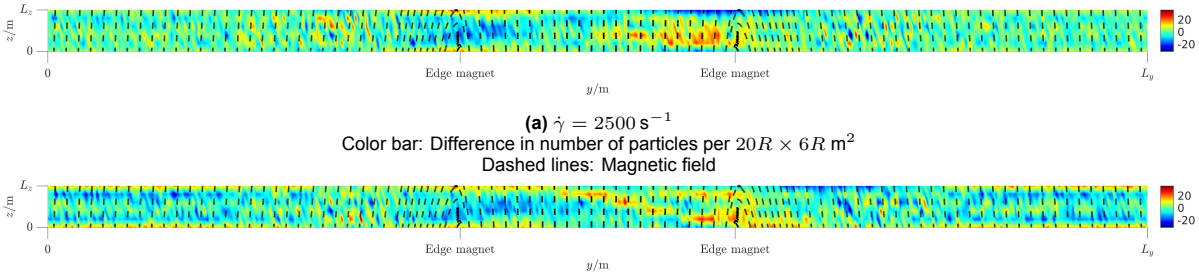


Figure 3.19: Density plot when $\dot{\gamma} = 0\,\mathrm{s}^{-1}$ for bar magnet 1
Color bar: Number of particles per $20R \times 6R$ m^2
Dashed lines: Magnetic field



(a) $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$
Color bar: Difference in number of particles per $20R \times 6R$ m^2
Dashed lines: Magnetic field

(b) $\dot{\gamma} = 7500\,\mathrm{s}^{-1}$
Color bar: Difference in number of particles per $20R \times 6R$ m^2
Dashed lines: Magnetic field

Figure 3.20: Density differences with respect to figure 3.19

The mean z-height of the particles for different shear rates is shown in figure 3.21. The results show that the mean z-height of the particles is constant for the uniform field and decreases over time for all shear rates in non-uniform fields. The first bar magnet shows the largest decline, followed by the second bar magnet. The third bar magnet shows the smallest decline. The mean z-height decreases faster when a shear rate is present.

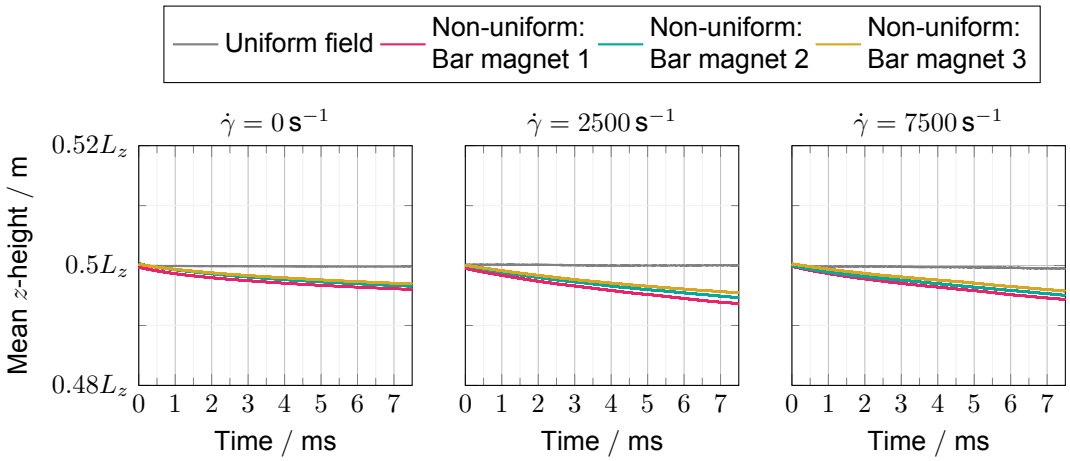


Figure 3.21: Mean z-height over time for the different fields

Increased t^{final}

The differences in the mean z-height and the slab volume fraction profiles in y- and z-direction get more apparent when t^{final} is increased to 0.05 s, as shown in figure 3.22. The slab volume fractions in y-direction show increases at the front half of the magnet and after the back of the magnet. Decreases are visible in front of the magnet. The volume fraction, where there is no shear applied to the MRF, remains almost constant in y-direction. The volume fraction in z-direction shows increases of the volume fraction at the bottom wall and decreases at the top wall when a shear rate is applied. Finally, the mean z-height shows that applying a shear rate will result in a faster decrease of $\bar{z}$.

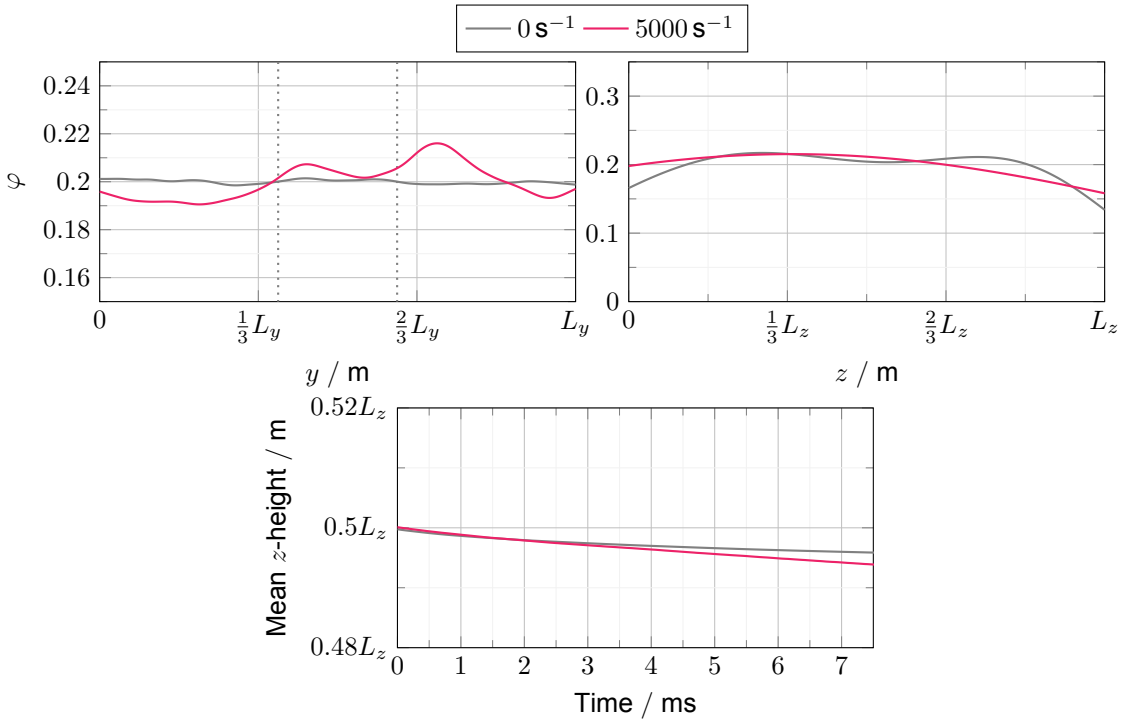
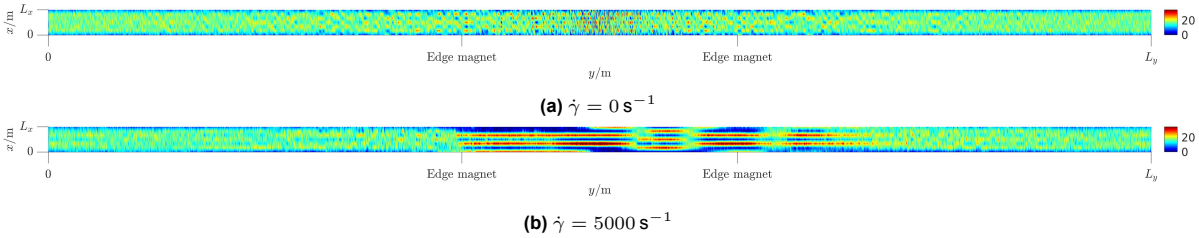


Figure 3.22: Mean z-height and slab volume fractions for the increased t^{final} using bar magnet 1

A density plot of the yx-plane is shown in figure 3.23 for the two shear rates using bar magnet 1 for the increased t^{final}. These plots show the formation of sheets on top of the magnet when a shear rate is applied. These planes are less intertwined than the sheets found in figure 3.13.



(a) $\dot{\gamma} = 0\,\text{s}^{-1}$

(b) $\dot{\gamma} = 5000\,\text{s}^{-1}$

Figure 3.23: Density plot of the yx-plane for the increased t^{final} using bar magnet 1
Color bar: Number of particles per $4R \times 3R$ m^2

The density plot of the yz-plane when there is no shear applied is shown in figure F.17. Just as the plots shown before, decreases are visible just in front and after the magnet, where the gradient of the magnetic field changes. Some slight increases at the bottom wall at the edges of the magnet are visible. The differences in the density of the number of particles when the shear rate is applied is shown in figure 3.25. An increase of the density is visible just after the magnet at the bottom wall, where the gradient of the magnetic field is the largest. Around the front of the magnet there are decreases in the density at the center of the film height. At the top wall and bottom wall at these locations, increases are visible. In the center of the magnet, the density increases overall. Finally, at the edges of the simulation domain, the density of particles has decreased.

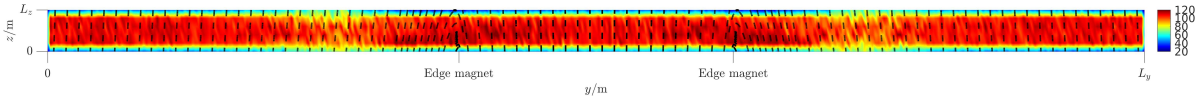


Figure 3.24: Density plot when $\dot{\gamma} = 0\,\mathrm{s}^{-1}$ for the increased t^{final} using bar magnet 1
Color bar: Number of particles per $20R \times 6R$ m^2
Dashed lines: Magnetic field

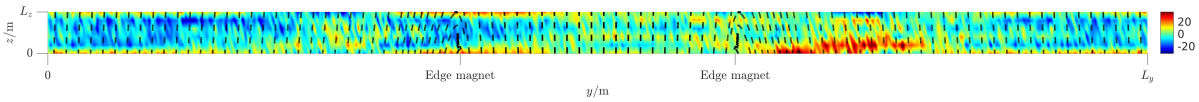


Figure 3.25: Density differences with respect to figure 3.24 where $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$ using bar magnet 1
Color bar: Difference in number of particles per $20R \times 6R$ m^2
Dashed lines: Magnetic field

3.2.2. Shear stresses

The shear stresses of the MRF in the different fields over time are shown in figure 3.26. The regions discussed in section 2.1.5, are clearly visible for all of the shear stress curves. In the uniform magnetic field, the shear stress follows the same curves as in the single core Fortran model. For all models the static and dynamic yield stresses increase with an increasing shear rate. The results show that the shear stress curves for the first and second magnet are almost identical. The shear stress of the third bar magnet follows the same curve, but with lowered shear stress values.

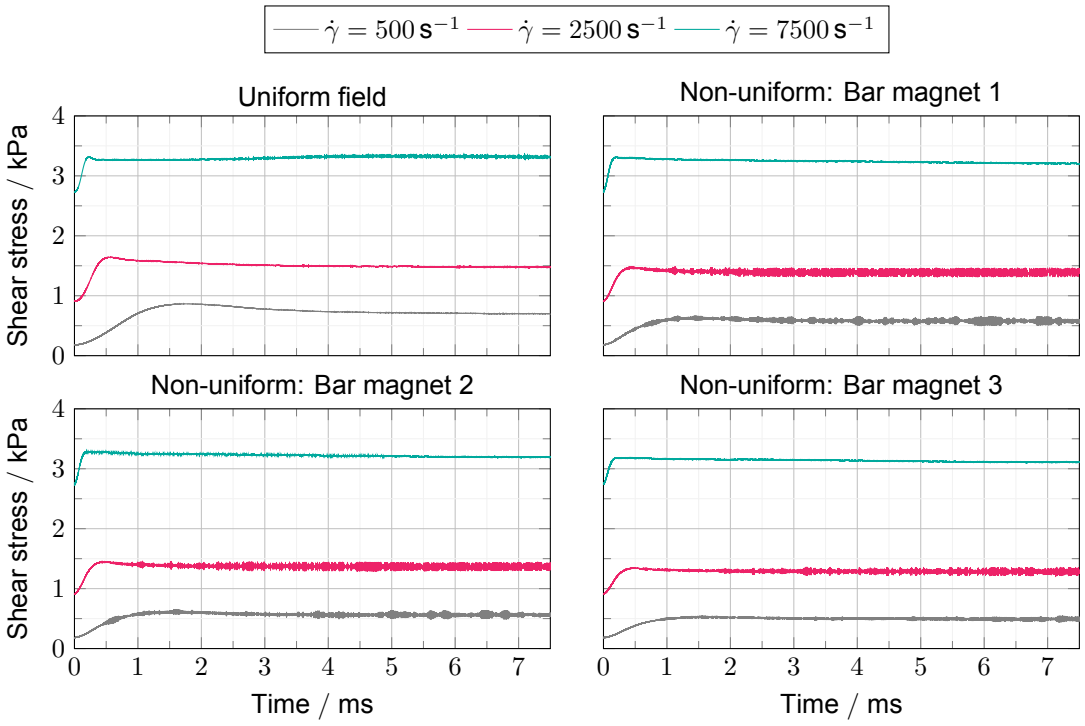


Figure 3.26: Shear stress over time for different shear rates

3.2.3. Rheological models

From the results of section 3.2.2 the dynamic yield stress of the MRF can be determined for each shear rate. These yield stresses are used to construct the rheological models for each magnetic field, shown in figure 3.27. The rheological model for the uniform field is the same as the rheological model shown in figure 3.10.

For all bar magnets the rheological models show an increase of the dynamic shear stress with an increasing shear rate. The first and second bar magnet show larger dynamic yield stresses than the

third magnet. For all magnetic fields, the rheological models follow the same trend.

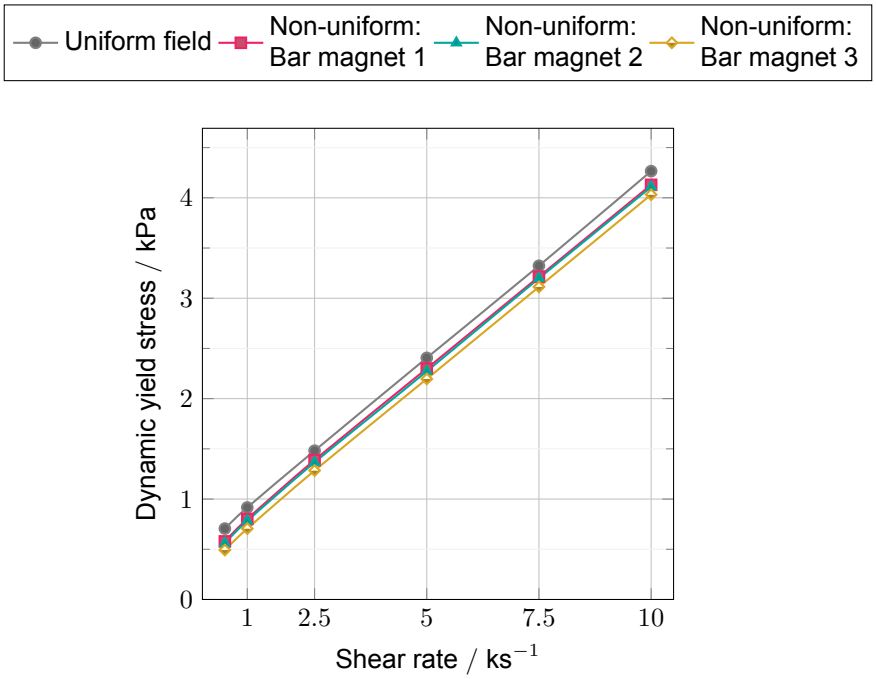


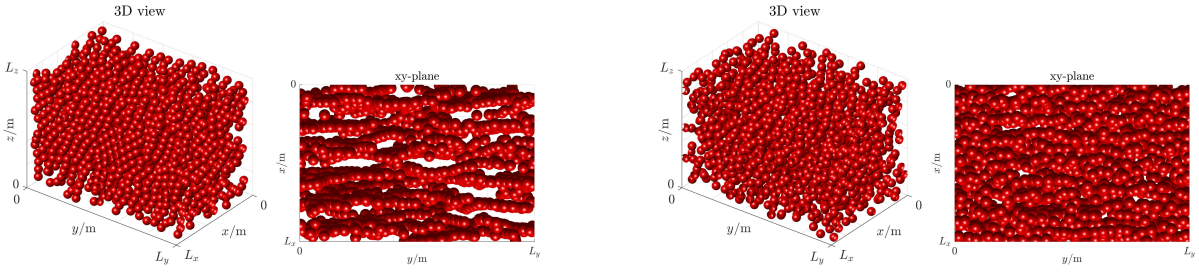
Figure 3.27: Rheological models

3.3. Breugem's DNS code with magnetic particles

This section will give the most relevant results for the model discussed in section 2.4. Only the most relevant results will be given. Results that are not shown can be found in appendix G. First, the steady state structures of the MRF will be shown. Then, the shear stress response will be compared to the other models. Using the shear stress response, the dynamic yield stresses are determined and a rheological model is set up. This rheological model is shown and compared to the other models.

3.3.1. Steady state structures

The MRF is simulated in a domain that is approximately the same as the single core Fortran model. The model thus simulates approximately 1600 particles. The steady state structures of the MRF are shown in figure 3.28 for two different shear rates. It is impossible to simulate the MRF with no shear rate using the DNS model. The plots show that the particles have formed chains that are rotated due to the applied shear rate. There is not much difference between the steady state structures for different shear rates. The particles do not seem to form planes of tightly packed particles at higher shear rates.



(a) $\dot{\gamma} = 500\,\mathrm{s}^{-1}$ **(b)** $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$

Figure 3.28: Steady state structures in a uniform magnetic field for different shear rates

3.3.2. Shear stress

Figure 3.29 shows the shear stress response of the MRF for Breugem's modified DNS model and the parallel Fortran model for different shear rates. The shear stress responses show the same curves as were found by the parallel Fortran model and the single core Fortran model that uses the fixed dipole method. The static yield stress is reached at a later time for the DNS model. Therefore, steady state is reached at a later time in the simulation. At lower shear rates the differences between the models are larger than at higher shear rates.

Just as in the other models, the shear stresses in the MRF get larger with an increase of the shear rate. The dynamic yield stresses from these curves are determined in order to construct a rheological model.

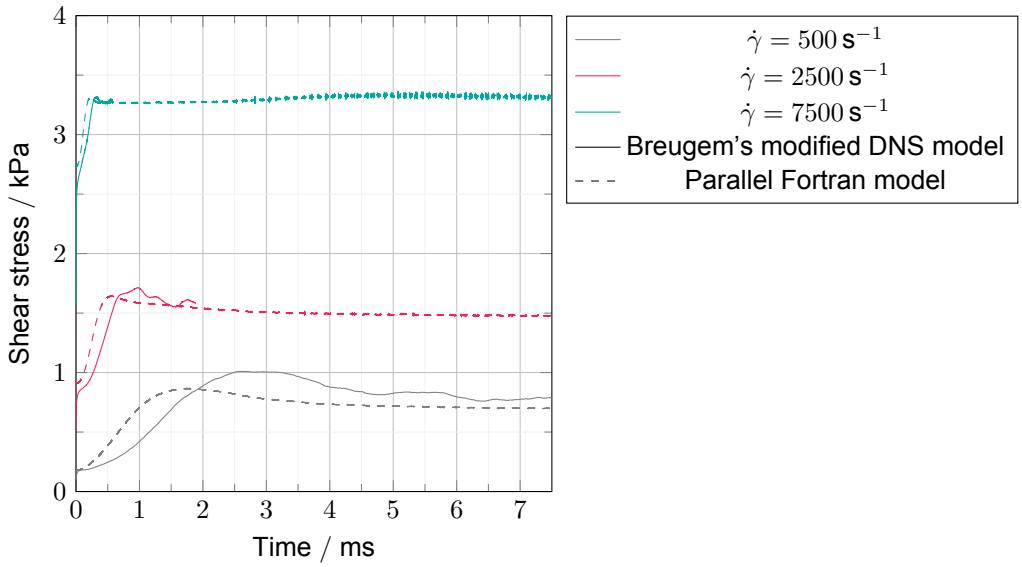


Figure 3.29: Shear stress response of the MRF using Breugem's modified DNS model

3.3.3. Rheological model

The rheological model using the DNS model, the parallel Fortran model and the single core Fortran model that uses the fixed dipole method are shown in figure 3.30. The rheological model that is determined using the DNS model shows slightly larger values of the dynamic yield stress at lower shear rates. At shear rates above $5000\,\mathrm{s}^{-1}$ the models are approximately the same. Overall the curves follow more or less the same trend.

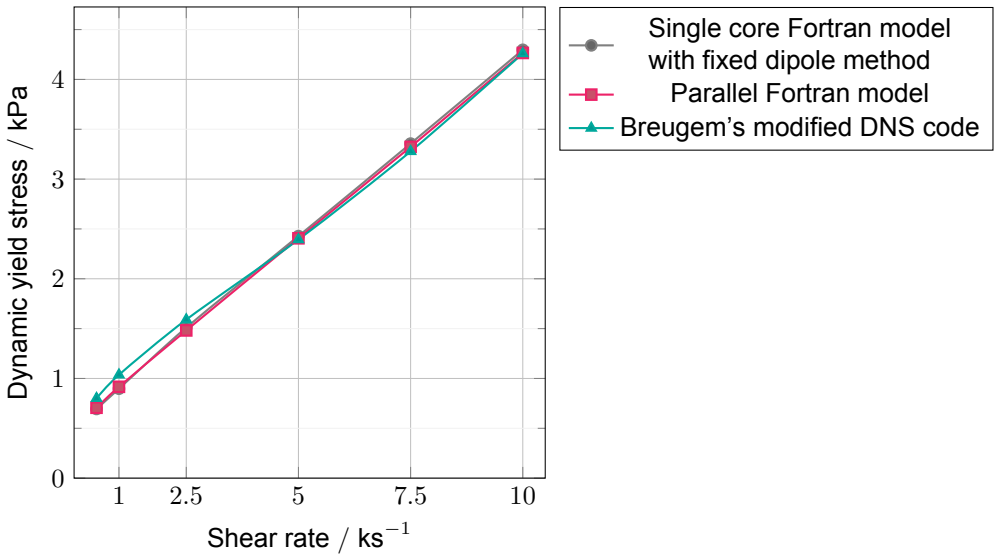


Figure 3.30: Rheological models for the models using the fixed dipole method

4

Discussion

The chapter will discuss the results from chapter 3. Firstly, the results of the single core Fortran models will be discussed. Then, the results of the parallel Fortran model will be discussed and compared to the single core model. Finally, the results of Breugem's modified DNS model will be discussed and compared to the other models.

4.1. Single core Fortran model

This section will discuss the results of the single core Fortran model shown in section 3.1. Firstly, the formation of structures and the aggregation on the bar magnet will be discussed. This section will be completed by discussing the rheological models.

4.1.1. Structure formation

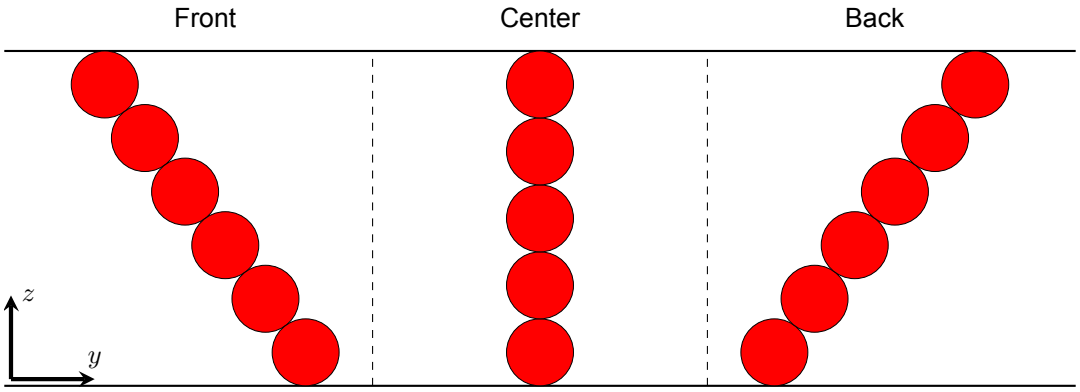
As shown in section 3.1.2, for higher shear rates the particles in the MRF form planes parallel to the direction of flow. These planes are evenly spaced over the simulation domain due to the repellent forces between the dipoles, when the poles are aligned. The formation of planes in a sheared MRF agrees with the sheets found by simulations of Lagger, Breinlinger, Korvink, *et al.*, Lagger, Bierwisch, Korvink, *et al.* and the sheets found by experiments of Bossis, Volkova, Lacis, *et al.*, Volkova, Cutillas, and Bossis [21], [28], [43], [60]. The planes are intertwined for higher shear rates when the mutual dipole method is used, because the particles are induced by other particles as well, as discussed in section 2.1.2. The poles of the particles will therefore be less aligned, which result in fewer repellent forces between the particles.

Aggregation on the bar magnet

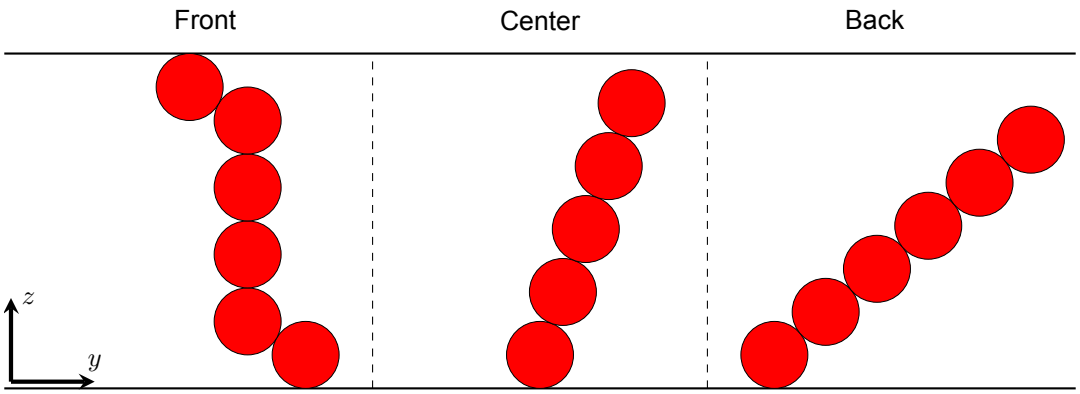
Figure 3.7 showed that the mean z-height does not decrease for the MRF in a uniform magnetic field, but does decrease in non-uniform magnetic fields. Because the particles are assumed to be neutrally buoyant, the particles do not sediment in the uniform magnetic field. In the non-uniform magnetic fields where $\dot{\gamma} = 0\,\mathrm{s}^{-1}$, the decrease of the mean z-height is caused by the attractive force of the magnet pulling the particles down. The results show that the gradient force by itself only result in minute increases of the volume fraction at the bottom wall, as shown in figure 3.8. The decrease of the volume fraction at the top wall shows that the particles are moving towards the magnet, which is more apparent at the front and back of the magnet, because the gradient in the field is larger there.

For the non-uniform fields where the shear rate $\dot{\gamma} \neq 0\,\mathrm{s}^{-1}$, the sedimentation behavior of the particles in the MRF changes. This is caused by the orientation of the particle chains with respect to the fluid flow direction. At the front of the magnet, the particle chains are oriented into the flow direction. At the center of the magnet, the particle chains are vertically placed. And at the back of the magnet, the particle chains are oriented with the flow direction, as shown in figure 3.5. Applying a shear rate will cause the chains to bend clockwise, as seen in figure 3.6, thereby changing the mean z height and the density distribution of the particles in the MRF. At the front of the magnet, the rotation of the particle chains will cause particles of the structure to accumulate at the top and bottom walls. The mean z-height of the particles will not change much when the chains are rotated. At the center and back of the magnet, the bending of the chains will cause the mean z-height to decline. Furthermore, the density at the top wall

will decrease and at the bottom wall the density will increase. This effect is more apparent at the back of the magnet, because the particle chains are orientated in the direction of the fluid flow. A schematic overview of the particle chains orientations and rotations is given in figure 4.1. The differences for the model that uses the mutual dipole method are larger due to the increase of the magnetic interaction forces between the particles. The larger interaction forces will result in sturdier chains that will shear at higher shear rates. The chains will thus stay intact for higher shear rates. For the fixed dipole method, more particle in the chains shear of the structures, thereby decreasing the differences in volume fraction.



(a) $\dot{\gamma} = 0\,\mathrm{s}^{-1}$



(b) $\dot{\gamma} > 0\,\mathrm{s}^{-1}$

Figure 4.1: Deformation of the chain-like particle structures when a shear rate is applied

4.1.2. Rheological models

The rheological models of figure 3.10 show a decline in the rate of increase of the dynamic yield stress versus the shear rate curves. The decline indicates that the MRF exhibits shear thinning behavior. This behavior is caused by the formation of planes at shear rates higher than $2500\,\mathrm{s}^{-1}$. The structures that are formed do not vary much at the higher shear rates. Therefore, the internal shear stresses caused by the interaction between the particles will be almost constant when $\dot{\gamma} > 2500\,\mathrm{s}^{-1}$, resulting in shear thinning. This effect is more apparent when the mutual dipole method is used. The internal stresses are larger and thus have a bigger impact on the total shear stress.

In the center of the magnet, the magnetic field strength is larger than at the edges. This results in larger magnetic interaction forces between the particles and thus the dynamic yield stresses are larger. At the front and the back of the magnet, the magnetic field strength is the same, but the magnetic field is mirrored. Therefore, as stated before, the particles at the front of the magnet are oriented into the fluid flow, which results in larger dynamic yield stresses. This corresponds to the findings of Xuan, Li, Li, *et al.* [6].

4.2. Parallel Fortran model using OpenMPI

This section will discuss the results of the parallel Fortran model shown in section 3.2. The formation of structures and the aggregation on the bar magnet will be discussed first. At the end of this section the constructed rheological models will be discussed.

4.2.1. Formation of planes

Just as discussed in section 4.1.1, the particles form planes at higher shear rates parallel to the direction of flow. However, the single core Fortran model showed the formation of evenly spaced tightly packed planes with empty spaces between them. The parallel model shows the formation of sheets that are still somewhat intertwined, which is caused by the larger domain that is simulated and the larger number of particles. There are more opportunities for the particles to intertwine the planes. For the non-uniform fields, the planes only form on top of the magnet. Because these planes do thus not form over the whole simulation domain, the changes in the volume fraction in x-direction are not as large as in the uniform field.

4.2.2. Aggregation on the magnet

The spikes in the volume fraction and the mean relative velocity in y-direction indicate that the particles slow down at the back of the magnet. The spike only shows that particles amass at the back of the magnet. It does not indicate whether particle are actually sticking to the magnet. Due to the dimensions of the simulation domain, the magnetic gradient in x-direction is approximately zero. Therefore, the only differences between the magnets are their remanence and the dimension in the y-direction. The results show that spikes of the volume fraction in the y-direction are largest for the first bar magnet and smaller for the other two bar magnets. The difference between the first and second bar magnet is the remanence, thus can be concluded that the accumulation of particles at the back of the magnet depends on the remanence of a bar magnet. A larger remanence results in larger magnetic field strengths, which results in larger interaction forces between the particles. The increase of this force results in stronger chains that are able to withstand more shear stress, which results in bigger clusters of particles.

The results of the volume fraction over the z-height from figure 3.16 only shows minute differences between the uniform field and the non-uniform field when there is no shear applied. From this can be concluded that the bar magnet only pulls the particles slightly towards the bottom wall. Just as was shown in the single core Fortran models, the gradient force by itself does not result in particles aggregating on the magnet and forming textures. At larger shear rates, the volume fraction in z-height increases at the bottom wall, which shows that particles have moved towards the magnet. It was concluded that the gradient force by itself does not result in sedimentation of particles at the bottom wall. This must be caused by interaction between the tilting of the chains and the magnetic gradient force, because the particles do move to the bottom wall at higher shear rates. The sedimentation of the particles due to the gradient field does not depend on the bar magnets, as the volume fraction in z-direction is approximately the same for all magnets.

Figure 3.18 showed increases of the density at the center of the film height in a uniform magnetic field when a shear rate is applied. These increases are caused by the clockwise rotation of the particle chains.

The decreases of the density in front and after the bar magnets that was shown in figure 3.19, are a result of the magnetic gradient force acting on the particles. The gradient next to these areas is larger, which results in migration of the particles towards the magnet.

The differences in the density of the yz-plane of figure 3.20, show increases at the bottom wall at the back of the magnet for lower shear rates. As was shown by the single core Fortran model, particles at the back of the magnet bend towards the magnet, thereby increasing the density at the bottom wall. At the front of the magnet, the particle aggregate on the walls due to the rotation of the chains. At larger shear rates, the particles aggregate where the gradient of the magnetic field is largest. The particles build up in front of this aggregation, resulting in increases of density at the back of the magnet.

Just as in the single core Fortran model, the particles do not sediment in the uniform field due to the assumption that the density of the fluid and particles are equal. The results of the mean z-height of figure 3.21 show that applying a shear rate results in larger decreases. Furthermore, it can be concluded that the reduction of mean z-height depends on the remanence and the dimension of the magnet in the direction of flow. A larger remanence and a larger dimension in the direction of flow result in larger

decreases.

4.2.3. Increased t^{final}

The results of the simulations with an increased t^{final} show that the particles will amass at the front of the magnet and at the back of the magnet, when a shear rate is applied. The formation of aggregates at the back of the magnet was also found by Lucieer [15]. The particles have moved from the small magnetic field strength at the edges of the simulation domain to the larger magnetic field strength closer to the magnet. It can be concluded that the structures in the MRF have not fully formed yet for the simulations with the smaller t^{final}, as the planes in these simulations are less intertwined.

It should be noted that the aggregation of particles at the back of the magnet was only found using the parallel Fortran model. In the single core Fortran model only a small part of the domain has been simulated. Therefore, simulating a larger domain of the magnetic field is required to show that particles amass at the discussed locations. The single core Fortran model showed that the aggregations are larger when the more accurate mutual dipole method is used. The aggregations formed in the parallel model will therefore be larger in reality than the results currently show.

4.2.4. Rheological models

The rheological model of an MRF in a uniform magnetic field, found using the parallel Fortran model, is the same for the single core Fortran model where the fixed dipole method is used. The models are thus comparable. Shear thinning is less apparent due to the use of the fixed dipole method, as discussed in section 4.1.2.

The rheological models for the MRF in a non-uniform field show that the dynamic yield stress in the MRF depends on the remanence and the y-dimension of the bar magnet. The dynamic yield stresses of the second bar magnet are only slightly smaller than the ones of the first bar magnet. The only difference between these magnets is their remanence. It can thus be concluded that the remanence only has a small influence on the shear stress of the MRF. The third bar magnet shows a larger drop in the dynamic yield stresses. The second bar magnet has the same remanence as the third bar magnet, but the latter is smaller in y-direction. As discussed in section 4.1.2, the magnetic field strength at the center of the magnet is larger than at the edges, which results in an increase of the dynamic yield stress. Reducing the y-dimension of the magnet will reduce the overall area where the magnetic field is comparable to the center of the magnet. The single core Fortran model showed that the dynamic yield stress at the center of the magnet is the largest. A reduction of this area will thus result in smaller shear stresses. It can be conclude that the viscoplastic models used to approximate the behavior of the MRF can still be used, because the rheological models follow the same trend.

4.3. Breugem's DNS code with magnetic particles

Just as in the other models the particles in the MRF form chains along the magnetic field lines that are rotated due to the applied shear rate. In this model, the particles do not to form sheets of tightly packed particles. Especially for higher shear rates, the particles do not form planes.

The shear stress response of this model showed that the static and dynamic yield stress are reached at a later time in the simulation than for the other models. Therefore, the steady state is postponed. This is a result of the interaction between the particles and the fluid. The lubrication forces that are present between particles that are moving towards each other, result in damping of the collisions. The damping between all the collisions in the simulation results in postponing the formation of structures and therefore steady state will be reached at a later time. The shear stress response shows that the MRF has not been simulated for the same t^{final} as the other models for all the shear rates. The simulations of the DNS model took 24 hours, therefore it was not feasible to simulate the MRF for the same t^{final}. This does result in differences of the dynamic yield stress and the steady state structures, because steady state is not fully reached yet. The dynamic yield stresses are estimated using the shear stress where the MRF has almost reached steady state. This does however result in some inaccuracies for the constructed rheological model. The rheological models of all the models that use the fixed dipole methods seem to agree. The differences in dynamic yield stresses at low shear rates are a result of the MRF not fully reaching steady state. From the rheological models can be concluded that the two-way coupling and the included slip between the particles and the walls does not affect the rheological behavior of the MRF in steady state.

5

Conclusion

This chapter will discuss the conclusions of this thesis. First, conclusions are made for the literature review. Then, the conclusion of the verification of the model with literature will be described. Finally, the conclusions for the formations of textures when the MRF is subjected to a non-uniform magnetic field and the conclusions for the rheological models will be given.

5.1. Conclusions from literature review

A review of literature, given in: *The transition from structures to textures in magnetorheological fluids: A literature review*, showed that there are only a few theories and models about the formation of textures in an MRF subjected to a non-uniform field. A model that can be used to research the behavior of an MRF in a non-uniform field was constructed by using existing modeling techniques for particulate flows and MRFs. The main conclusions of the literature review, that were used to construct the models, will be discussed.

- Currently, there are only models that can simulate a small domain of an MRF. Literature shows that there is only one model that currently simulates the behavior of the MRF in a non-uniform field [6]. This model was not used to research sedimentation of particles in a non-uniform magnetic field and does not show any signs of aggregation on the magnet. A model that is able to simulate a larger domain is required to research to aggregation of particles on a permanent magnet.

Flow

- Magnetorheological fluids are mostly used in applications with low Reynolds numbers, therefore the fluid force acting on the particles can be determined using Stokes' drag law. In literature there are existing discrete element models that take two-way coupling between the particles and the fluid into account. It is however not known whether this impacts the steady state behavior of the MRF [6], [17]–[21], [23], [28], [32]–[34], [38]–[41].

Particle forces

- The contact between particles in the MRF is often described by a simplified Hertzian contact force. This model does not take slip into account which might affect the behavior of the MRF [6], [17]–[26].
- Several methods exist to determine the magnetic interaction force between two particles. It is not known how these methods affect the MRF in a non-uniform magnetic field, where the particles aggregate on the magnet [6], [17], [18], [21]–[23], [28]–[37].

Magnetic fields

- The magnetic field produced by a bar magnet can be approximated by integrating the dipole magnetic moment over the volume of the bar magnet [48]–[50].

Shear stress

- Literature shows that the shear stress in an MRF depends on the forces between the particles at low Mason numbers [6], [35], [36], [51], [52]. The viscous shear stress only contributes to the total shear stress at higher Mason numbers [21], [53].

5.2. Verification of the models

The constructed Fortran models have been verified using existing literature and the modified particulate flow model of Breugem [2].

Literature

- The models are used to simulate the MRF with parameters used in literature, as shown in **??**. These simulations are in agreement with the mutual dipole method. The models using the fixed dipole method use the same principles to simulate the MRF. Therefore, these models can be used to approximate the behavior of the MRF. However, there are some differences in the formation of structures and the dynamic yield stresses.

Breugem's model with magnetic particles

- Breugem's parallel DNS model showed that the rheological models of the Fortran models and the modified DNS model agree. From this can be concluded that two-way coupling and slip does not influence the steady state behavior of the MRF. Two-way coupling between the particles and the fluid delays when the MRF reaches steady state. Furthermore, it is an extra verification of the Fortran models.

5.3. Aggregation of the particles

- The models showed that the magnetic gradient force acting on the particles by itself does not result in the formation of aggregation. The aggregation of particles at the back of the magnet depends on the remanence of the magnet and the applied shear rate. The aggregates at the back of the magnet get larger with increasing shear rates.
- At the front of the magnet, particles aggregate at the walls due to the rotation of the chains when a shear rate is applied to the MRF

5.4. Rheological models

- The models showed that the dynamic yield stresses of the MRF in a non-uniform field mainly depend on the length of the magnet in the direction of flow. Increasing the dimension will increase the area where a large magnetic field strength is present, thus increasing the dynamic yield stress.
- The parallel Fortran model showed that the behavior of the MRF in a non-uniform field can still be approximated using a viscoplastic model for the simulated shear rates and magnets.

5.5. Recommendations

The following recommendations are made:

- Currently the MRF has only been simulated for a small variety of Mason numbers. Therefore, the results can only be compared to literature in a small extent. By simulating the MRF over a larger range of Mason numbers, a more extensive rheological model can be constructed that could be compared to literature. This way, it can be determined whether the models are accurate for the other Mason numbers as well. Furthermore, this would allow to fit the rheological models in the non-uniform magnetic field with one of the viscoplastic models that are used to model MRFs.
- As shown by the single core Fortran model, the use of the mutual dipole method affects the dynamic yield stresses and steady state structure of the fluid. The parallel Fortran model currently uses the fixed dipole method to simulate the MRF. The mutual dipole method would affect the aggregation as the interaction forces are larger. Currently, the mutual dipole method would require communication between all process to determine the correct magnetic moment of the particles. This greatly increases the run times of the models. It should however be possible to incorporate this in such a way that communication between all the processes is unnecessary, which would greatly decrease the run time. An idea to include this method is given in appendix H.
- The parallel Fortran model is able to determine the steady state structure of the MRF. However, using the slab volume profiles, it is hard to say if particles are actually sticking to the magnet or just slowing down at the back of the magnet. The density plots give a better representation of the formation of aggregates in the simulation domain, but can not be used to determine the height of the aggregates that are formed at the back of the magnet. A way to better determine the density of particles in the simulation domain might be used to compute the dimensions of the textures. These dimensions could then be used in continuum models of an MRF. It should however be noted that the accumulation of particles on the magnet has not reached a steady state yet. In order to determine the dimensions of the texture, the models should be simulated until the textures have fully formed. An idea to determine the dimensions of aggregations is given in appendix H.
- In all the constructed models, the particles are assumed to be of the same size. In reality the particles are not of the same size, which might affect the behavior of the MRF. The models should be modified to simulate particles of different sizes.
- The models make the assumption that particles are neutrally buoyant due to the methods that manufacturers use to prevent sedimentation of the particles. These methods do however affect the MR effect and thus should be taken into account.

References

[1] *Home - bohai: A trusted magnetorheological technology expert*, 2022. [Online]. Available: https://bohai.co/.

[2] W. P. Breugem, "A combined soft-sphere collision / Immersed Boundary Method for resolved simulations of particulate flows," *American Society of Mechanical Engineers, Fluids Engineering Division (Publication) FEDSM*, vol. 1, no. PARTS A, B AND C, pp. 2381–2392, 2010, ISSN: 08888116. DOI: 10.1115/FEDSM-ICNMM2010-30634.

[3] J. R. Morillas and J. De Vicente, "Magnetorheology: a review," *Soft Matter*, vol. 16, no. 42, pp. 9614–9642, 2020, ISSN: 17446848. DOI: 10.1039/d0sm01082k.

[4] S. Thiagarajan and A. S. Koh, "Performance and Stability of Magnetorheological Fluids—A Detailed Review of the State of the Art," *Advanced Engineering Materials*, vol. 2001458, pp. 1–24, 2021, ISSN: 15272648. DOI: 10.1002/adem.202001458.

[5] J. D. Carlson and M. R. Jolly, "MR fluid, foam and elastomer devices," *Mechatronics*, vol. 10, no. 4-5, pp. 555–569, 2000. DOI: https://doi.org/10.1016/S0957-4158(99)00064-1.

[6] T. Xuan, J. Li, B. Li, and W. Fan, "Effects of the non-uniform magnetic field on the shear stress and the microstructure of magnetorheological fluid," *Journal of Magnetism and Magnetic Materials*, vol. 535, no. April, p. 168 066, 2021, ISSN: 03048853. DOI: 10.1016/j.jmmm.2021.168066. [Online]. Available: https://doi.org/10.1016/j.jmmm.2021.168066.

[7] W. Chen, G. Zhang, X. He, S. Ren, and P. Huang, "Developing of the experimental device that test magneto rheological fluid (MRF) shear yield stress nnder non uniform composite field," *Key Engineering Materials*, vol. 667, pp. 385–390, 2016, ISSN: 16629795. DOI: 10.4028/www.scientific.net/KEM.667.385.

[8] D. X. Phu and S. B. Choi, "Magnetorheological fluid based devices reported in 2013–2018: Mini-review and comment on structural configurations," *Frontiers in Materials*, vol. 6, 2019, ISSN: 22968016. DOI: 10.3389/fmats.2019.00019.

[9] A. G. Olabi and A. Grunwald, "Design and application of magneto-rheological fluid," *Materials and Design*, vol. 28, no. 10, pp. 2658–2664, 2007, ISSN: 18734197. DOI: 10.1016/j.matdes.2006.10.009.

[10] H. Lu, D. Hua, B. Wang, *et al.*, "The Roles of Magnetorheological Fluid in Modern Precision Machining Field: A Review," *Frontiers in Materials*, vol. 8, no. May, pp. 1–11, 2021, ISSN: 22968016. DOI: 10.3389/fmats.2021.678882.

[11] N. Tala-Ighil, M. Fillon, and P. Maspeyrot, "Effect of textured area on the performances of a hydrodynamic journal bearing," *Tribology International*, vol. 44, no. 3, pp. 211–219, 2011, ISSN: 0301679X. DOI: 10.1016/j.triboint.2010.10.003. [Online]. Available: http://dx.doi.org/10.1016/j.triboint.2010.10.003.

[12] N. Singh and R. K. Awasthi, "Influence of texture geometries on the performance parameters of hydrodynamic journal bearing," *Proceedings of the Institution of Mechanical Engineers, Part J: Journal of Engineering Tribology*, vol. 235, no. 10, pp. 2056–2072, 2021, ISSN: 2041305X. DOI: 10.1177/1350650120982691.

[13] M. C. de Graaf, "A new concept in bearing technology: Magnetorheological texturing (master thesis)," no. January, p. 86, 2019.

[14] G. T. D. van Der Meer, "Modelling self-healing magnetic textures in hydrodynamic bearings," *Thesis Topology*, no. December, 2020.

[15] B. T. D. Lucieer, "Experimental investigation of self-healing magnetorheological surface textures," no. December, 2020.

[16] E. Gabriel, G. E. Fagg, G. Bosilca, *et al.*, "Open MPI: Goals, concept, and design of a next generation MPI implementation," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 3241, pp. 97–104, 2004, ISSN: 16113349. DOI: 10.1007/978-3-540-30218-6_19.

[17] A. Ghaffari, S. H. Hashemabadi, and M. Ashtiani, "A review on the simulation and modeling of magnetorheological fluids," *Journal of Intelligent Material Systems and Structures*, vol. 26, no. 8, pp. 881–904, 2015, ISSN: 1045-389X. DOI: 10.1177/1045389X14546650. [Online]. Available: http://journals.sagepub.com/doi/10.1177/1045389X14546650.

[18] T. Leps and C. Hartzell, "High fidelity, discrete element method simulation of magnetorheological fluids using accurate particle size distributions in LIGGGHTS extended with mutual dipole method," *Materials Research Express*, vol. 8, no. 8, p. 085 701, 2021. DOI: 10.1088/2053-1591/ac113c.

[19] K. J. Son, "A discrete element model for the influence of surfactants on sedimentation characteristics of magnetorheological fluids," *Korea Australia Rheology Journal*, vol. 30, no. 1, pp. 29–39, 2018, ISSN: 20937660. DOI: 10.1007/s13367-018-0004-z.

[20] M. Kroupa, M. Vonka, M. Soos, and J. Kosek, "Utilizing the Discrete Element Method for the Modeling of Viscosity in Concentrated Suspensions," *Langmuir*, vol. 32, no. 33, pp. 8451–8460, 2016, ISSN: 0743-7463. DOI: 10.1021/acs.langmuir.6b02335. [Online]. Available: https://pubs.acs.org/doi/10.1021/acs.langmuir.6b02335.

[21] H. G. Lagger, T. Breinlinger, J. G. Korvink, *et al.*, "Influence of hydrodynamic drag model on shear stress in the simulation of magnetorheological fluids," *Journal of Non-Newtonian Fluid Mechanics*, vol. 218, pp. 16–26, 2015, ISSN: 03770257. DOI: 10.1016/j.jnnfm.2015.01.010. [Online]. Available: http://dx.doi.org/10.1016/j.jnnfm.2015.01.010.

[22] K. Han, Y. T. Feng, and D. R. Owen, "Modelling of magnetorheological fluids with combined lattice Boltzmann and discrete element approach," *Communications in Computational Physics*, vol. 7, no. 5, pp. 1095–1117, 2010, ISSN: 19917120. DOI: 10.4208/cicp.2009.09.130.

[23] K. Han, Y. T. Feng□, and D.R.J.Owen, "Three-dimensional modelling and simulation of magnetorheological fluids," *International*, no. May, pp. 1102–1119, 2010. DOI: 10.1002/nme. [Online]. Available: http://onlinelibrary.wiley.com/doi/10.1002/nme.3279/full.

[24] H. P. Zhu, Z. Y. Zhou, R. Y. Yang, and A. B. Yu, "Discrete particle simulation of particulate systems: Theoretical developments," *Chemical Engineering Science*, vol. 62, no. 13, pp. 3378–3396, 2007, ISSN: 00092509. DOI: 10.1016/j.ces.2006.12.089.

[25] A. Di Renzo and F. P. Di Maio, "Comparison of contact-force models for the simulation of collisions in DEM-based granular flow codes," *Chemical Engineering Science*, vol. 59, no. 3, pp. 525–541, 2004, ISSN: 00092509. DOI: 10.1016/j.ces.2003.09.037.

[26] M. Paulick, M. Morgeneyer, and A. Kwade, "A new method for the determination of particle contact stiffness," *Granular Matter*, vol. 17, no. 1, pp. 83–93, 2015, ISSN: 1434-5021. DOI: 10.1007/s10035-014-0537-x. [Online]. Available: http://link.springer.com/10.1007/s10035-014-0537-x.

[27] J. Shäfer, S. Dippel, and D. E. Wolf, "Force Schemes in Simulations of Granular Materials," *Journal de Physique I*, vol. 6, no. 1, pp. 5–20, 1996, ISSN: 1155-4304. DOI: 10.1051/jp1:1996129.

[28] H. G. Lagger, C. Bierwisch, J. G. Korvink, and M. Moseler, "Discrete element study of viscous flow in magnetorheological fluids," *Rheologica Acta*, vol. 53, no. 5-6, pp. 417–443, 2014, ISSN: 00354511. DOI: 10.1007/s00397-014-0768-0.

[29] E. E. Keaveny and M. R. Maxey, "Modeling the magnetic interactions between paramagnetic beads in magnetorheological fluids," *Journal of Computational Physics*, vol. 227, no. 22, pp. 9554–9571, 2008, ISSN: 00219991. DOI: 10.1016/j.jcp.2008.07.008. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0021999108003677.

[30] Y. Fu, J. Yao, H. Zhao, G. Zhao, and Y. Qiu, "Simulation of a bidisperse magnetorheological fluid using the combination of a two-component lattice Boltzmann method and a discrete element approach," *Soft Matter*, vol. 15, no. 34, pp. 6867–6877, 2019, ISSN: 17446848. DOI: 10.1039/c9sm01408j.

[31] A. Sand, J. F. Stener, M. O. Toivakka, J. E. Carlson, and B. I. Pålsson, “A Stokesian dynamics approach for simulation of magnetic particle suspensions,” *Minerals Engineering*, vol. 90, pp. 70–76, 2016, ISSN: 08926875. DOI: 10.1016/j.mineng.2015.10.015.

[32] C. Ciocanel, G. Lipscomb, and N. G. Naganathan, “A particle pair model for magnetorheological fluids,” *Smart Structures and Materials 2006: Active Materials: Behavior and Mechanics*, vol. 6170, 61700S, 2006, ISSN: 0277786X. DOI: 10.1117/12.658743.

[33] K. Manikas, G. G. Vogiatzis, M. Hütter, and P. D. Anderson, “Structure formation in suspensions under uniform electric or magnetic field,” *Multiscale and Multidisciplinary Modeling, Experiments and Design*, vol. 4, no. 2, pp. 77–97, 2021, ISSN: 25208179. DOI: 10.1007/s41939-021-00091-9. [Online]. Available: https://doi.org/10.1007/s41939-021-00091-9.

[34] M. Kargulewicz, I. Iordanoff, V. Marrero, and J. Tichy, “Modeling of magnetorheological fluids by the discrete element method,” *Journal of Tribology*, vol. 134, no. 3, pp. 1–9, 2012, ISSN: 07424787. DOI: 10.1115/1.4006021.

[35] J. C. Fernández-Toledano, J. A. Ruiz-López, R. Hidalgo-Álvarez, and J. de Vicente, “Simulations of polydisperse magnetorheological fluids: A structural and kinetic investigation,” *Journal of Rheology*, vol. 59, no. 2, pp. 475–498, 2015, ISSN: 0148-6055. DOI: 10.1122/1.4906544.

[36] J. A. Ruiz-López, Z. W. Wang, J. C. Fernández-Toledano, R. Hidalgo-Alvarez, and J. de Vicente, “Start-up rheometry of highly polydisperse magnetorheological fluids: experiments and simulations,” *Rheologica Acta*, vol. 55, no. 3, pp. 245–256, 2016, ISSN: 00354511. DOI: 10.1007/s00397-016-0914-y.

[37] D. Du, F. Toffoletto, and S. L. Biswal, “Numerical calculation of interaction forces between paramagnetic colloids in two-dimensional systems,” *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 89, no. 4, pp. 1–8, 2014, ISSN: 15502376. DOI: 10.1103/PhysRevE.89.043306.

[38] D. Ji, Y. Luo, H. Ren, D. Wei, and J. Shao, “Numerical Simulation and Experimental Analysis of Microstructure of Magnetorheological Fluid,” *Journal of Nanomaterials*, vol. 2019, 2019, ISSN: 16874129. DOI: 10.1155/2019/6312606.

[39] K. Shahrivar, E. Carreón-González, J. R. Morillas, and J. De Vicente, “Aggregation kinetics of carbonyl iron based magnetic suspensions in 2D,” *Soft Matter*, vol. 13, no. 14, pp. 2677–2685, 2017, ISSN: 17446848. DOI: 10.1039/c7sm00075h.

[40] C. P. Reynolds, K. E. Klop, F. A. Lavergne, S. M. Morrow, D. G. Aarts, and R. P. Dullens, “Deterministic aggregation kinetics of superparamagnetic colloidal particles,” *Journal of Chemical Physics*, vol. 143, no. 21, 2015, ISSN: 10897690. DOI: 10.1063/1.4936323.

[41] H. T. Banks, K. Ito, M. R. Jolly, H. V. Ly, F. L. Reitich, and T. M. Simon, “Dynamic simulations and nonlinear homogenization study for magnetorheological fluids,” *Smart Structures and Materials 1999: Mathematics and Control in Smart Structures*, vol. 3667, no. March, p. 92, 1999, ISSN: 0277786X. DOI: 10.1117/12.350134.

[42] J. De Vicente, D. J. Klingenberg, and R. Hidalgo-Alvarez, “Magnetorheological fluids: A review,” *Soft Matter*, vol. 7, no. 8, pp. 3701–3710, 2011, ISSN: 1744683X. DOI: 10.1039/c0sm01221a.

[43] G. Bossis, O. Volkova, S. Lacis, and A. Meunier, “Magnetorheology: Fluids, Structures and Rheology,” no. June 2014, pp. 202–230, 2002. DOI: 10.1007/3-540-45646-5_11.

[44] C. L. Berli and J. De Vicente, “A structural viscosity model for magnetorheology,” *Applied Physics Letters*, vol. 101, no. 2, 2012, ISSN: 00036951. DOI: 10.1063/1.4734504.

[45] Y. Zhang, D. Li, and Z. Zhang, “The study of magnetorheological fluids sedimentation behaviors based on volume fraction of magnetic particles and the mass fraction of surfactants,” *Materials Research Express*, vol. 6, no. 12, 2019, ISSN: 20531591. DOI: 10.1088/2053-1591/ab5ed8.

[46] S. Kumar, R. Sehgal, M. F. Wani, and M. D. Sharma, “Stabilization and tribological properties of magnetorheological (MR) fluids: A review,” *Journal of Magnetism and Magnetic Materials*, vol. 538, no. July, p. 168 295, 2021, ISSN: 03048853. DOI: 10.1016/j.jmmm.2021.168295. [Online]. Available: https://doi.org/10.1016/j.jmmm.2021.168295.

[47] M. N. Aruna, M. R. Rahman, S. Joladarashi, and H. Kumar, “Influence of additives on the synthesis of carbonyl iron suspension on rheological and sedimentation properties of magnetorheological (MR) fluids,” *Materials Research Express*, vol. 6, no. 8, 2019, ISSN: 20531591. DOI: 10.1088/2053-1591/ab1e03.

[48] Z. J. Yang, T. H. Johansen, H. Bratsberg, G. Helgesen, and A. T. Skjeltorp, “Potential and force between a magnet and a bulk Y1Ba 2Cu3O7-δ superconductor studied by a mechanical pendulum,” *Superconductor Science and Technology*, vol. 3, no. 12, pp. 591–597, 1990, ISSN: 09532048. DOI: 10.1088/0953-2048/3/12/004.

[49] Z. J. Yang, “Magnetic Fields Around Permanent Magnets,” in Oslo: University of Oslo, 1990. [Online]. Available: https://inis.iaea.org/collection/NCLCollectionStore/_Public/22/029/22029781.pdf.

[50] J. M. Camacho and V. Sosa, “Alternative method to calculate the magnetic field of permanent magnets with azimuthal symmetry,” *Revista Mexicana de Fisica E*, vol. 59, no. 1, pp. 8–17, 2013, ISSN: 18703542.

[51] J. A. Ruiz-López, Z. W. Wang, R. Hidalgo-Alvarez, and J. de Vicente, “Simulations of model magnetorheological fluids in squeeze flow mode,” *Journal of Rheology*, vol. 61, no. 5, pp. 871–881, 2017, ISSN: 0148-6055. DOI: 10.1122/1.4990641.

[52] J. Zhou, J. Mo, C. Shao, and Z. Li, “Effects of magnetized walls on the particle structure and the yield stress of magnetorheological fluids,” *Journal of Magnetism and Magnetic Materials*, vol. 389, pp. 124–129, 2015, ISSN: 03048853. DOI: 10.1016/j.jmmm.2015.03.088. [Online]. Available: http://dx.doi.org/10.1016/j.jmmm.2015.03.088.

[53] I. M. Krieger, “Rheology of monodisperse latices,” *Advances in Colloid and Interface Science*, vol. 3, no. 2, pp. 111–136, 1972, ISSN: 00018686. DOI: 10.1016/0001-8686(72)80001-0.

[54] M. P. Allen and D. J. Tildesley, *Computer Simulation of Liquids*. 2nd ed. Oxford: Oxford University Press, 2017, pp. 100–106, ISBN: 978–0–19–880319–5.

[55] L. Pei, H. Pang, K. Chen, S. Xuan, and X. Gong, “Simulation of the optimal diameter and wall thickness of hollow Fe3O4 microspheres in magnetorheological fluids,” *Soft Matter*, vol. 14, no. 24, pp. 5080–5091, 2018, ISSN: 17446848. DOI: 10.1039/c8sm00704g.

[56] R. D. Groot and P. B. Warren, “Dissipative particle dynamics: Bridging the gap between atomistic and mesoscopic simulation,” *Journal of Chemical Physics*, vol. 107, no. 11, pp. 4423–4435, 1997, ISSN: 00219606. DOI: 10.1063/1.474784.

[57] R. d. Fainchtein, *Introduction to mpi a tutorial with exercises*, 2001. [Online]. Available: https://edoras.sdsu.edu/~mthomas/docs/mpi/nasa.tutorial/.

[58] Delft High Performance Computing Centre (DHPC), *DelftBlue Supercomputer (Phase 1)*, https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase1, 2022.

[59] J. C. Brändle de Motta, W. P. Breugem, B. Gazanion, J. L. Estivalezes, S. Vincent, and E. Climent, “Numerical modelling of finite-size particle collisions in a viscous fluid,” *Physics of Fluids*, vol. 25, no. 8, 2013, ISSN: 10706631. DOI: 10.1063/1.4817382.

[60] O. Volkova, S. Cutillas, and G. Bossis, “Shear banded flows and nematic-to-isotropic transition in er and mr fluids,” *Physical Review Letters*, vol. 82, no. 1, pp. 233–236, 1999, ISSN: 10797114. DOI: 10.1103/PhysRevLett.82.233.

[61] O. I. Vinogradova, “Drainage of a Thin Liquid Film Confined between Hydrophobic Surfaces,” *Langmuir*, vol. 11, no. 6, pp. 2213–2220, 1995, ISSN: 15205827. DOI: 10.1021/la00006a059.

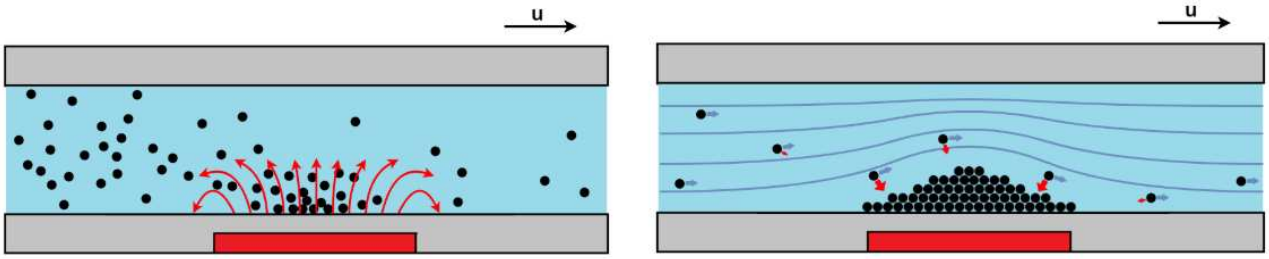
[62] T. B. Jones, *Electromechanics of Particles*. 1995, ISBN: 9780521431965. DOI: 10.1017/cbo9780511574498.

[63] S. Meskos, L. Kedward, A. Markus, O. Certik, and M. Curcic, *The fortran programming language*, 2021. [Online]. Available: https://fortran-lang.org/en/.

The transition from structures to textures in magnetorheological fluids:
A literature review

Thobias van Kuik (4544986)

October 29, 2022



Contents

1 Fundamentals of MRFs **2**
1.1 Operating principle . . . 2
1.2 Operational modes . . . 4
1.3 Defects . . . 4
1.3.1 Hard cake . . . 4
1.3.2 Clumping . . . 4
1.3.3 Fluid particle separation (FPS) . . . 5
1.3.4 Oxidation of particles . . . 5
1.3.5 Sedimentation . . . 5
1.4 Experimental techniques . . . 7
1.4.1 Optical microscopy . . . 7
1.4.2 Scattering dichroism . . . 7
1.4.3 Dynamic light scattering . . . 8
1.4.4 Digital micro-holographic particle image/tracking velocimetry . . . 9

2 Modeling of MRFs **10**
2.1 Discrete modeling . . . 10
2.1.1 Interaction between particle and the fluid . . . 10
2.1.2 Lubrication . . . 11
2.1.3 Contact force between particles . . . 11
2.1.4 Magnetic force between particles . . . 11
2.2 Continuum modeling . . . 14
2.2.1 Rheological models . . . 14
2.2.2 Structural models . . . 16

1 | Fundamentals of MRFs

To get an insight into magnetorheological fluids, the fundamental of MRFs are discussed. This chapter will go over the operating principle of MRFs, the operational modes in which MRFs are used, the defects of MRFs and finally the experimental techniques used to examine the behavior of MRFs.

1.1 Operating principle

Magnetorheological fluids generally behave as normal suspensions in the absence of a magnetic field. Applying a magnetic field results in a magnetic dipole and multipole moment on each particle. These moments bring about magnetic forces that act between the particles encouraging the alignment of the moments. Due to this phenomenon, chain-like structures are formed within the MRF [18, 19]. These structures resist the flow, which changes the properties of the fluid making the MRF behave in a non-Newtonian way [20].

In order to form these structures the MRF needs to overcome Brownian motion. The coupling parameter λ defines the ratio between the magnetic and thermal energies shown in 1.1 [21, 22].

$$\lambda = \frac{\pi \mu_o \mu_c \beta^2 a^3 H_0^2}{2\kappa_B T} \tag{1.1}$$

where μ_0 is the vacuum permeability, μ_c is relative permeability of the continuous phase , $\beta = \frac{\mu_p - \mu_c}{\mu_p + 2\mu_c}$, μ_p is the relative permeability of the particles, a is the particle radius, H_0 is the magnetic field strength, κ_B is the Boltzmann constant and T is the temperature.

Different kind of structures can be formed. These so-called aggregation schemes depend on the coupling parameter λ and the particle volume fraction ϕ. Figure 1.1 gives an overview of the structures that could form within a MRF. The equilibrium structure is same the structure of the MRF without the application of a magnetic field [23]. Tip-to-tip structures are aggregations of particles that form in long-thin chains, shown in figure 1.2a. These structures are the most common in MRFs and are often found in literature [24, 25]. An increase of the particle volume fraction causes these chains to intersect and form thick chains. When the particle volume increases even further, all chains connect and a fibrous structure appears as shown in figure 1.2b [21].

The kinetics of the aggregation in a 2D system depend on the following two length scales:

- The characteristic inter particle distance, R_0, in the absence of a magnetic field given by the particle volume fraction: $R_\phi = \sqrt{\pi}\frac{d}{2}\phi^{-1/2}$.
- The distance, R_m, where the magnetostatic interaction energy of a pair of particles equals the thermal fluctuations: $R_\mathrm{m} = d\lambda^{1/3}$.

The ratio of these length scales determines whether the process of aggregation is limited by diffusion ($\frac{R_\mathrm{m}}{R_\phi} < 1$) or whether it is of a deterministic nature ($\frac{R_\mathrm{m}}{R_\phi} > 1$) [22, 24].

In a flow, the particles are forced apart by hydrodynamic forces. The Mason number (Mn) defines the ratio of the hydrodynamic forces and the approximation of the dipolar force magnitude on the particles, as shown in equation (1.2). Furthermore, the Mason number can be used to determine whether chains are being formed or broken [1, 26].

$$\mathrm{Mn} = \frac{F^\mathrm{f}}{F_0} = \frac{9}{2}\frac{\eta_c \dot{\gamma} \phi^2}{\mu_0 \mu_c M^2} \tag{1.2}$$

where η_c the viscosity of the continuous phase, $\dot{\gamma}$ is the macroscopic shear rate, ϕ is the particle volume fraction and M is the suspension magnetization [27]. It should be noted that these parameters are based on a simplification of the magnetostatic, were multipole and multiple particle interaction is neglected. From

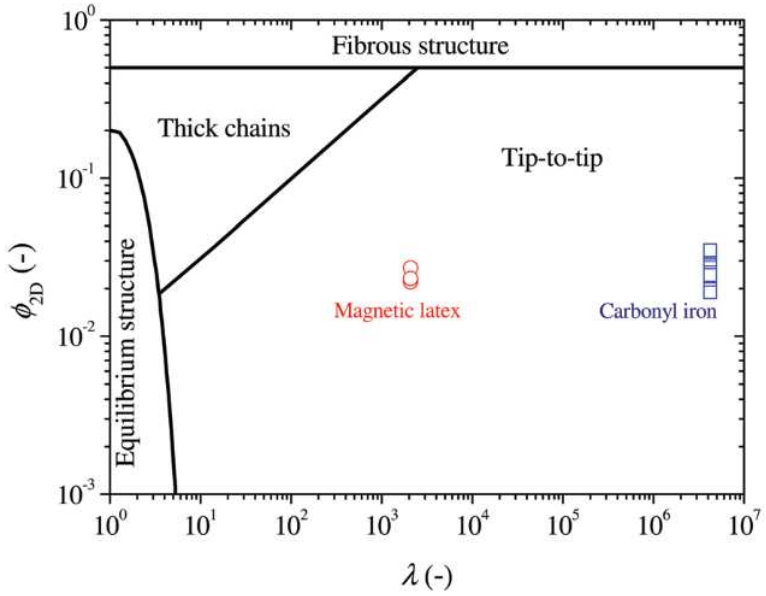
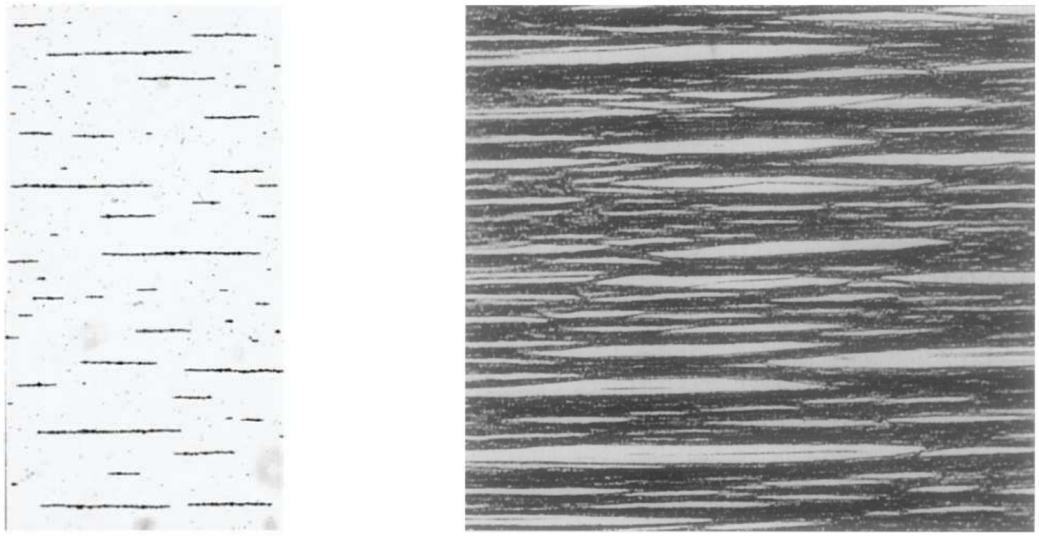


Figure 1.1: Aggregation regimes of a MRF. Taken from [22].



(a) Tip-to-tip structure. Taken from [22]

(b) Fibrous structure. Taken from [21]

Figure 1.2: Structures of a MRF

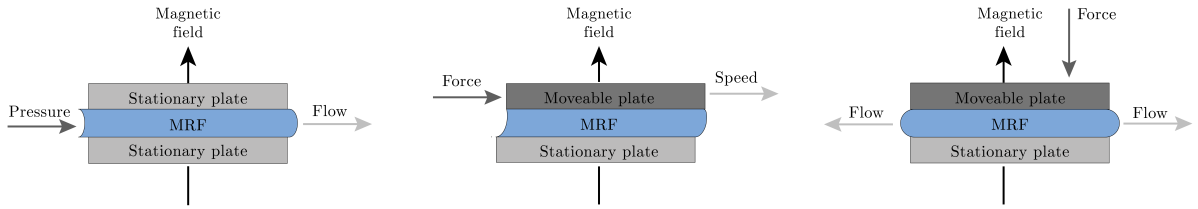
these assumptions follows that the rheological behavior of MRFs are dictated by the Mason number (Mn), the coupling parameter λ and the particle volume fraction ϕ.

The structures formed due to the MR effect are characterized by their yield stress. The yield stress is typically divided into two parts, the static yield stress and the dynamic yield stress. The static yield stress is the minimum stress required to make the MRF flow. The dynamic yield stress is the stress at which the structures are completely broken due to continuous shearing [1, 2, 26]. The dynamic yield stress is usually estimated using viscoplastic models, which will be further discussed in section 2.2.1.

The field dependent yield stress is expressed by the relative MR effect. This is defined as the ratio between the stress increment when a magnetic field is applied and the stress value in the absence of a magnetic field. Therefore, this effect depends on the on-state and off-state properties of the material. The on-state properties are the permeability of the particle and the particle volume fraction, the off-state properties are the viscous dissipation within the carrier [1].

1.2 Operational modes

Magnetorheological fluids are commonly used in a shearing flow or an elongational flow. The shearing flows can be distinguished into a drag flow (shear mode) or capillary flow (valve mode) and are the most common modes. MRFs used in dampers and shock absorbers operate in the valve mode. In this mode, the MRF is pushed through a narrow channel. The shear mode is a result of two plates moving relative to each other. This modes can be found in clutches and brakes. The elongational flow is also known as the squeeze mode, where MRF is squeezed by a force in the same direction as the applied magnetic field. MRFs are commonly used in shear and valve mode. An overview of the modes is shown in figure 1.3 [1, 7, 28].



(a) Valve mode (b) Shear mode (c) Squeeze mode

Figure 1.3: Operational modes

1.3 Defects

There are several defects that might come up when using MRFs. This section will provide an overview of these defect and the techniques used to minimize them.

1.3.1 Hard cake

Due to the agglomeration of particles and remnants of magnetization, hard cakes can be formed during operation. Due to the remnants of magnetization of the particles, these hard cakes remain even after the external magnetic field is turned off. It is hard to break these agglomerates apart due to the high forces between each particle. A higher shear rate is required to make the MRF flow due to this process. Several techniques are used to counter this defect. Coating the particles can delay the settling time and facilitate re-dispersion [29, 30].

1.3.2 Clumping

Very high magnetic fields that are applied for a long time causes the particles to be trapped in their structures. Therefore, the carrier fluid is able to flow freely. Applying a force to the particles that are stuck in place, results in the formation of clusters. This effect is called clumping and can only take place when high magnetic fields are applied, while particles can agglomerate at low magnetic fields [29, 30, 31]. The process is shown in figure 1.4.

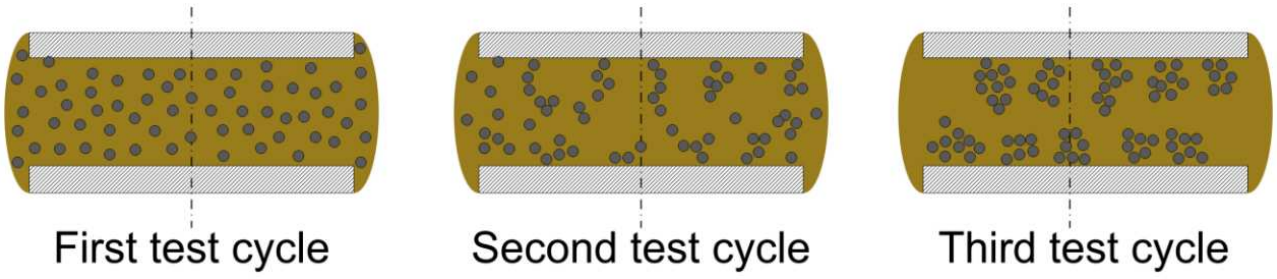


Figure 1.4: Clumping process troughout three test cycles. Taken from [32].

1.3.3 Fluid particle separation (FPS)

Fluid particle separation only occurs in squeeze mode. This defect causes the particles to be separate from the fluid which changes the chain structures. FPS only occurs at low viscosity, low compression speed and high magnetic field strengths. Therefore, FPS can be prevented by having a sufficiently high viscosity [29, 30, 33].

1.3.4 Oxidation of particles

The particles used in MRFs are mainly made of iron types or steel types. When iron based particles are exposed to the atmosphere, oxidation will occur. This leads to a performance drop in the MRF due to a decrease in the ability to magnetize the oxidized particles. Furthermore, the off-state viscosity is increased due to the increase in solid volume. Oxidation can be prevented by coating the particles, thereby reducing exposure [29, 30].

1.3.5 Sedimentation

Due to the difference in density of the continuous phase and the particle, sedimentation occurs. This section discusses the techniques used to prevent sedimentation, the physics of sedimentation of MRFs and experimental techniques to determine this behavior.

Stabilization techniques

A few stabilization techniques are used to improve the stability of an MRF and reduce sedimentation. The stability of the fluid can be improved by coating the particles, adding nanoparticles to the fluid, changing the carrier fluid and adding additives. These techniques will be discussed in the following sections.

Particle coating A common method to reduce the sedimentation problem is to coat the particles in a organic polymer. This polymer is used to reduce the difference in density, thereby stabilizing the MRF. Furthermore, it prevents aggregation of particles and protects the core material from external degradation effects. However, depending on the coating, the MR effect and the off-state viscosity might be altered. The effects of the coating are experimentally determined [2, 34, 35].

Nanoparticles Adding solid nanoparticles to the MRF can be used to lessen the sedimentation of the magnetic particles. Nanoparticles with a relatively low density are used. This lower density and the larger specific surface area due to the smaller size, can prevent sedimentation. The MR-effect will be changed due to the addition of these nanoparticles, because they tend to break up the aggregates and promote the formation of anisotropic structures. The nanoparticles added distribute between the gaps of the micro particles. This increases the yield strength of the structures and therefore enhances the MR-effect [34]. Furthermore, the stability is improved due to the Brownian motion that is generated by the nanoparticles in proximity to the magnetic particles. Using ferrofluid in a MRF increases the yield stress of the suspension fourfold. This is caused by the contact between the particles and the carrier fluid due to the higher magnetic permeability of the particles [35].

Carrier fluid Another way to reduce sedimentation within a MRF is by changing the carrier fluid. Increasing the density of the carrier fluid results in less sedimentation. However, this does increase the viscosity of the fluid in the absence of a magnetic field, which is unwanted. Silicon oil is still used in most MRFs and other uses of carrier fluid have not been found suitable for industrial use [2, 34].

Additives Finally, it is possible to add additives to the MRF in order to stabilize the fluid and improve the magnetostatic force between particles [35]. The stability is improved due to the additives occupying the empty spaces between particles in the MRF and due to the high surface area and high aspect ratio of these additives [36]. The additives can also be used to improve the chemical stability of the fluid [2].

Sedimentation behavior

Most currently available studies are focused on ways to improve the stability of MRFs, without taking physics into account. At low volume concentration, Stokes' law can be used to approximate the settling behavior of a MRF. However, for large concentrations of particles, Stokes' law does not hold [37].

Different settling models which take mechanical interaction between particles into account can be used to determine the settling behavior of MRFs. The empirical Steinour model [38], shown in equation (1.3), gives a insight into the settling behavior of MRFs depending on the particle volume fraction. The Batchelor model [39] is an analytical model that can be used for very dilute monodisperse suspensions at low Reynolds numbers [40], this model is shown in equation (1.4). The drawback of this model is that it under-predicts the settling velocity.

$$v(\phi) = \left[v_0 (1-\phi)^2\right] \left(10^{-1.82\phi}\right) \tag{1.3}$$

$$v(\phi) = v_0 (1 - k\phi) \tag{1.4}$$

where v_0 is the Stokes velocity and k is an experimentally determined parameter.

Other models can be used to take the aggregation behavior of MRFs into account. The Vesilind model [41], given in equation (1.5), is an empirical model that determines the average settling velocity in flocculated suspensions. The parameters $v_{\max}$ is determined experimentally. This model is the most common model to predict settling behavior [40].

The Dick model [42], given in equation (1.6), is a semi-empirical model that determines the average settling velocity of flocculated suspensions. The parameters $v_{\max}$ and k are determined experimentally. This model is widely used, however it proves to be less accurate than the Vesilind model [40].

The Richardson & Zaki model [43], given in equation (1.7), is an empirical hydrodynamic model that can be used for monodisperse suspensions at any Reynolds number. The parameter n is determined depending on the Reynolds number and the ratio of the particle diameter, d_s, to the vessel diameter, D, which is given in equation (1.8) [43]. It should be noted that these models cannot be used to predict the sedimentation behavior of MRFs to which additives or nanoparticles are added.

$$|v_s(\phi)| = v_{\max} e^{-k\phi} \tag{1.5}$$

$$|v_s(\phi)| = v_{\max} \phi^k \tag{1.6}$$

$$|v_s(\phi)| = v_{\max} (1-\phi)^n \tag{1.7}$$

$$n = \begin{cases} 4.65 + 19.5\frac{d_s}{D}, & \mathrm{Re} < 0.2 \\ \left(4.35 + 17.5\frac{d_s}{D}\right) \mathrm{Re}^{-0.03}, & 0.2 < \mathrm{Re} < 1 \\ \left(4.45 + 18\frac{d_s}{D}\right) \mathrm{Re}^{-0.1}, & 1 < \mathrm{Re} < 200 \\ 4.45\mathrm{Re}^{-0.1}, & 200 < \mathrm{Re} < 500 \\ 2.39, & \mathrm{Re} > 500 \end{cases} \tag{1.8}$$

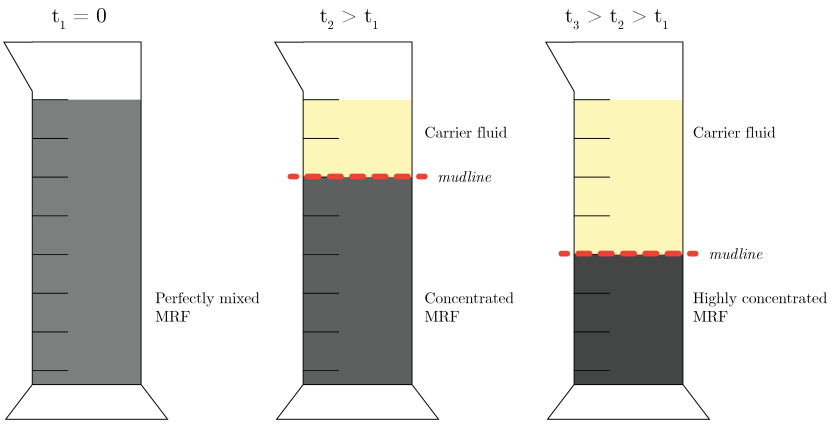


Figure 1.5: Sedimentation experiment

Sedimentation experiment

The previous section showed that the sedimentation behavior of MRF of which the stability has been improved can not always be determined using the conventional models. Therefore, experimental set-ups are often used. Commonly this behavior is examined using direct observational methods. This is achieved by putting a MRF into a graduated cylinder and measuring the height of the mudline at a certain time interval. This mudline indicates where the particles and carrier fluid have separated, illustrated by figure 1.5. Using this method, the sedimentation ratio of the MRF can be determined [1, 36, 37, 44]. Other methods exist, such as the magnetic inductance monitoring techniques. These techniques use the mismatch in permeability of the particles and the fluid and the dependency of a MRF on its particle loading to determine the sedimentation behavior [1].

1.4 Experimental techniques

This section focuses on the experimental methods used to understand the formation of structures in MRFs. Optical methods are most frequently used within this research area. These optical methods can only be used for suspensions through which light can pass. These methods can not be used to identify the structural behavior of MRFs when the particle volume fraction is too high. The most common experimental techniques are discussed below.

1.4.1 Optical microscopy

Optical microscopy is a way to study the structure evolution of MRFs in 2D [45]. An optical microscope and camera are used to capture this evolution. The images can be examined in order to identify the behavior of the MRF. Some images captured using this method are shown in figure 1.6.

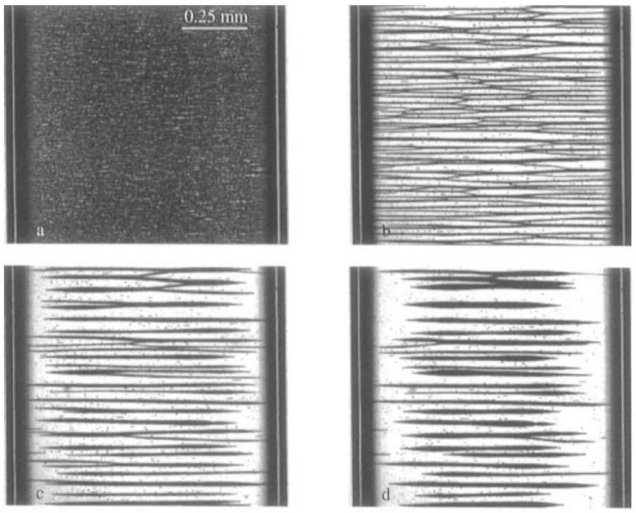


Figure 1.6: Images of MRF structures at different time intervals. Taken from [45].

1.4.2 Scattering dichroism

Scattering dichroism can be used to capture the structural evolution of an MRF in a rotating magnetic field. A monochromatic light is circularly polarized using a polarizer, modulator and a quarter wave plate. The light then passes through the MRF sample and is detected by a photodiode. The signal from the diode is amplified and processed. The resulting signal consists of two intensities with different frequencies. Using these intensities and the signal of the original light, the dichroism can be determined. This dichroism is related to the number of particles forming aggregates per unit volume [46].

1.4.3 Dynamic light scattering

The dynamics and kinetics of chain formation can also be examined using dynamic light scattering. A laser beam is passed through a MRF sample. The collision of the particles in the MRF and the beam cause a scattering of the light which can be detected, shown in figure 1.7. The signal from the detector is then processed to determine the intensity of the scatter. More fluctuations in the intensity, means the aggregates are small. The fluctuations of the intensity can be converted into a correlation function, from which the size of the aggregates can be determined [47, 48]. An illustration of this is shown in figure 1.8.

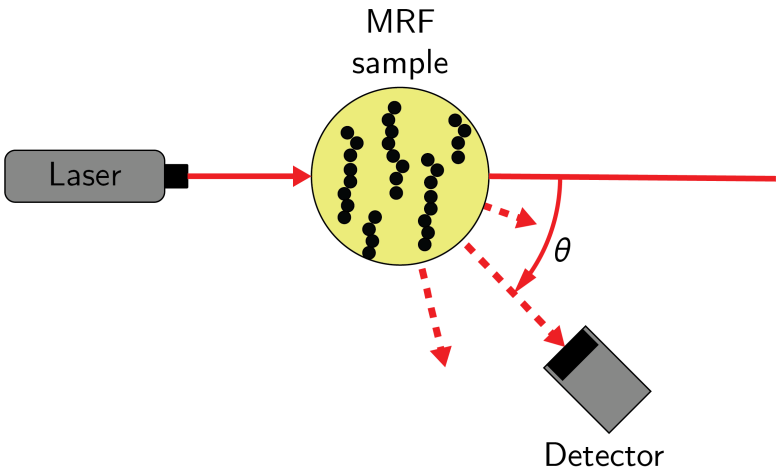
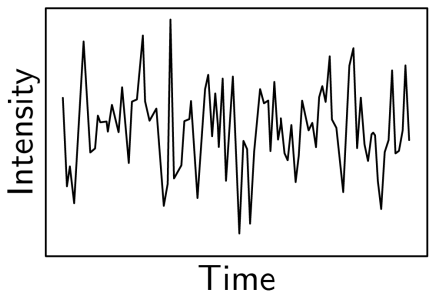
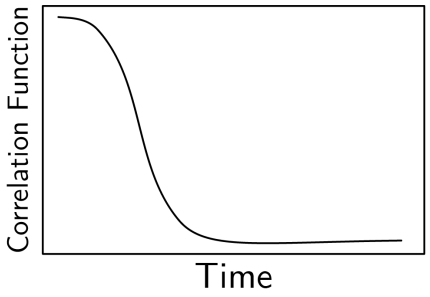


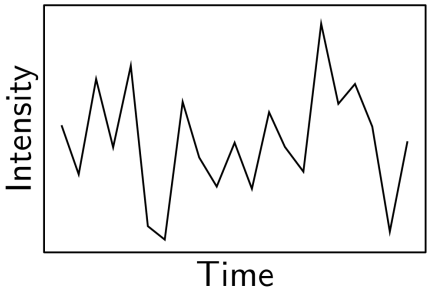
Figure 1.7: Schematic of the dynamic light scattering technique



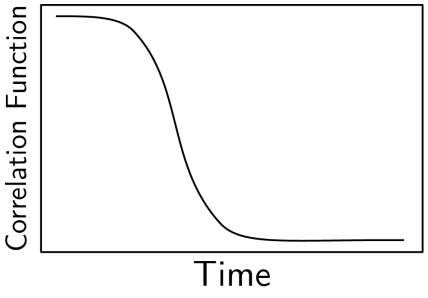
(a) Intensity signal for small aggregates



(b) Correlation function for small aggregates



(c) Intensity signal for large aggregates

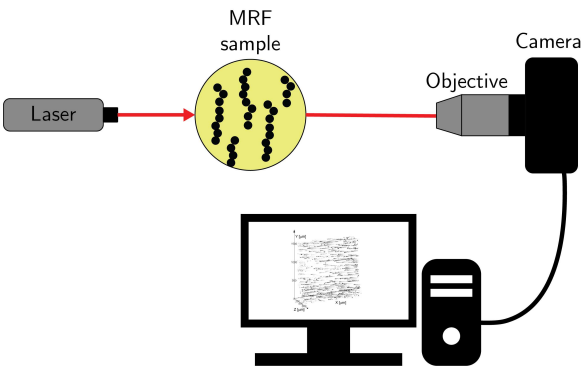


(d) Correlation function for large aggregates

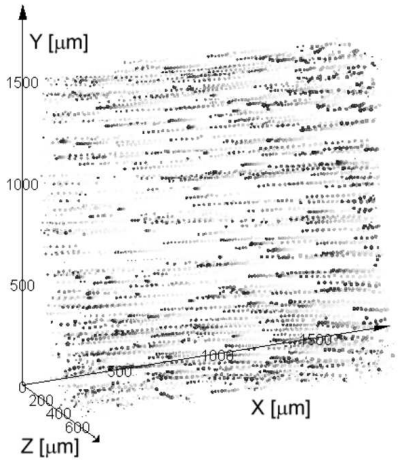
Figure 1.8: Dynamic light scattering signal processing

1.4.4 Digital micro-holographic particle image/tracking velocimetry

Digital micro-holographic particle image/tracking velocimetry is a common technique used to examine the dynamic properties of a flow. This technique visualizes a MRF in 2D [22, 49, 50], 3D [50] and even 4D [51, 52]. Images of a sample through which a laser beam is passed through, are captured with a camera. These images are then analyzed using different algorithms to identify particles or clusters of particles and the dynamics of the particles. A schematic illustration of this technique and a digitally constructed 3D model are shown in figure 1.9a and figure 1.9b respectively.



(a) Schematic of the digital micro-holographic particle image/tracking techniques



(b) 3D model of an MRF constructed using digital micro-holographic particle image/tracking techniques. Taken from [52].

Figure 1.9: Digital micro-holographic particle image/tracking velocimetry

2 | Modeling of MRFs

The magnetorheological fluid can be modeled in order to examine and predict its behavior. Two main methods exist in literature. The discrete method models the fluid and particles separately. This method is mainly used to examine the particle dynamics and structure formation within the MRF. The continuum approach provides a faster way to determine the behavior of MRF, however only the macroscopic properties can be determined. The following sections will discuss both methods of modeling MRFs in more detail.

2.1 Discrete modeling

In a discrete model the continuous phase (carrier fluid), the discrete phase (particles), and the magnetic field are considered separately. Basic physical laws are used to model the three phases and their interactions. Numerical solving techniques are used to solve this kind of simulation. The main problem with the discrete approach is the high computational cost. However, because the models are based on basic physical laws, there are few assumptions and therefore the method will be more accurate than the continuum approach [20].

The behavior of suspensions can be modeled using Newton's equations of motion for the particles and the Navier-stokes equations for the fluid. Next to this, boundary and initial conditions, the interaction between the continuous and discrete phase, and the interaction between particles should be taken into account.

The equation of motion of a particle i with a mass m_i is shown in equation (2.1) with the initial condition

$$m_i \frac{\mathrm{d}^2 r_i}{\mathrm{d}t^2} = \sum_j F_{ij}^{\mathrm{c}} + \sum_k F_{ik}^{\mathrm{nc}} + F_i^{\mathrm{f}} + F_i^{\mathrm{ext}} \tag{2.1}$$

where F_{ij}^{c} is the contact force acting from particle j on particle i, F_{ik}^{nc} the non-contact force acting from particle k on particle i, F_i^{f} represents the particle-fluid interaction and F_i^{ext} are the forces acting on particle i due to external fields (gravity, magnetic, electric). It has been experimentally determined that the magnetic forces dominate the Brownian forces and are therefore negligible [26].

2.1.1 Interaction between particle and the fluid

Several forces act on a particle due to the interaction between the continuous and discrete phase. These forces consist of the drag force, the pressure gradient force, the virtual mass force, the Basset force and lift forces such as the Magnus and Saffman forces [53]. The pressure gradient force is a result of a difference in pressure across a surface. The virtual mass force results from the force required to accelerate the fluid surrounding a particle. The Basset force is a result of the lagging boundary layer development with changing relative velocity. In the modeling of MRFs, only the more significant drag force is taken into account [20]. Due to the low Reynolds numbers in which MRFs are commonly used, Stokes law is approximated by equation (2.2) [4, 15, 16, 17, 54, 55, 56, 57, 58, 59, 60, 61].

$$F^{\mathrm{drag}} = 6\pi\eta_c a(v^{\infty}(\mathbf{r}_i) - v_i) \tag{2.2}$$

where $v^{\infty}(\mathbf{r}_i)$ is the velocity of the continuous phase at $\mathbf{r}_i$ and v_i is the particle's velocity.

2.1.2 Lubrication

When two particles or a particle and the wall get close together in a viscous fluid, a hydrodynamic force is exerted on the particles due to lubrication. An analytical expression of this force is given in equation (2.3). Corrections are applied to this expression in order to take non-normal collision or rotation of particles into account. The amplification factors are given in equations (2.4) and (2.5).

$$F^{\text{lub}}(\epsilon, u_i) = -6\pi\eta_f a u_i \left[\lambda(\epsilon) - \lambda(\epsilon_{\text{al}})\right] \tag{2.3}$$

$$\lambda_{\text{pp}} = \frac{1}{2\epsilon} - \frac{9}{20}\ln(\epsilon) - \frac{3}{56}\epsilon\ln(\epsilon) + 1.346 \quad + \mathcal{O}(\epsilon) \tag{2.4}$$

$$\lambda_{\text{pw}} = \frac{1}{\epsilon} \; - \; \frac{1}{5}\ln(\epsilon) \; - \frac{1}{21}\epsilon\ln(\epsilon) + 0.9713 + \mathcal{O}(\epsilon) \tag{2.5}$$

where u_i is the particle velocity, ϵ is the ratio between the distance separating the surfaces and the particle radius, ϵ_{al} is the distance for which lubrication should be taken into account, λ_{pp} is the amplification factor for particle-particle lubrication and λ_{pw} is the amplification factor for particle-wall lubrication [62, 63].

2.1.3 Contact force between particles

Due to the deformation of particles, the contact between particles is on a finite area. The forces acting on this area are composed of normal and tangential forces. The total force and torque on a particle is related to geometrical factors, physical factors, material properties and movement of the particles. Due to these factors a general and accurate description of the forces acting on the particle is hard to come by. Therefore simplified methods are used.

The linear spring-dashpot model is a commonly used linear model. In this model a spring is used to describe the elastic deformation and a dashpot accounts for viscous dissipation. The Hertz-Mindlin and Deresiewics model provides a more complex and accurate model. This model is based on the non-linear relationship between the normal force and normal displacement of an elastic contact between two spheres. The model shows that the relation between the force and the displacement is subject to the loading history and the instantaneous rate of change of the forces or displacements. Because of the complexity of this model, simplifications have been made. The most commonly used simplification of the model uses a direct force-displacement relation for the tangential force and the Hertz theory for the normal force. This simplification has been widely used for the study of dynamic behavior of granular matter [53, 63, 64].

Often, in the modeling of MRFs an exponential repulsion law, or a simplified Hertzian model is used to model this force between particles. It has been shown that both models are equally suited for modeling the repulsive force in MRFs [56].

The simplified Hertzian model only makes use of the normal component of the model. This normal force is given in equation (2.6).

$$F_{ij}^{\text{con}} = \left(\frac{\frac{2}{3}E}{1-\nu^2}\sqrt{R_{\text{eff}}}\xi^{3/2}\right)\frac{\mathbf{r}_{ij}}{r_{ij}} \tag{2.6}$$

where E is Young's modulus, ν is Poisson's ratio, $R_{\text{eff}} = R_i R_j / (R_i + R_j)$ is the effective particle radius and $\xi = \max\{R_i + R_j - r_{ij}, 0\}$ is the overlap between the particles i and j [4, 15, 54, 55, 56, 58, 65].

2.1.4 Magnetic force between particles

When a external magnetic field is applied to an MRF, each magnetizable particle can be induced with a magnetic moment. Each magnetized particle will then generate a magnetic field due to this magnetic moment. This results in a force that will act on each particle due to the magnetic field from each neighboring particle. There are a few methods to determine these magnetic interaction forces, which will be explained in detail below.

Accurate magnetic model

The general model gives an exact answer to the magnetic interaction forces between particles. The model is based on the simplified Maxwell equations with no free electric currents in the medium domain, subject to boundary conditions. The Maxwell equations are shown in equations (2.7) and (2.8) .

$$\nabla \times \mathbf{H} = 0 \tag{2.7}$$
$$\nabla \cdot \mathbf{B} = 0 \tag{2.8}$$

where $\mathbf{H}$ is the magnetic field intensity and $\mathbf{B}$ is the flux density of the medium. It is assumed that the medium is linear isotropic with a permeability μ, which results in the constitutive equation (2.9).

$$\mathbf{B} = \mu \mathbf{H} \tag{2.9}$$

Equation (2.7) allows for a scalar potentials such that $\mathbf{H} = -\nabla \phi_{\text{out}}$ for the fluid phase and $\mathbf{H} = -\nabla \phi_{\text{in}}^{i}$ for the inside of a particle i. Solving equations (2.7) to (2.9) results in the Laplace equation for the both phases, shown in equations (2.10) and (2.11).

$$\nabla^2 \phi_{\text{out}} = 0 \tag{2.10}$$
$$\nabla^2 \phi_{\text{in}}^{(i)} = 0 \tag{2.11}$$

Equations (2.10) and (2.11) can then be solved by adding the appropriate boundary conditions [14, 58].

For N particles in a uniform magnetic field in a domain Ω_i, the general solution for equations (2.10) and (2.11) is given by equations (2.12) and (2.13).

$$\phi_p^i(\mathbf{r}_i) = \psi_i + \phi_0(\mathbf{r}_i) + \sum_{l(\geq 0),m} \beta_{lm}^i r_i^l Y_{lm}(\theta_i, \varphi_i) \tag{2.12}$$

$$\begin{aligned} \phi_f(\mathbf{r}_i) = \psi_i + \phi_0(\mathbf{r}_i) &+ \sum_{l(\geq 0),m} \frac{1}{2l+1} Q_{lm}^i r_i^{-(l+1)} Y_{lm}(\theta_i, \varphi_i) \\ &+ \sum_{j \neq i} \sum_{l(\geq 0),m} \frac{1}{2l+1} Q_{lm}^j \beta_{lm}^i r_j^{-(l+1)} Y_{lm}(\theta_i, \varphi_i) \end{aligned} \tag{2.13}$$

where $\mathbf{r}_i$ is the position vector related to the center of particle i; $\phi_0(\mathbf{r}_i)$ is the potential due to the external field; $\psi_i = \phi_0(\mathbf{x}_i)$ is the external potential at the center of particle i; $Y_{lm}(\theta_i, \varphi_i)$ give the spherical harmonics defined to the center of particle i; Q_{lm}^i are the multi-poles moment and the coefficients β_{lm}^i are unknown [20, 58].

Equations (2.12) and (2.13) show that this model takes multiple particle interactions and multi-poles effects into account. The equations can be solved by determining the unknown coefficients and the multipoles based on the boundary conditions of the particles. The main limitation of this model is the high computational cost. Due to the large number of particles, this model is difficult to apply to MRF [20]. In order to simplify the model, a fast multipole method (FMM) can be applied. This method uses multipole expansions to determine the far-field contribution to a slowly decaying potential [60, 61, 66].

Fixed dipole model

To simplify the problem, the particles are considered to be point dipoles and the multiple particle interaction is ignored. The point dipole are thus only induced by the external field applied to the MRF. This causes each particle to acquire a magnetic dipole moment $\mathbf{m}$ shown in equation (2.14) [14, 15, 16, 17, 20, 56, 57, 58, 67].

$$\mathbf{m} = V_p \chi_e \mathbf{H}_0 \tag{2.14}$$

where $V_p = \frac{4\pi a^3}{3}$ is the particle volume, $\chi_e = \frac{3(\chi-1)}{\chi+2}$ the effective susceptibility, $\chi = \frac{\mu_p}{\mu_c}$ the relative susceptibility.

From the magnetic dipole moment of a particle $\mathbf{m}^{(i)}$, the magnetic field produced by this particle can be determined using equation (2.15).

$$\mathbf{H}^{\text{dip}}(\mathbf{r}_i) = \frac{1}{4\pi} \left(\frac{3\mathbf{m}^{(i)} \cdot \mathbf{r}_i}{r_i^5} - \frac{\mathbf{m}^{(i)}}{r_i^3} \right) \tag{2.15}$$

where $\mathbf{r}_i = \mathbf{x} - \mathbf{P}^{(i)}$, $\mathbf{P}^{(i)}$ is the origin of particle i, $\mathbf{x}$ is a point in space and $r_i = |\mathbf{r}_i|$.

The force acting on a point dipole due to the magnetic field from all remaining dipoles is then given by equation (2.16).

$$\mathbf{F}_i^{\text{mag}} = \mu_0 \nabla_{\mathbf{P}^{(i)}} \left(\mathbf{m}_i \cdot \sum_{j=1, j\neq i}^{N} \mathbf{H}^{\text{dip}} \left(\mathbf{P}^{(i)} - \mathbf{P}^{(j)} \right) \right) \tag{2.16}$$

Due to its simplicity, this model is most commonly used for simulating MRFs. However, this model becomes inaccurate when particles are close together because multiple particle interactions and multipoles are neglected [14].

Mutual dipole model

The mutual dipole model is based on the same principle as the fixed dipole, where particles are replaced by dipoles [4, 14, 15, 58]. However, in this model, multiple particle interaction is taken into account. Particles are magnetically induced by the external magnetic field and the magnetic field generated by other particles. This results in the magnetic dipole moment of a particle i given by equation (2.17).

$$\mathbf{m}^{(i)} = V_p \chi_e \left(\mathbf{H}_0 + \sum_{j=1, j\neq i}^{N} \mathbf{H}_{\text{dip}} \left(\mathbf{P}^{(i)} - \mathbf{P}^{(j)} \right) \right) \tag{2.17}$$

The force can then be determined using the same formulation as in equation (2.16). These equations need to be solved for each particle at each time step, which results in high computational cost. This model proves to be more accurate than the fixed dipole model of section 2.1.4, however the model still shows inaccuracies when particles are in close proximity, because then the higher multipoles become important [14, 20].

The relative susceptibility, χ, influences the accuracy of the dipole models. An increase of this ratio will cause the models to be less accurate [14, 58].

In order to calculate the magnetic forces, the dipole models require an $\mathcal{O}(N^2)$ pairwise calculation. A few methods exist in order to minimize computational cost. Firstly a dipole model based on the finite distribution of the current density can make use of many existing numerical techniques [14]. Furthermore, the Gauss-Seidel algorithm of equation (2.18) can be used. This algorithm requires no more than three iterations in order to converge to a relative error of 10^{-5} [58]. Finally, a cut-off distance can be added to ignore particles beyond this distance. Because the fixed dipole is accurate at large distances, the multiple particle and multipole interaction can be ignored. When particles get within this cut off distance, the mutual dipole model can be used for a better accuracy. The relative error for taking the cut off distance as $r_c = 6a$ is less than 5%.

$$\begin{aligned} \mathbf{m}_i^{k+1} = V_p \chi_e \Bigg[& \mathbf{H}_0 + \sum_{j=1}^{i-1} \mathbf{H}_{\text{dip}} \left(\mathbf{m}_j^{k+1}, \left(\mathbf{P}^{(i)} - \mathbf{P}^{(j)} \right) \right) \\ & + \sum_{j=i+1}^{N} \mathbf{H}_{\text{dip}} \left(\mathbf{m}_j^{k}, \left(\mathbf{P}^{(i)} - \mathbf{P}^{(j)} \right) \right) \Bigg]; \\ k = 0, 1, 2, \ldots & \end{aligned} \tag{2.18}$$

Inclusion method

The discussed dipole models show that these models are inaccurate when particles are close together. This is because higher multipoles are neglected. The inclusion method aims to include these higher multipoles by implementing the accurate magnetic model from section 2.1.4 when particles get close together. It is based on a specific case of the interaction between two particles in a uniform magnetic field, where the solution is given by the sum of spherical harmonics. This specific case can be applied to multiple particles by using a corrected force method. This model is only used for particles that are sufficiently close together, outside this cut-off distance, the dipole models can be used. The performance of this model declines at high volume fractions or in rapidly rotating fields, but it is still more accurate than the dipole models [14].

2.2 Continuum modeling

Another way to model an MRF is by treating the suspension as one homogeneous phase. The behavior of this phase is modeled with the use of appropriate rheological models. This way of modeling requires less computational power than the discrete method. However, due to the dependency on rheological models, the method is less general. It is hard to find a rheological model that can approximate the behavior of MRF in a wide range of shear rates. Furthermore, this method can not be used to examine the microscopic properties of an MRF, such as the formation of structures [26].

2.2.1 Rheological models

In order to use a continuum approach to model MRFs, suitable rheological models need to be determined. Rheometers can be used to in order to determine the appropriate model. The model depends on which operational mode the fluid is used. These modes are discussed in section 1.2. A MRF starts to flow when a certain yield stress is achieved. Viscoplastic rheological models can therefore be used to determine the behavior of the MRF [20]. Different types of viscoplastic models are used in the continuum approach. The most common models will be discussed. An overview of these models is shown in figure 2.1.

The Bingham model

The Bingham model is mainly used for an MRF in medium shear rates. Due to its accuracy and simplicity, the model is often used. However, at high shear rate this model is no longer accurate, due to the shear thinning that occurs in most MRFs [68, 69]. At low shear rates the model does not take the Newtonian nature of the fluid into account [70]. The model is shown in equation (2.19).

$$\tau = \tau_y + \eta\dot{\gamma}, \quad |\tau| > \tau_y \tag{2.19}$$

where τ is the shear stress, η is the post yield fluid viscosity, τ_y is the field induced yield stress and $\dot{\gamma}$ is the shear rate.

The biviscous model

The biviscous model, shown in equation (2.20), is a model that tries to take the Newtonian nature of the fluid at low shear rates into account. However, the model does not take the initial yield stress into account [70]. As can be seen, when $\eta_r \to \infty$, the model reduces to the Bingham model [69].

$$\tau = \begin{cases} \eta_r\dot{\gamma}, & \tau \leq \tau_1 \\ \tau_y + \eta\dot{\gamma}, & \tau > \tau_1 \end{cases} \tag{2.20}$$

where η_r is the pre-yield viscosity and $\tau_1 = \frac{\tau_y}{1-\frac{\eta}{\eta_r}}$.

The biplastic Bingham model

The biviscous model states that the fluid starts flowing without a need to overcome a certain stress. However, the MRF only starts flowing after the shear stress exceeds an initial yield stress. Implementing this initial yield stress results in the model shown in equation (2.21).

$$\tau = \begin{cases} \tau_{y,\text{initial}} + \eta_r\dot{\gamma}, & \tau \leq \tau_2 \\ \tau_y + \eta\dot{\gamma}, & \tau > \tau_2 \end{cases} \tag{2.21}$$

where $\tau_{y,\text{initial}}$ is the initial yield stress and $\tau_2 = \frac{\tau_y}{1-\frac{\eta}{\eta_r}(1-\delta)}$ and $\delta = \frac{\tau_{y,\text{initial}}}{\tau_2}$.

The Herschel-Bulkley model

At higher shear rates, the Herschel-Bulkley model gives a better approximation for the behavior of MRFs than the previous models. The shear thinning that occurs at higher shear rates for most MRFs can be seen in this model [68].

$$\tau = \tau_y + \eta\left(\dot{\gamma}\right)^n, \quad |\tau| > \tau_y \tag{2.22}$$

where n is the flow index. If $n > 1$, the model takes shear thickening into account. When $n < 1$ the model accounts for shear thinning. For $n = 1$ the model is exactly the same as the Bingham model from equation (2.19) [69].

The Casson model

The Casson model is a model commonly used to study viscoplastic fluids such as mud, slurries and blood flows [71]. The model, expressed in equation (2.23), shows a more gradual transition from Newtonian behavior to non-Newtonian behavior [72].

$$\sqrt{\tau} = \sqrt{\tau_y} + \sqrt{\eta\dot{\gamma}}, \quad |\tau| > \tau_y \tag{2.23}$$

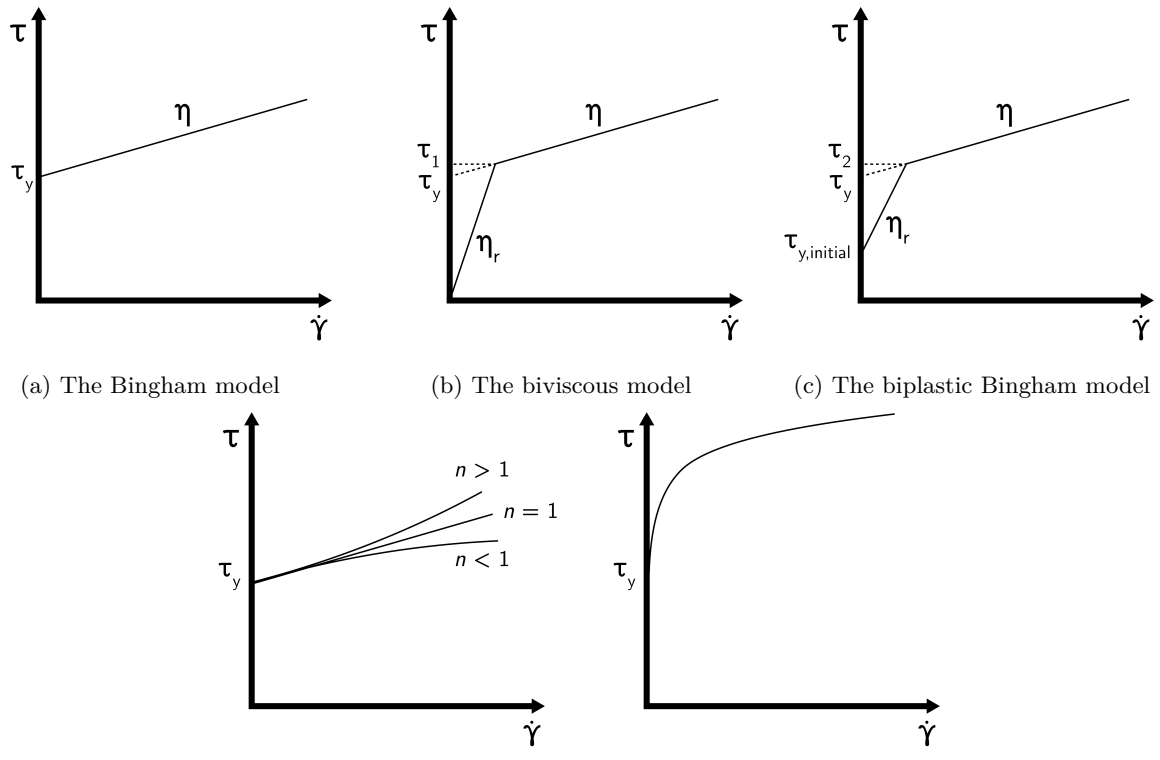
The Papanastaiou model

The Bingham, Herschel-Bulkley and Casson model all imply an infinite viscosity at zero shear rates. This means that the models are not usable in numerical approaches without modifications. One such modification is given by the Papanastaiou model [71]. This models introduces a parameter m that controls the exponential growth of stress [73]. The modification can be applied to the Bingham, Herschel-Bulkley and Casson models. The modified models are shown in equation (2.24), equation (2.25) and equation (2.26) respectively.

$$\tau = \tau_y \left[1 - e^{-m\dot{\gamma}}\right] + \eta\dot{\gamma} \tag{2.24}$$

$$\tau = \tau_y \left[1 - e^{-m\dot{\gamma}}\right] + \eta\left(\dot{\gamma}\right)^n \tag{2.25}$$

$$\sqrt{\tau} = \sqrt{\tau_y} \left[1 - e^{-m\dot{\gamma}}\right] + \sqrt{\eta\dot{\gamma}} \tag{2.26}$$



(a) The Bingham model (b) The biviscous model (c) The biplastic Bingham model

(d) The Herschel-Bulkley model (e) The Casson model

Figure 2.1: Overview of rheological models

2.2.2 Structural models

The rheological behavior of MRFs can also be determined by looking at the structure of the MRF. These structural models are based on Newton's law of motion. The dimensionless numbers used in these models are the Mason number Mn, see equation (1.2), the coupling parameter λ, see equation (1.1), and the Peclet number [20]. The Peclet number expresses the ratio of hydrodynamic to Brownian forces as shown in equation (2.27) [74].

$$\mathrm{Pe} = \frac{6\pi\eta_c\dot{\gamma}a^3}{kT} \tag{2.27}$$

These dimensionless parameters are related as follows: $\mathrm{Mn}\lambda = \frac{2\mathrm{Pe}}{3}$. This shows that only two dimensionless parameters are needed in order to determine the rheological behavior of an MRF.

Using the dimensionless numbers and the Bingham model from equation (2.19), the effective viscosity of the MRF can be determined, shown in equation (2.28). More complex models, that explain deviations in the Bingham model, exist to determine the effective viscosity of the MRF [75].

$$\frac{\eta_c}{\eta_{c,\infty}} = 1 + \left(\frac{\mathrm{Mn}}{\mathrm{Mn}^*}\right)^{-1} \tag{2.28}$$

where $\mathrm{Mn}^* = \frac{C\phi\eta_c}{\eta_{c,\infty}}$ is the critical Mason number, η_∞ is the high shear viscosity and C depends on the mechanical stability of the MRF [26].

Bibliography

[1] J. R. Morillas and J. De Vicente, "Magnetorheology: a review," *Soft Matter*, vol. 16, no. 42, pp. 9614–9642, 2020.

[2] S. Thiagarajan and A. S. Koh, "Performance and Stability of Magnetorheological FluidsA Detailed Review of the State of the Art," *Advanced Engineering Materials*, vol. 2001458, pp. 1–24, 2021.

[3] J. D. Carlson and M. R. Jolly, "MR fluid, foam and elastomer devices," *Mechatronics*, vol. 10, no. 4-5, pp. 555–569, 2000.

[4] T. Xuan, J. Li, B. Li, and W. Fan, "Effects of the non-uniform magnetic field on the shear stress and the microstructure of magnetorheological fluid," *Journal of Magnetism and Magnetic Materials*, vol. 535, no. April, p. 168066, 2021. [Online]. Available: https://doi.org/10.1016/j.jmmm.2021.168066

[5] W. Chen, G. Zhang, X. He, S. Ren, and P. Huang, "Developing of the experimental device that test magneto rheological fluid (MRF) shear yield stress nnder non uniform composite field," *Key Engineering Materials*, vol. 667, pp. 385–390, 2016.

[6] D. X. Phu and S. B. Choi, "Magnetorheological fluid based devices reported in 20132018: Mini-review and comment on structural configurations," *Frontiers in Materials*, vol. 6, 2019.

[7] A. G. Olabi and A. Grunwald, "Design and application of magneto-rheological fluid," *Materials and Design*, vol. 28, no. 10, pp. 2658–2664, 2007.

[8] H. Lu, D. Hua, B. Wang, C. Yang, A. Hnydiuk-Stefan, G. Królczyk, X. Liu, and Z. Li, "The Roles of Magnetorheological Fluid in Modern Precision Machining Field: A Review," *Frontiers in Materials*, vol. 8, no. May, pp. 1–11, 2021.

[9] N. Tala-Ighil, M. Fillon, and P. Maspeyrot, "Effect of textured area on the performances of a hydrodynamic journal bearing," *Tribology International*, vol. 44, no. 3, pp. 211–219, 2011. [Online]. Available: http://dx.doi.org/10.1016/j.triboint.2010.10.003

[10] N. Singh and R. K. Awasthi, "Influence of texture geometries on the performance parameters of hydrodynamic journal bearing," *Proceedings of the Institution of Mechanical Engineers, Part J: Journal of Engineering Tribology*, vol. 235, no. 10, pp. 2056–2072, 2021.

[11] M. C. de Graaf, "A new concept in bearing technology: Magnetorheological texturing (master thesis)," no. January, p. 86, 2019.

[12] G. T. D. van Der Meer, "Modelling self-healing magnetic textures in hydrodynamic bearings," *Thesis Topology*, no. December, 2020.

[13] B. T. D. Lucieer, "Experimental investigation of self-healing magnetorheological surface textures," no. December, 2020.

[14] E. E. Keaveny and M. R. Maxey, "Modeling the magnetic interactions between paramagnetic beads in magnetorheological fluids," *Journal of Computational Physics*, vol. 227, no. 22, pp. 9554–9571, 2008.

[15] T. Leps and C. Hartzell, "High fidelity, discrete element method simulation of magnetorheological fluids using accurate particle size distributions in LIGGGHTS extended with mutual dipole method," *Materials Research Express*, vol. 8, no. 8, p. 085701, 2021.

[16] Y. Peng and P. Pei, "Microstructure evolution based particle chain model for shear yield stress of magnetorheological fluids," *Journal of Intelligent Material Systems and Structures*, vol. 32, no. 1, pp. 49–64, 2021.

[17] D. Ji, Y. Luo, H. Ren, D. Wei, and J. Shao, "Numerical Simulation and Experimental Analysis of Microstructure of Magnetorheological Fluid," *Journal of Nanomaterials*, vol. 2019, 2019.

[18] J. M. Ginder, "Behavior of magnetorheological fluids," *MRS Bulletin*, vol. 23, no. 8, pp. 26–29, 1998.

[19] G. L. Gulley and R. Tao, "Structures of a Magnetorheological fluid," *International Journal of Modern Physics B*, vol. 15, no. 6-7, pp. 851–858, 2001.

[20] A. Ghaffari, S. H. Hashemabadi, and M. Ashtiani, "A review on the simulation and modeling of magnetorheological fluids," *Journal of Intelligent Material Systems and Structures*, vol. 26, no. 8, pp. 881–904, 2015.

[21] M. Fermigier and A. P. Gast, "Structure evolution in a paramagnetic latex suspension," *Journal of Colloid And Interface Science*, vol. 154, no. 2, pp. 522–539, 1992.

[22] K. Shahrivar, E. Carreón-González, J. R. Morillas, and J. De Vicente, "Aggregation kinetics of carbonyl iron based magnetic suspensions in 2D," *Soft Matter*, vol. 13, no. 14, pp. 2677–2685, 2017.

[23] P. Domínguez-García, J. M. Pastor, and M. A. Rubio, "Aggregation and disaggregation dynamics of sedimented and charged superparamagnetic micro-particles in water suspension," *European Physical Journal E*, vol. 34, no. 4, 2011.

[24] C. P. Reynolds, K. E. Klop, F. A. Lavergne, S. M. Morrow, D. G. Aarts, and R. P. Dullens, "Deterministic aggregation kinetics of superparamagnetic colloidal particles," *Journal of Chemical Physics*, vol. 143, no. 21, 2015.

[25] H. See and M. Doi, "Aggregation Kinetics in Electro-Rheological Fluids," pp. 2778–2782, 1991.

[26] J. De Vicente, D. J. Klingenberg, and R. Hidalgo-Alvarez, "Magnetorheological fluids: A review," *Soft Matter*, vol. 7, no. 8, pp. 3701–3710, 2011.

[27] D. J. Klingenberg, J. C. Ulicny, and M. A. Golden, "Mason numbers for magnetorheology," *Journal of Rheology*, vol. 51, no. 5, pp. 883–893, 2007.

[28] A. Farjoud, R. Cavey, M. Ahmadian, and M. Craft, "Magneto-rheological fluid behavior in squeeze mode," *Smart Materials and Structures*, vol. 18, no. 9, 2009.

[29] J. S. Kumar, P. S. Paul, G. Raghunathan, and D. G. Alex, "A review of challenges and solutions in the preparation and use of magnetorheological fluids," *International Journal of Mechanical and Materials Engineering*, vol. 14, no. 1, 2019.

[30] S. A. Wahid, I. Ismail, S. Aid, and M. S. Rahim, "Magneto-rheological defects and failures: A review," *IOP Conference Series: Materials Science and Engineering*, vol. 114, no. 1, 2016.

[31] B. Sapiński, "Observations of MR fluid clumping behaviour in a squeeze-mode damper," *Acta Mechanica et Automatica*, vol. 10, no. 3, pp. 239–245, 2016.

[32] K. Adjerid, "A Study on the Dynamic Characterization of a Tunable Magneto-Rheological Fluid- Elastic Mount in Squeeze Mode Vibration," *Thesis (Virginia Polytechnic Institute and State University)*, 2011.

[33] I. Ismail, S. A. Mazlan, H. Zamzuri, and A. G. Olabi, "Fluid-particle separation of magnetorheological fluid in squeeze mode," *Japanese Journal of Applied Physics*, vol. 51, no. 6 PART 1, 2012.

[34] M. Ashtiani, S. H. Hashemabadi, and A. Ghaffari, "A review on the magnetorheological fluid preparation and stabilization," *Journal of Magnetism and Magnetic Materials*, vol. 374, pp. 711–715, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.jmmm.2014.09.020

[35] S. Kumar, R. Sehgal, M. F. Wani, and M. D. Sharma, "Stabilization and tribological properties of magnetorheological (MR) fluids: A review," *Journal of Magnetism and Magnetic Materials*, vol. 538, no. July, p. 168295, 2021. [Online]. Available: https://doi.org/10.1016/j.jmmm.2021.168295

[36] M. N. Aruna, M. R. Rahman, S. Joladarashi, and H. Kumar, "Investigation of sedimentation, rheological, and damping force characteristics of carbonyl iron magnetorheological fluid with/without additives," *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, vol. 42, no. 5, pp. 1–13, 2020. [Online]. Available: https://doi.org/10.1007/s40430-020-02322-5

[37] Y. Zhang, D. Li, and Z. Zhang, "The study of magnetorheological fluids sedimentation behaviors based on volume fraction of magnetic particles and the mass fraction of surfactants," *Materials Research Express*, vol. 6, no. 12, 2019.

[38] H. H. Steinour, "Nonflocculated Suspensions of Uniform Spheres," *Portland Cement Association*, vol. 36, no. 7, pp. 618–624, 1944. [Online]. Available: https://pubs.acs.org/doi/pdf/10.1021/ie50415a005

[39] G. K. Batchelor, "Sedimentation in a dilute dispersion of spheres," *Journal of Fluid Mechanics*, vol. 52, no. 2, pp. 245–268, 1972.

[40] A. Zeidan, S. Rohani, A. Bassi, and P. Whiting, "Review and comparison of solids settling velocity models," *Reviews in Chemical Engineering*, vol. 19, no. 5, pp. 473–530, 2003.

[41] P. A. Vesilind, "Design of prototype thickeners from batch settling tests," *Water Sewage Works*, vol. 115, no. 7, pp. 302–307, 1968.

[42] R. I. Dick and K. Young, "Analysis of thickening performance of final settling tanks," p. 33, 1972.

[43] J. F. Richardson and W. N. Zaki, "Sedimentation and fluidisation: Part I," *Chemical Engineering Research and Design*, vol. 75, no. 1 SUPPL., pp. S82–S100, 1997. [Online]. Available: http://dx.doi.org/10.1016/S0263-8762(97)80006-8

[44] M. N. Aruna, M. R. Rahman, S. Joladarashi, and H. Kumar, "Influence of additives on the synthesis of carbonyl iron suspension on rheological and sedimentation properties of magnetorheological (MR) fluids," *Materials Research Express*, vol. 6, no. 8, 2019.

[45] J. H. Promislow and A. P. Gast, "Magnetorheological fluid structure in a pulsed magnetic field," *Langmuir*, vol. 12, no. 17, pp. 4095–4102, 1996.

[46] S. Melle, G. G. Fuller, and M. A. Rubio, "Structure and dynamics of magnetorheological fluids in rotating magnetic fields," *Physical Review E - Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics*, vol. 61, no. 4, pp. 4111–4117, 2000.

[47] J. Liu and M. Hagenbuchle, "Chain dynamics in a dilute magnetorheological fluid," *NASA Conference Publication*, no. 3338, pp. 389–394, 1996.

[48] P. M. Carvalho, M. R. Felício, N. C. Santos, S. Gonçalves, and M. M. Domingues, "Application of light scattering techniques to nanoparticle characterization and development," *Frontiers in Chemistry*, vol. 6, no. June, pp. 1–17, 2018.

[49] C. E. Willert and M. Gharib, "Digital particle image velocimetry," *Experiments in Fluids*, vol. 10, no. 4, pp. 181–193, 1991.

[50] Y. Yang, B. S. Kang, and Y. J. Choo, "Application of the correlation coefficient method for determination of the focal plane to digital particle holography," *Applied Optics*, vol. 47, no. 6, pp. 817–824, 2008.

[51] D. Ekimov, V. Kaikkonen, and A. Mäkynen, "Using digital holographic microscopy for 4D tracking of colloid particles," *Laser Applications in Life Sciences*, vol. 7376, p. 737615, 2010.

[52] D. Ekimov and A. Makynen, "In-line digital holography for high speed 4D tracking of particles," *Conference Record - IEEE Instrumentation and Measurement Technology Conference*, pp. 1434–1437, 2011.

[53] H. P. Zhu, Z. Y. Zhou, R. Y. Yang, and A. B. Yu, "Discrete particle simulation of particulate systems: Theoretical developments," *Chemical Engineering Science*, vol. 62, no. 13, pp. 3378–3396, 2007.

[54] K. J. Son, "A discrete element model for the influence of surfactants on sedimentation characteristics of magnetorheological fluids," *Korea Australia Rheology Journal*, vol. 30, no. 1, pp. 29–39, 2018.

[55] H. G. Lagger, T. Breinlinger, J. G. Korvink, M. Moseler, A. Di Renzo, F. Di Maio, and C. Bierwisch, "Influence of hydrodynamic drag model on shear stress in the simulation of magnetorheological fluids," *Journal of Non-Newtonian Fluid Mechanics*, vol. 218, pp. 16–26, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.jnnfm.2015.01.010

[56] H. G. Lagger, C. Bierwisch, J. G. Korvink, and M. Moseler, "Discrete element study of viscous flow in magnetorheological fluids," *Rheologica Acta*, vol. 53, no. 5-6, pp. 417–443, 2014.

[57] M. Kargulewicz, I. Iordanoff, V. Marrero, and J. Tichy, "Modeling of magnetorheological fluids by the discrete element method," *Journal of Tribology*, vol. 134, no. 3, pp. 1–9, 2012.

[58] K. Han, Y. T. Feng, and D.R.J.Owen, "Three-dimensional modelling and simulation of magnetorheological fluids," *International*, no. May, pp. 1102–1119, 2010. [Online]. Available: http://onlinelibrary.wiley.com/doi/10.1002/nme.3279/full

[59] C. Ciocanel, G. Lipscomb, and N. G. Naganathan, "A particle pair model for magnetorheological fluids," *Smart Structures and Materials 2006: Active Materials: Behavior and Mechanics*, vol. 6170, p. 61700S, 2006.

[60] H. V. Ly, F. Reitich, M. R. Jolly, H. T. Banks, and K. Ito, "Simulations of Particle Dynamics in Magnetorheological Fluids," *Journal of Computational Physics*, vol. 155, no. 1, pp. 160–177, 1999.

[61] H. T. Banks, K. Ito, M. R. Jolly, H. V. Ly, F. L. Reitich, and T. M. Simon, "Dynamic simulations and nonlinear homogenization study for magnetorheological fluids," *Smart Structures and Materials 1999: Mathematics and Control in Smart Structures*, vol. 3667, no. March, p. 92, 1999.

[62] J. C. Brändle de Motta, W. P. Breugem, B. Gazanion, J. L. Estivalezes, S. Vincent, and E. Climent, "Numerical modelling of finite-size particle collisions in a viscous fluid," *Physics of Fluids*, vol. 25, no. 8, 2013.

[63] W. P. Breugem, "A combined soft-sphere collision / Immersed Boundary Method for resolved simulations of particulate flows," *American Society of Mechanical Engineers, Fluids Engineering Division (Publication) FEDSM*, vol. 1, no. PARTS A, B AND C, pp. 2381–2392, 2010.

[64] P. Costa, B. J. Boersma, J. Westerweel, and W. P. Breugem, "Collision model for fully resolved simulations of flows laden with finite-size particles," *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 92, no. 5, pp. 1–14, 2015.

[65] A. Di Renzo and F. P. Di Maio, "Comparison of contact-force models for the simulation of collisions in DEM-based granular flow codes," *Chemical Engineering Science*, vol. 59, no. 3, pp. 525–541, 2004.

[66] L. F. Greengard, *The Rapid Evaluation of Potential Fields in Particle Systems*, 1988.

[67] Y. Fu, J. Yao, H. Zhao, G. Zhao, and Y. Qiu, "Simulation of a bidisperse magnetorheological fluid using the combination of a two-component lattice Boltzmann method and a discrete element approach," *Soft Matter*, vol. 15, no. 34, pp. 6867–6877, 2019.

[68] A. Farjoud, N. Vahdati, and Y. F. Fah, "Mathematical model of drum-type MR brakes using herschel-bulkley shear model," *Journal of Intelligent Material Systems and Structures*, vol. 19, no. 5, pp. 565–572, 2008.

[69] J. Goldasz and B. Sapinski, "Nondimensional characterization of flow-mode magnetorheological/ electrorheological fluid dampers," *Journal of Intelligent Material Systems and Structures*, vol. 23, no. 14, pp. 1545–1562, 2012.

[70] M. Zubieta, S. Eceolaza, M. J. Elejabarrieta, and M. M. Bou-Ali, "Magnetorheological fluids: Characterization and modeling of magnetization," *Smart Materials and Structures*, vol. 18, no. 9, 2009.

[71] W. Elsaady, S. O. Oyadiji, and A. Nasser, "A review on multi-physics numerical modelling in different applications of magnetorheological fluids," *Journal of Intelligent Material Systems and Structures*, vol. 31, no. 16, pp. 1855–1897, 2020.

[72] F. J. Adewale, A. P. Lucky, A. P. Oluwabunmi, and E. F. Boluwaji, "Selecting the most appropriate model for rheological characterization of synthetic based drilling mud," *International Journal of Applied Engineering Research*, vol. 12, no. 18, pp. 7614–7629, 2017.

[73] E. Mitsoulis, "Flows of viscoplastic materials: Models and computations," *Rheology Reviews*, pp. 135–178, 2007.

[74] G. Bossis, O. Volkova, S. Lacis, and A. Meunier, "Magnetorheology: Fluids, Structures and Rheology," no. June 2014, pp. 202–230, 2002.

[75] C. L. Berli and J. De Vicente, "A structural viscosity model for magnetorheology," *Applied Physics Letters*, vol. 101, no. 2, 2012.

The transition from structures to textures in magnetorheological fluids

T.L. van Kuik[a], R.A.J. van Ostayen[a] and G.H.G. van der Meer[a]

[a]*Delft University of Technology, Mekelweg 2, Delft, 2628 CD, The Netherlands*

ARTICLE INFO

Keywords:
magnetorheological fluid
discrete element method
surface texture
parallel modeling

ABSTRACT

Textures in hydrodynamic bearings are used to minimize the rotational velocity required to fully form a fluid film. These textures wear over time due to mechanical contact at start and stop. Magnetorheological fluids can be used to replace the textures in the bearing by self-healing textures consisting of particles. In a uniform field, the particles in the magnetorheological fluid form structures that follow the magnetic field. In a non-uniform field, particles in the magnetorheological fluid separate from the fluid and aggregate on the surface of the magnet. These particles form textures that are of self-healing nature. Currently, not much is known about the formation of these textures and the influence of their formation on the rheological behavior of the magnetorheological fluid. A parallel discrete element model that can simulate a magnetorheological fluid in non-uniform field is presented in this paper. The model uses basic physical laws to determine the dynamics of the particles. The model shows that the particles do not separate from the fluid due to the attractive force of the magnet by itself, but rather by a combination of the deformation of the particle structures and the attractive force. Furthermore, the rheological behavior of the fluid can still be approximated using viscoplastic models.

Acronyms

Acronym	Description
DHPC	Delft High Performance Computing Centre
MRF	magnetorheological fluid
OpenMPI	Open Message Passing Interface
PBC	Periodic Boundary Condition

Symbols

Symbol	Description	Unit
Latin		
a	Particle a	1
$\boldsymbol{a}$	Acceleration vector	m
$\boldsymbol{B}$	Magnetic flux density vector	T
b	Particle b	1
d	Wall d	1
E	Young's modulus	Pa
$\boldsymbol{e}$	Euclidian vector	1
$\boldsymbol{F}$	Force vector	N
H	Magnetic field strength	$\mathrm{A\,m^{-1}}$
$\boldsymbol{H}$	Magnetic field strength vector	$\mathrm{A\,m^{-1}}$
L	Length	m
M	Mass	kg
$\boldsymbol{M}$	Magnetization vector field	$\mathrm{A\,m^{-1}}$
$\boldsymbol{m}$	Magnetic moment	$\mathrm{A\,m^{2}}$
N	Number of	1
Pe	Peclet number	1
R	Particle radius	m
$\boldsymbol{r}$	Position vector	m
$\hat{\boldsymbol{r}}$	Unit position vector	1
T	Temperature	°C
t	Time	s
V	Volume	$\mathrm{kg\,m^{-3}}$
v	Scalar velocity	$\mathrm{m\,s^{-1}}$
$\boldsymbol{v}$	Velocity vector	$\mathrm{m\,s^{-1}}$
$\bar{v}$	Mean velocity	m
x	x position	m
y	y position	m
z	z position	m
Greek		
$\boldsymbol{\nabla}$	Nabla operator	1
Δt	Time step	m
β	Contrast factor	1
$\dot{\gamma}$	Shear rate	$\mathrm{s^{-1}}$
η	Dynamic viscosity	1
κ_{B}	Boltzmann constant	1
λ	Emperical parameter for the modified velocity-Verlet algorithm	1
μ	Permeability	$\mathrm{N\,A^{-2}}$
ν	Poisson"s ratio	1
ξ	Particle overlap	m
ρ	Mass density	$\mathrm{kg\,m^{-3}}$
$\boldsymbol{\sigma}$	(Maxwell) stress tensor	Pa
τ	Shear stress	Pa
φ	Volume fraction	1

Superscripts

c	Contact
dip	Dipole
eff	Effective
ext	External
f	Fluid
final	Final
fr	Relative fluid
int	Internal
mag	Magnetic
max	Maximum
p	Particle(s)
pm	Permenant magnet
pr	Relative particle
pred	Prediction
proc	Process
r	Relative
rem	Remanence
top	Top wall
visc	Viscous

Subscripts

0	Free space
a	Particle a
ab	From particle b to/on a
ad	From wall d to/on a
b	Particle b
x	x-direction
y	y-direction
z	z-direction

1. Introduction

Many mechanical systems in which a fluid is used, operate in a wide range of operating conditions. However, the fluids used are often only optimal for a small range of these conditions. A fluid that can change its rheological properties is thus desired.

Magnetorheological fluids (MRFs) are fluids that change their rheological properties in the presence of a magnetic field. Magnetorheological fluids are suspensions of magnetizable particles and, commonly non-magnetic, carrier fluids [31, 42]. In the absence of a magnetic field, the suspension behaves as a non-Newtonian fluid [6]. Applying a uniform magnetic field results in the formation of structures within the MRF that form along the magnetic field. These structures resist flow, thereby changing the yield stress and viscosity of the fluid. Only a few studies exist were a non-uniform magnetic field is applied to an MRF. The particles in the MRF still form structures that follow the magnetic field but the fluid behaves differently [45, 7].

Currently MRFs are mainly used for dampers, brakes, clutches [35, 32] and in the modern precision machining field for flexible fixtures or surface polishing [28]. Magnetorheological fluids could be of use in hydrodynamic bearings to increase the performance of the bearing. Hydrodynamic bearings consist of a rotating shaft, a supporting structure and a lubricant. When the shaft has no rotational velocity the shaft and the supporting structure are in contact. When the shaft rotates, the lubricant is squeezed and a fluid film forms between the shaft and the supporting structure. At low speeds the fluid film has not formed fully and there is still contact between the shaft and the supporting structure, which results in friction and wear. To minimize the rotational velocity required to fully form a fluid film, surface textures can be applied to the bearing surfaces to increase the fluid pressure. However, at low rotational speeds, the surfaces will still be in contact which results in wear of the textures [41, 39]. Replacing the lubricant by an MRF and applying a uniform magnetic field on the location of the textures would result in structures that resist flow, thereby increasing the film pressure. Applying a non-uniform field, using a permanent magnet, results in the formation of physical textures, due to the particles separating from the fluid and accumulating on the permanent magnet. Due to the bearing being a closed system, any particles that are sheared of these textures eventually accumulates on the texture again, resulting in self-healing behavior [15, 43, 29].

The phenomenon that particles separate from the fluid and accumulate on the permanent magnet has recently been discovered. Therefore, not much is known about this behavior. A numerical model to predict the steady state behavior of these self-healing textures does exist, however quite a few aspects of this phenomenon have not been investigated. One of which is the transition from the formation of structures to the formation of textures in a non-uniform magnetic field. A model that can simulate the behavior of an MRF in non-uniform magnetic fields would give insight into how the textures are formed and if the transition from structures to textures needs to be taken into account for the rheological behavior of the MRF.

This study has the following objectives:

- Design a numerical model that gives insight into the formation of textures when a non-uniform field is applied
- The model should be able to simulate the MRF until steady state is reached in a feasible time frame.
- The model must be able to simulate large domains so that a large part of the non-uniform field can be simulated.
- The model must be able to show whether the transition from structures to textures needs to be taken into account for the rheological behavior of the MRF.

The MRF will be modeled using the discrete element method. The discrete method models the fluid and particles separately. This method is mainly used to examine the particle dynamics and structure formation within the MRF. The constructed model will be based on existing discrete element models for MRFs and particulate flows.

This paper first discusses the method used to produce a parallel discrete element model, then goes on to the results of the simulations done using this model. These results will then be discussed, after which the conclusions of this paper are given.

2. Methodology

In this section the parallel discrete element model is presented. Firstly, the discrete element method will be described. Then the parallelization methods will be discussed, after which the details of the full model will be given. Finally, the stable time step for this model is determined.

2.1. Discrete element model

In discrete element models the fluid and the particles are modeled separately, where the dynamics of the fluid is solved using the Navier-Stokes equation and the dynamics of each particle is solved using Newton's equation of motion. Basic physical laws are used to model individual phases and their interaction.

The equation of motion of a particle a with a mass M_a and an acceleration $\boldsymbol{a}_a$ is shown in Eq. (1).

$$M_a \frac{\mathrm{d}^2 \boldsymbol{r}_a}{\mathrm{d}t^2} = M_a \boldsymbol{a}_a = \boldsymbol{F}_a^{\mathrm{c}} + \boldsymbol{F}_a^{\mathrm{mag}} + \boldsymbol{F}_a^{\mathrm{f}} + \boldsymbol{F}_a^{\mathrm{ext}} \tag{1}$$

This section will start by discussing the contact force acting on a particle, $\boldsymbol{F}_a^{\mathrm{c}}$. Then the magnetic forces, $\boldsymbol{F}_a^{\mathrm{mag}}$, are discussed, after which the interaction between the fluid and the particles, $\boldsymbol{F}_a^{\mathrm{f}}$, is described. Lastly, the external forces, $\boldsymbol{F}_a^{\mathrm{ext}}$, are discussed. The section will continue by discussing the method to determine the shear stresses in the MRF. Then the integration scheme used in the models will be described and the boundary condition for the model will be presented.

2.1.1. Contact force

The contact force, $\boldsymbol{F}_a^{\mathrm{c}}$, acting on a particle consists of the contact force acting from particle b on particle a and of contact forces acting on particle a due to a wall collision. The contact forces are modeled using a simplified Hertzian

model [45, 27, 40, 23, 26, 14, 26, 17, 18, 50, 11, 33]. This model only takes the normal component of the collision force into account, slip is thus neglected. The contact force acting from particle b on particle a is determined using Eq. (2).

$$\boldsymbol{F}_{ab}^{\mathrm{c}} = \left(\frac{\frac{2}{3}E}{1-\nu^2} \sqrt{R^{\mathrm{eff}}} \xi_{ab}^{3/2} \right) \hat{\boldsymbol{r}}_{ab} \tag{2}$$

where $R^{\mathrm{eff}} = R_a R_b / \left(R_a + R_b\right)$, $\xi_{ab} = \max\left\{R_a + R_b - \left\|\boldsymbol{r}_{ab}\right\|, 0\right\}$ and $\boldsymbol{r}_{ab} = \left(\boldsymbol{r}_a - \boldsymbol{r}_b\right)$.

The time step of the simulation depends on the minimum collision time of particles in the simulation. A stable time step can be determined by sampling the collision time of Eq. (3) over approximately five time steps [45].

$$t_{ab}^{\mathrm{c}} = 3.21 \left(\frac{\left(1-\nu^2\right) M_a M_b}{\frac{2}{3} E \sqrt{R^{\mathrm{eff}}} \left(M_a + M_b\right)} \right)^{2/5} \left\|\boldsymbol{v}_{ab}\right\|^{-1/5} \tag{3}$$

The contact force from a wall d on a particle a is determined using Eq. (4).

$$\boldsymbol{F}_{ad}^{\mathrm{c}} = \left(\frac{\frac{2}{3}E}{1-\nu^2} \sqrt{R} \xi_{ad}^{3/2} \right) \hat{\boldsymbol{r}}_{ad} \tag{4}$$

where $\xi_{ad} = \max\left\{R_a - \left\|\boldsymbol{r}_{ad}\right\|, 0\right\}$. The total contact force acting on a particle is given by the summation of particle contacts and wall contacts. It should be noted that the Young's modulus can be reduced so that the time step can be increased, while still accurately approximating the behavior of the MRF [26].

2.1.2. Magnetic interaction force

The magnetic interaction force is modeled using the fixed dipole method. Particles are considered to be dipoles and are only induced by the external magnetic field [13, 38, 26, 14, 25, 17, 18, 21, 8, 30, 20, 12, 36]. The magnetic moment of the particles is determined using Eq. (5).

$$\boldsymbol{m}_a = 3 V^{\mathrm{p}} \beta \boldsymbol{H}^{\mathrm{ext}}\left(\boldsymbol{r}_a\right) \tag{5}$$

where $\beta = \frac{\mu^{\mathrm{pr}} - \mu^{\mathrm{fr}}}{\mu^{\mathrm{pr}} + 2\mu^{\mathrm{fr}}}$, $\mu^{\mathrm{pr}} = \frac{\mu^{\mathrm{p}}}{\mu_0}$ and $\mu^{\mathrm{fr}} = \frac{\mu^{\mathrm{f}}}{\mu_0}$.

The induced particles generate a magnetic field that result in interaction forces between the particles. The magnetic field generated by a dipole, as seen from its centroid, is determined using Eq. (6).

$$\boldsymbol{H}_a^{\mathrm{dip}}\left(\boldsymbol{r}_a^{\mathrm{p}}\right) = \frac{1}{4\pi} \left(\frac{3\left(\boldsymbol{m}_a \cdot \boldsymbol{r}_a^{\mathrm{p}}\right)}{\left\|\boldsymbol{r}_a^{\mathrm{p}}\right\|^5} - \frac{\boldsymbol{m}_a}{\left\|\boldsymbol{r}_a^{\mathrm{p}}\right\|^3} \right) \tag{6}$$

The interaction force acting on a dipole due to the magnetic field produced by another dipole is given by Eq. (7).

$$\begin{aligned} \boldsymbol{F}_{ab}^{\mathrm{mag}} &= \mu_0 \nabla_{\boldsymbol{r}_a} \left(\boldsymbol{m}_a \cdot \boldsymbol{H}_b^{\mathrm{dip}}\left(\boldsymbol{r}_{ab}\right) \right) \\ &= \frac{3\mu_0}{4\pi \left\|\boldsymbol{r}_{ab}\right\|^4} \Big(\\ &\left(\boldsymbol{m}_a \cdot \boldsymbol{m}_b\right) \hat{\boldsymbol{r}}_{ab} + \left(\boldsymbol{m}_b \cdot \hat{\boldsymbol{r}}_{ab}\right) \boldsymbol{m}_a + \\ &\left(\boldsymbol{m}_a \cdot \hat{\boldsymbol{r}}_{ab}\right) \boldsymbol{m}_b - \\ &5 \left(\boldsymbol{m}_a \cdot \hat{\boldsymbol{r}}_{ab}\right) \left(\boldsymbol{m}_b \cdot \hat{\boldsymbol{r}}_{ab}\right) \hat{\boldsymbol{r}}_{ab} \Big) \end{aligned} \tag{7}$$

Cut-off distance In order to improve the computational speed of these models, the forces are not calculated over the whole simulation domain. The magnetic interaction force will only be taken into account when $\left\|\boldsymbol{r}_{ab}\right\| \leq 6R$, which is called the cut-off distance [19, 18, 20, 45].

2.1.3. Particle fluid interaction

Magnetorheological fluids are mainly used in configurations where the inertial forces are small compared with the viscous forces. The force acting on the particles by the fluid can therefore be approximated using Stokes' drag force of Eq. (8).

$$\boldsymbol{F}_a^{\mathrm{f}} = 6\pi\eta^{\mathrm{f}} R \left(\boldsymbol{v}^{\mathrm{f}}\left(\boldsymbol{r}_a\right) - \boldsymbol{v}_a\right) \tag{8}$$

The MRF is confined between two plates in the z-direction. The upper plate has a velocity of $\boldsymbol{v}^{\mathrm{top}} = v^{\mathrm{top}} \boldsymbol{e}_y$ and the bottom plate is stationary. A Couette flow, with a shear rate of $\dot{\gamma} = v^{\mathrm{top}} / L_z$, is thus generated. The fluid velocity at a particles centroid is determined using Eq. (9).

$$\boldsymbol{v}^{\mathrm{f}}\left(\boldsymbol{r}_a\right) = \dot{\gamma}\left(\boldsymbol{r}_a\right)_z \boldsymbol{e}_y \tag{9}$$

2.1.4. External forces

Brownian forces Brownian forces act on small particles in a fluid due to thermal fluctuations. The Peclet number, (Pe), given in Eq. (10), gives the ratio of the shear forces to the Brownian forces. The parameters used for this research, result in large values of Pe, therefore Brownian motion can be neglected [14, 9, 23, 4, 3].

$$\mathrm{Pe} = \frac{6\pi\eta^{\mathrm{f}} R^3 \dot{\gamma}}{\kappa_{\mathrm{B}} T} \tag{10}$$

Gravity The two phases of the MRF have a mismatch in density. This results in the sedimentation of particles which is unwanted in long term use of the fluid [48]. Several methods are used by industry to overcome the settling behavior of MRFs [24, 42, 2]. In order to incorporate these stabilization techniques into the model, the particles are assumed to be neutrally buoyant. It should be noted that these stabilization techniques influence the magnetorheological effect of the MRF, which are not taken into account.

External magnetic force The force acting on a magnetic dipole due to an external magnetic field is given by Eq. (11). In a uniform magnetic field, this force is equal to zero because there is no gradient in the magnetic field [45].

$$\boldsymbol{F}_a^{\mathrm{mag,ext}} = \mu_0 \nabla \left(\boldsymbol{m}_a \cdot \boldsymbol{H}^{\mathrm{ext}} \left(\boldsymbol{r}_a \right) \right) \tag{11}$$

Permanent bar magnets are modeled by assuming that the magnet is a continuous distribution of dipoles with a volume $\mathrm{d}V$ and a magnetic dipole moment of $\mathrm{d}\boldsymbol{m} = \boldsymbol{m}^{\mathrm{dip}} = \frac{1}{\mu_0} \boldsymbol{B}^{\mathrm{rem}} \, \mathrm{d}V$. The magnetic field produced by a dipole is given by Eq. (6). The external magnetic field generated by the bar magnet is determined by integrating the magnetic field contributions of the continuous distribution of dipoles over the volume of the magnet. The magnetic field of a homogeneously polarized bar magnet with dimensions $2L_x^{\mathrm{pm}} \times 2L_y^{\mathrm{pm}} \times 2L_z^{\mathrm{pm}}$ is thus given by Eqs. (12).

$$\begin{aligned} \mathrm{d}\boldsymbol{B}^{\mathrm{ext}} \left(\boldsymbol{r}_0 \right) &= \frac{\mu_0}{4\pi} \frac{3\hat{\boldsymbol{r}} \left(\hat{\boldsymbol{r}} \cdot \mathrm{d}\boldsymbol{m} \right) - \mathrm{d}\boldsymbol{m}}{\| \boldsymbol{r} \|^3} \\ &= \frac{1}{4\pi} \frac{3\hat{\boldsymbol{r}} \left(\hat{\boldsymbol{r}} \cdot \boldsymbol{B}^{\mathrm{rem}} \right) - \boldsymbol{B}^{\mathrm{rem}}}{\| \boldsymbol{r} \|^3} \, \mathrm{d}V \end{aligned} \tag{12a}$$

$$\begin{aligned} \boldsymbol{B}^{\mathrm{ext}} \left(\boldsymbol{r}_0 \right) = &\int_{-L_x^{\mathrm{pm}}}^{L_x^{\mathrm{pm}}} \int_{-L_y^{\mathrm{pm}}}^{L_y^{\mathrm{pm}}} \int_{-L_z^{\mathrm{pm}}}^{L_z^{\mathrm{pm}}} \\ &\mathrm{d}\boldsymbol{B}^{\mathrm{ext}} \left(\boldsymbol{r}_0 \right) \mathrm{d}x^{\mathrm{dip}} \, \mathrm{d}y^{\mathrm{dip}} \, \mathrm{d}z^{\mathrm{dip}} \end{aligned} \tag{12b}$$

$$\boldsymbol{H}^{\mathrm{ext}} \left(\boldsymbol{r}_0 \right) = \frac{\boldsymbol{B}^{\mathrm{ext}} \left(\boldsymbol{r}_0 \right)}{\mu_0} - \boldsymbol{M}^{\mathrm{pm}} \left(\boldsymbol{r}_0 \right) \tag{12c}$$

where $\boldsymbol{r} = \boldsymbol{r}_0 - \boldsymbol{r}^{\mathrm{dip}}$, $\boldsymbol{r}_0$ is a position in free space, $\boldsymbol{r}^{\mathrm{dip}}$ is the location of the dipole in the magnet, $\boldsymbol{M}^{\mathrm{pm}} \left(\boldsymbol{r}_0 \right) = \frac{\boldsymbol{B}^{\mathrm{rem}}}{\mu_0}$ inside the magnet, $\boldsymbol{M}^{\mathrm{pm}} \left(\boldsymbol{r}_0 \right) = 0$ outside the magnet and the origin is located at the center of the magnet [47, 46, 5].

2.1.5. Shear stress

The shear stress of the fluid is of interest due to the applied shear rate. The shear stress of the MRF is given by the (z, y) component of the stress tensor. The internal and external shear stresses are determined using Eqs. (13) [45, 37, 36, 49, 12]. Next to the contribution of the shear stress of the particle structures, there is a viscous addition. The viscous shear stress in the fluid is determined using the Krieger-Daugherty model of Eq. (14) where $\varphi^{\max} = 0.63$ [26, 22]. Finally, the total shear stress is determined by the summation of the external or internal shear stress and the viscous shear stress.

$$\tau^{\mathrm{int}} = -\frac{1}{V} \sum_a^{N^{\mathrm{p}}} \sum_{b=a+1}^{N^{\mathrm{p}}} \left(\boldsymbol{r}_{ab} \right)_z \left(\boldsymbol{F}_{ab}^{\mathrm{c}} + \boldsymbol{F}_{ab}^{\mathrm{mag}} \right)_y \tag{13a}$$

$$\tau^{\mathrm{ext}} = \frac{1}{V} \sum_a^{N^{\mathrm{p}}} \left(\boldsymbol{r}_a \right)_z \left(\boldsymbol{F}_a^{\mathrm{f}} \right)_y \tag{13b}$$

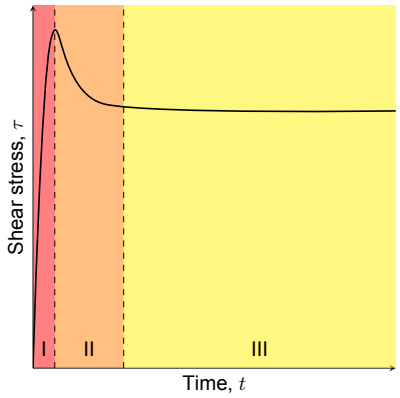


Figure 1: Shear stress response

$$\tau^{\mathrm{visc}} = \eta^{\mathrm{f}} \dot{\gamma} \left(1 - \frac{\varphi}{\varphi^{\max}} \right)^{-2.5\varphi^{\max}} \tag{14}$$

The shear stress versus time response for an MRF in a uniform magnetic field can be distinguished into three regions, as shown in Fig. 1. Region I is the elastic region which shows the low-strain regime where the stress is proportional to the shear strain. When the viscous contribution can be neglected, the static yield stress is located at the peak. In region II, the shear stress starts to decline to a steady state in region III. The steady state stress is known as the dynamic yield stress [12, 36].

2.1.6. Integration scheme

The equation of motion from Eq. (1) is solved using the modified velocity-Verlet algorithm. This is a simple, convenient, numerically stable method that is reversible in time. Furthermore, if the forces are conservative the method guarantees conservation of linear momentum [1]. The original velocity-Verlet algorithm assumes that the acceleration of a particle only depends on its position. However, due to the Stokes drag force of Eq. (8) the acceleration of the particle depends on its velocity as well. The algorithm is therefore modified to make a prediction for the new velocity. This prediction is corrected in the final step of the algorithm. The algorithm is shown in Eqs. (15) where the empirical parameter λ is chosen as 0.65 [34, 16, 45].

$$\boldsymbol{r}_a \left(t + \Delta t \right) = \boldsymbol{r}_a \left(t \right) + \Delta t \boldsymbol{v}_a \left(t \right) + \frac{1}{2} \left(\Delta t \right)^2 \boldsymbol{a}_a \left(t \right) \tag{15a}$$

$$\boldsymbol{v}_a^{\mathrm{pred}} \left(t + \Delta t \right) = \boldsymbol{v}_a \left(t \right) + \lambda \Delta t \boldsymbol{a}_a \left(t \right) \tag{15b}$$

$$\boldsymbol{a}_a \left(t + \Delta t \right) = \boldsymbol{a}_a \Big(\boldsymbol{r}_a \left(t + \Delta t \right), \boldsymbol{v}_a^{\mathrm{pred}} \left(t + \Delta t \right) \Big) \tag{15c}$$

$$\boldsymbol{v}_a \left(t + \Delta t \right) = \boldsymbol{v}_a \left(t \right) + \frac{1}{2} \Delta t \Big(\boldsymbol{a}_a \left(t \right) + \boldsymbol{a}_a \left(t + \Delta t \right) \Big) \tag{15d}$$

2.1.7. Boundary conditions

The MRF is confined in a simulation box of dimensions $L_x \times L_y \times L_z$ for each model. Conditions are placed on the

boundaries of this box. On the bottom and top of the box at $z = 0$ and $z = L_z$ walls are placed, confining the particles in the z-direction. As discussed in Section 2.1.3, the top wall has a velocity, $\boldsymbol{v}^{\text{top}}$, and the bottom wall is stationary which results in a Couette flow.
In the x-direction and y-direction PBCs (Periodic Boundary Condition) are placed. This is equivalent to each particle having an image of itself in neighboring image cells, as shown in Fig. 2, where the blue square is the simulated domain. By using PBCs, a small part of a large domain can be simulated to approximate the large domain [13, 20]. To make sure a particle only interacts with one image of any other particle, the minimum image convention is used. The position vector between two particles is determined using Eqs. (16) [45]. The minimum image convention requires the distance between the boundaries to be at least twice the cut-off distance, therefore $L_x \geq 2 \times 6R$ and $L_y \geq 2 \times 6R$. If this is not the case, particles can interact multiple times with each other.

$$\left(\boldsymbol{r}_{ab}\right)_x = \begin{cases} \left(\boldsymbol{r}_{ab}\right)_x + L_x, & \text{if} \quad \left(\boldsymbol{r}_{ab}\right)_x < -\frac{L_x}{2} \\ \left(\boldsymbol{r}_{ab}\right)_x - L_x, & \text{if} \quad \left(\boldsymbol{r}_{ab}\right)_x > \frac{L_x}{2} \end{cases} \tag{16a}$$

$$\left(\boldsymbol{r}_{ab}\right)_y = \begin{cases} \left(\boldsymbol{r}_{ab}\right)_y + L_y, & \text{if} \quad \left(\boldsymbol{r}_{ab}\right)_y < -\frac{L_y}{2} \\ \left(\boldsymbol{r}_{ab}\right)_y - L_y, & \text{if} \quad \left(\boldsymbol{r}_{ab}\right)_y > \frac{L_y}{2} \end{cases} \tag{16b}$$

2.2. Parallelization

To simulate large domains of the MRF in a feasible time frame, the simulation domain is decomposed into columns. Each column represents a process using one core. Particles that are present in a column are mastered by that process. The behavior of the particles is thus only determined by their master process, which greatly reduces the run time of the simulations. The processes are able to communicate with their direct neighbors to determine the interaction forces between the particles. Each process thus communicates with 8 other processes, taking PBCs into account. The communication between the processes is managed by OpenMPI.

All particles within the cut-off distance must be included, therefore the dimension of a process must be at least equal to the cut-off distance i.e., $L_x^{\text{proc}} \geq 6R$ and $L_y^{\text{proc}} \geq 6R$. An overview is given by a top view projection in Fig. 2, where the light blue square is the simulation domain, the dark blue cell is a process within the domain and the marked squares are the neighboring 8 processes. A pseudocode of the final model is given by Algorithm 1.

2.3. Model stability

The MRF is simulated in a uniform field using a small time step to determine the maximum allowed time step. This uniform field only has a component in the z-direction of $H^{\text{ext}} = 0.1/\mu_0$T. The simulations are done using the parameters of Table 1 where $\dot{\gamma} = 0\,\text{s}^{-1}$, $\Delta t = 1$ ns and $t^{\text{final}} = 1$ ms. The maximum kinetic energy of the simulation is shown in Fig. 3. The figure shows that the maximum kinetic energy of the particles is lower than 0.6 fJ. The maximum relative velocity between the particles is thus less than $0.097\,\text{m}\,\text{s}^{-1}$. Equation (3) gives that the collision times between the particles are lager thus than 2.6733×10^{-7} s. Using the sampling of Xuan et al., the stable time step is determined as approximately 5.5×10^{-8} s [45]. To ensure the model is stable, a time step of 5×10^{-8} s will be used.

Algorithm 1 Pseudocode

```
Load parameters
t = 0
Initialize the processes
Divide the simulation domain
Initialize and assign particles
Determine and save the dynamic evolution
for each step until t^final is reached do
    Integration of the positions and velocities
    Determine and save the dynamic evolution
    if modulo of step/60 = 0 then
        Store the positions and velocities
    end if
    t = t + Δt                                  ▷ Update time
end for
```

2.4. Number of processes

The model is simulated on the Delft High Performance Computing Centre (DHPC) [10]. To make sure that DHPC's resources are not misused, the number of processes required to simulate the MRF in a feasible time frame is determined. Using a power interpolation, the run time for a single process is estimated to be 8.234×10^5 min. By dividing the domain in 480 processes, the simulation is sped up approximately 2.5×10^4 times. 240 processes are used for the remaining simulations to make sure that DHPCs resources are not misused. It takes approximately 60 minutes to simulate the MRF using this amount of processes on DHPC.

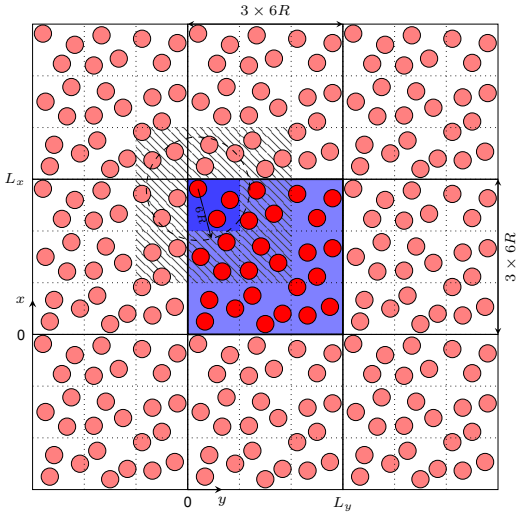


Figure 2: Domain decomposition with PBCs

Table 1
Parameters

Parameter	Value	Unit
General parameters		
μ_0	$4\pi \times 10^{-7}$	$\mathrm{N\,A^{-2}}$
λ	0.65	
φ	0.2	
t^{final}	7.5	ms
L_x	$9R$	m
L_y	$8000R$	m
L_z	$15R$	m
Fluid parameters		
μ^{f}	μ_0	$\mathrm{N\,A^{-2}}$
ρ^{f}	1000	$\mathrm{kg\,m^{-3}}$
η^{f}	0.2	Pa s
Particle parameters		
μ^{p}	2000 μ_0	$\mathrm{N\,A^{-2}}$
R	2.5	ţm
ρ^{p}	7800	$\mathrm{kg\,m^{-3}}$
E	500	MPa
ν	0.3	

The stable model will be used to simulate the MRF in the uniform magnetic field and three non-uniform magnetic fields produced by three bar magnets that were used in Lucieers experiments [29].The first is a 5 mm × 5 mm × 2 mm (N52) bar magnet, the second a 6 mm × 5 mm × 2 mm (N45) bar magnet and the third a 7 mm × 6 mm × 2 mm (N45) bar magnet. Just as in Lucieer's experiments, the magnets are placed 2 mm under the simulation domain [29]. The domain in the x-direction is small, because the direction of flow is of interest. Furthermore, this way the gradient of the magnetic field in x-direction can be neglected. The MRF will be simulated using of shear rates of 0, 500, 1000, 2500, 5000, 7500 and 10 000 $\mathrm{s^{-1}}$.

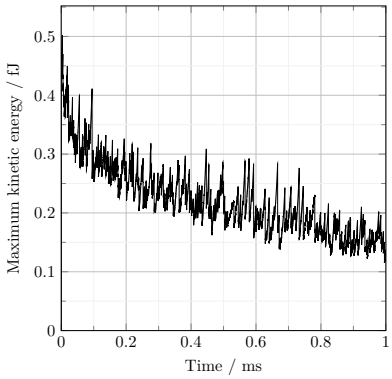


Figure 3: Maximum kinetic energy to determine the stable time step

3. Results

This section will go over the results of the simulations performed using the parallel discrete element model. First, the steady state structures within the MRF will be shown. This section continuous by giving the shear stress response results, from which a rheological model is constructed.

3.1. Steady state structures

The model is used to simulate large domains with large numbers of particles. The proportions of the simulation domain make it unable to show the structures that are formed within the MRF in a graphic way. Therefore, the steady state structures of the MRF are inspected using slab volume fraction profiles in the y- and z-directions and density plots of the yx- and yz-planes.

3.1.1. Formation of planes

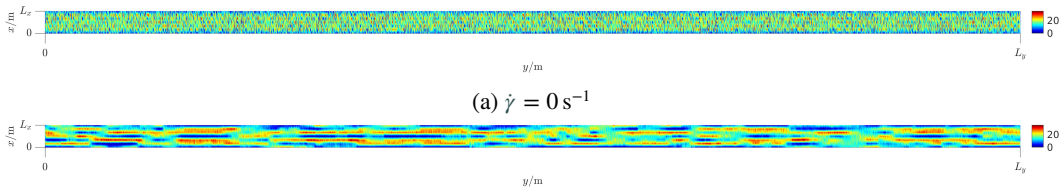
The density plots of the yx-planes for a uniform field, using different shear rates at steady state, are given in Fig. 4. The plots show a uniform distribution of the particles when there is no shear rate applied to the MRF. Applying a shear rate results in the formation of intertwined sheets parallel to the direction of flow. These planes become less intertwined with an increasing shear rate. Figure 5 shows a top view of the density of particles for different shear rates, using the first bar magnet. The density plots show that the particles only form sheets in the center of the simulation domain, where the magnet is located. The same holds for the other bar magnets.

3.1.2. Aggregation on the magnet

The volume fraction profiles in y-direction are given in Fig. 6. The profiles show that the volume fraction is constant when $\dot{\gamma} = 0\,\mathrm{s^{-1}}$. An increase of the shear rate results in an increase of the volume fraction at the back half of the magnet, after which the volume fraction decreases. The plots show that the fluctuations become larger with increasing shear rates. The differences of the volume fraction are larger for the first bar magnet and approximately the same for the second and third bar magnets.

Figure 7 shows the volume fraction profiles in z-direction for the different magnetic fields using different shear rates. The volume fractions between the magnetic fields do not differ much when $\dot{\gamma} = 0\,\mathrm{s^{-1}}$. There is a minute increase of the volume fraction at the bottom wall and decrease at the top wall for each bar magnet. For higher shear rates, differences in the volume fraction start to appear. For each bar magnet, an increase of the volume fraction can be seen at the bottom wall. Furthermore, the plots show a very small increase of the volume fraction at the top wall. At the center of the film height, an overall decrease of the volume fraction can be seen. The differences between the uniform field and the non-uniform fields increase with increasing shear rates. The relation between the volume fraction and z-height is approximately the same for all bar magnets.

Density plots of the yz-planes are used to show aggregations of particles in the simulation domain. Figure 8 shows the density plot for a uniform field when there is no shear

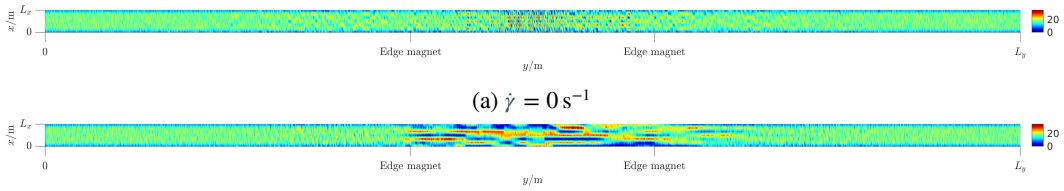


(a) $\dot{\gamma} = 0\,\mathrm{s}^{-1}$

(b) $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$

Figure 4: Density plots of the yx-plane for a uniform field

Color bar: Number of particles per $4R \times 3R$ m^2



(a) $\dot{\gamma} = 0\,\mathrm{s}^{-1}$

(b) $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$

Figure 5: Density plots of the yx-plane for the first bar magnet

Color bar: Number of particles per $4R \times 3R$ m^2

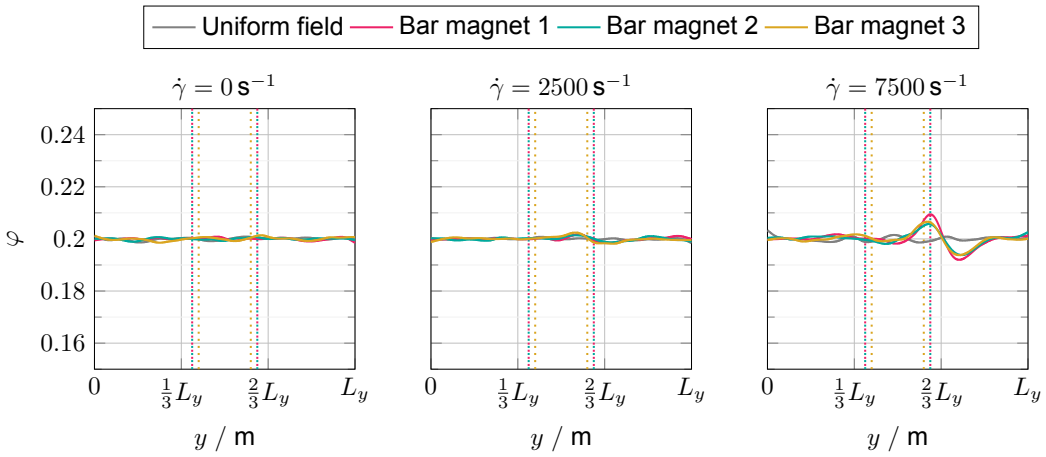


Figure 6: Slab volume fraction profiles in y-direction

Dotted lines: Edges of the magnets

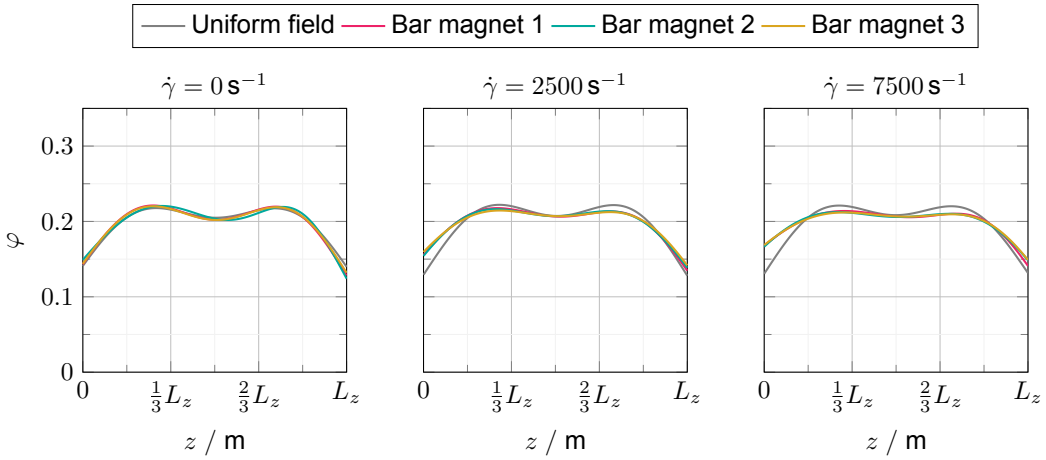


Figure 7: Slab volume fraction profiles in z-direction

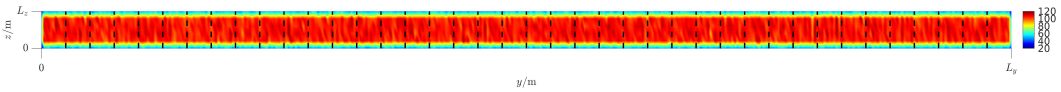


Figure 8: Density plot of the yz-plane for a uniform field where $\dot{\gamma} = 0\,\mathrm{s}^{-1}$

Color bar: Number of particle per $20R \times 6R$ m^2
Dashed lines: Magnetic field

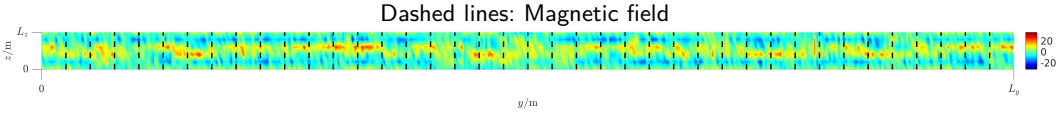


Figure 9: Density differences with respect to Fig. 8 where $\dot{\gamma} = 7500\,\mathrm{s}^{-1}$

Color bar: Number of particle per $20R \times 6R$ m^2
Dashed lines: Magnetic field

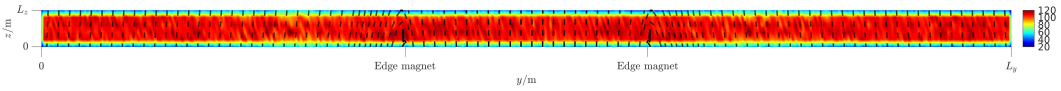
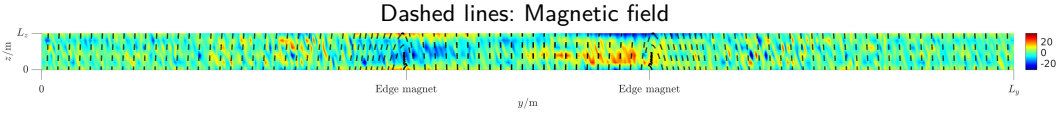
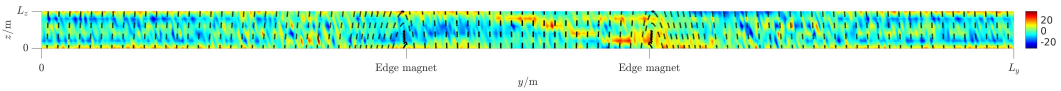


Figure 10: Density plot of the yz-plane for the first bar magnet

Color bar: Number of particle per $20R \times 6R$ m^2
Dashed lines: Magnetic field



(a) $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$



(b) $\dot{\gamma} = 7500\,\mathrm{s}^{-1}$

Figure 11: Density differences with respect to Fig. 10

Color bar: Number of particle per $20R \times 6R$ m^2
Dashed lines: Magnetic field

rate applied to the MRF. The plot shows a large number of particles at the center of the film height and a small number of particles at the walls. The differences in the number of particles with respect to Fig. 8 when a shear rate is applied, are shown in Fig. 9. These plots show an increase of the number of particles at the center of the film height.

Figure 10 shows the density plot for bar magnet 1 when there is no shear applied. The plot shows decreases of the number of particles just before and after the magnet with respect to Fig. 8. The differences in the density when a shear rate is applied are shown in Fig. 11. For lower shear rates, at the back of the magnet, particles have moved downwards from the top wall, resulting in an aggregation. At higher shear rates, there are increases over the whole film height. At the front of the magnet, the number of particles in the center of the film height decreases while increasing at the

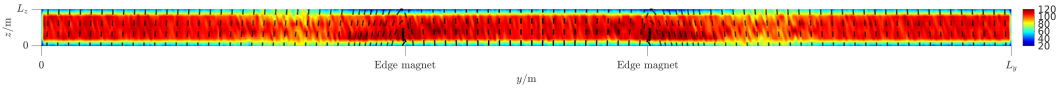


Figure 12: Density plot of the yz-plane for the first bar magnet with an increased t^{final}

Color bar: Number of particle per $20R \times 6R$ m^2
Dashed lines: Magnetic field

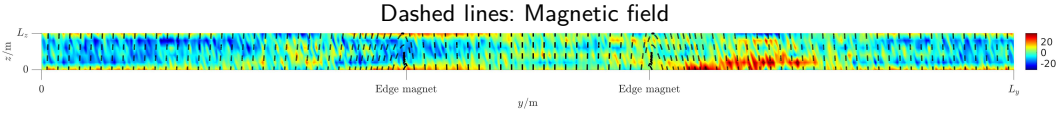


Figure 13: Density difference with respect to Fig. 12 where $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$

Color bar: Number of particle per $20R \times 6R$ m^2
Dashed lines: Magnetic field

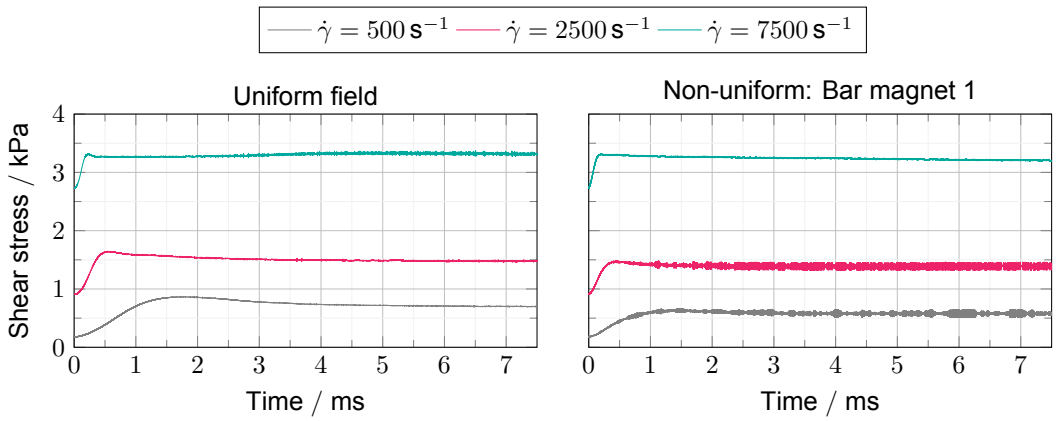


Figure 14: Shear stress response

walls. At higher shear rates, the number of particles at the top wall increases at the center of the magnet.

3.1.3. *Increased* t^{final}

Because the particles within the MRF have not fully reached steady state yet, the model is simulated with an increased t^{final} of 0.05 s. Figure 12 shows that decreases between the uniform field and the first bar magnet are more apparent when there is no shear applied. There are less particles located in front of and after the magnet. The density differences with respect to Fig. 12, when a shear rate of 5000 s^{-1} is applied, is shown in Fig. 13. There are increases after the back of the magnet at the bottom wall. Around the front of the magnet, there are decreases in the number of particles in the center of the film height and increases at the walls. In the center of the magnet, the overall number of particles increases, while at the edges of the simulation domain, the density decreases.

3.2. Shear stress response

The shear stress responses of the MRF in different fields are shown in Fig. 14. The regions discussed in Section 2.1.5 are clearly visible for all the shear rates. For all shear rates, the static and dynamic yield stresses increase with an increasing shear rate. The shear stress response of the first and second bar magnet are almost identical. The shear stress response of the third magnet follows the same curve, but the shear stresses are lower.

3.3. Rheological models

The results of Fig. 14 are used to determine the dynamic yield stress and construct a rheological model. The rheological models for all fields are shown in Fig. 15. For all fields the rheological models show an increase of the dynamic yield stress with an increasing shear rate. The first and second bar magnet show larger dynamic yield stresses than the third bar magnet. All curves follow approximately the same trend. The slope of the rheological models slightly declines at shear rates above 1000 s^{-1}.

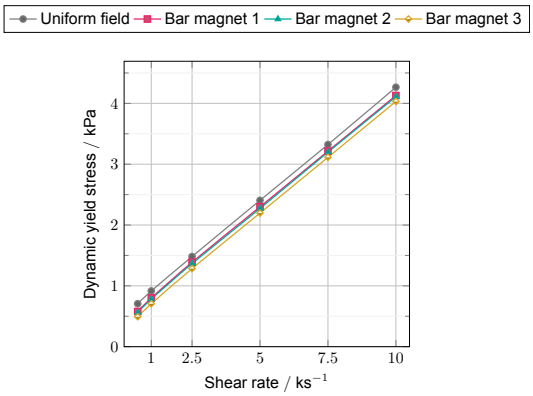


Figure 15: Rheological models

4. Discussion

4.1. Formation of planes

The particles within the MRF form planes parallel to the direction of flow. These planes become less intertwined with increasing shear rates. The planes are evenly spaced over the simulation domain due to the repellent forces between the dipoles when the poles are aligned. The formation of planes in a sheared MRF agrees with the sheets found by simulations of Lagger et al., Lagger et al. and the sheets found by experiments of Bossis et al., Volkova et al. [25, 26, 4, 44]. For the non-uniform fields the planes only form on the magnet, which is a result of the larger magnetic field strengths at these locations.

4.2. Aggregation on the magnet

The results show that the magnetic gradient force by itself only has a small influence on the formation of textures within the MRF. When there is no shear rate applied, there are only minute increases of the volume fraction in z-direction at the bottom wall and the volume fraction in y-direction remains constant. The density plot does show that particles have migrated from the low gradient magnetic field

just in front of and after the magnet, which is more apparent when the MRF is simulated with an increased t^{final}.

An increase of the shear rate results in the deformation of the particle chains. For a uniform field, this results in increases of the density at the center of the film height. For the non-uniform field, the orientation of the magnetic field lines results in different behavior. At the front of the magnet, the particle chains are oriented into the fluid flow. Particles amass at the top and bottom wall due to the rotation of these chains. This effect is more apparent when the MRF is simulated for a longer time. At the back of the magnet, particles rotate towards higher magnetic fields, thereby increasing the magnetic interaction force, which results in stronger structures that are able to resist the flow. This results in an increase of the density at the bottom wall. At larger shear rates, the particles aggregate where the gradient of the magnetic field is largest. The particles build up in front of this aggregation. With an increased t^{final}, the dimensions of the aggregations become larger. The large aggregation that is formed after the magnet, was also found by experiments of Lucieer [29].

4.3. Rheological models

The rheological models show an increase of the dynamic yield stress with an increased shear rate. The slope of the curves does however slightly decrease at shear rates larger than $1000\,\text{s}^{-1}$, which indicates shear thinning. The shear thinning in the MRF is a result of the stagnation of the internal shear stresses. These shear stresses are caused by the interaction forces between the particles, but at shear rates larger than $1000\,\text{s}^{-1}$ the steady state structures do not differ much. The internal shear stress will therefore plateau while the viscous shear stress will linearly increase, resulting in shear thinning.

The results show that the dynamic yield stress depends on the remanence and the y-dimension of the bar magnet. Bar magnet 1 has a larger remanence than bar magnet 2, resulting in larger magnetic field strength and thus larger magnetic interaction forces. The increase of these forces results in sturdier structures that are able to withstand the flow. Bar magnet 3 is smaller in the direction of flow than the other two magnets. Therefore, the area of large magnetic field strength within the simulation domain is decreased. This results in smaller magnetic interaction forces, therefore smaller internal shear stresses and thus a decreased dynamic yield stress.

The rheological models for all magnetic fields follow the same trend. It can therefore be stated that the same viscoplastic models can be used to approximate the behavior of the MRF in a uniform and non-uniform magnetic field for these shear rates.

5. Conclusion

This research has provided more insight into the formation of aggregates when a non-uniform magnetic field is applied to the MRF. The model that is constructed can simulate large domains of the MRF such that these effects can be studied. The findings show that the particles within the MRF form planes at increasing shear rates. The formation of these planes results in a stagnation of the internal shear stress and thus shear thinning occurs. The model shows that the particles do not aggregate on the magnets due to the attractive force of the bar magnets on the particles by itself. These aggregates are only formed when a shear rate is applied, therefore the formation must be a result of the orientation of the particle structures within the non-uniform field and the gradient magnetic force. The rheological models of the non-uniform fields follow the same trend as the uniform magnetic field. Therefore, for these shear rates, the viscoplastic models that are used to approximate the shear behavior of an MRF can also be applied in a non-uniform field.

For further research, the current model could be used to determine the dimension of the aggregation that is formed at the back of the magnet. The dimensions of the aggregate can be of use for modeling the MRF in a continuum way. Because the MRF has only been simulated for a small number of shear rates, more shear rates need to be simulated to confirm that the viscoplastic models can be used for non-uniform fields as well.

References

[1] Allen, M.P., Tildesley, D.J., 2017. Computer Simulation of Liquids. 2 ed., Oxford University Press, Oxford.

[2] Aruna, M.N., Rahman, M.R., Joladarashi, S., Kumar, H., 2019. Influence of additives on the synthesis of carbonyl iron suspension on rheological and sedimentation properties of magnetorheological (MR) fluids. Materials Research Express 6.

[3] Berli, C.L., De Vicente, J., 2012. A structural viscosity model for magnetorheology. Applied Physics Letters 101.

[4] Bossis, G., Volkova, O., Lacis, S., Meunier, A., 2002. Magnetorheology: Fluids, Structures and Rheology , 202–230.

[5] Camacho, J.M., Sosa, V., 2013. Alternative method to calculate the magnetic field of permanent magnets with azimuthal symmetry. Revista Mexicana de Fisica E 59, 8–17.

[6] Carlson, J.D., Jolly, M.R., 2000. MR fluid, foam and elastomer devices. Mechatronics 10, 555–569.

[7] Chen, W., Zhang, G., He, X., Ren, S., Huang, P., 2016. Developing of the experimental device that test magneto rheological fluid (MRF) shear yield stress nnder non uniform composite field. Key Engineering Materials 667, 385–390.

[8] Ciocanel, C., Lipscomb, G., Naganathan, N.G., 2006. A particle pair model for magnetorheological fluids. Smart Structures and Materials 2006: Active Materials: Behavior and Mechanics 6170, 61700S.

[9] De Vicente, J., Klingenberg, D.J., Hidalgo-Alvarez, R., 2011. Magnetorheological fluids: A review. Soft Matter 7, 3701–3710.

[10] Delft High Performance Computing Centre (DHPC), 2022. DelftBlue Supercomputer (Phase 1). https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase1.

[11] Di Renzo, A., Di Maio, F.P., 2004. Comparison of contact-force models for the simulation of collisions in DEM-based granular flow codes. Chemical Engineering Science 59, 525–541.

[12] Fernández-Toledano, J.C., Ruiz-López, J.A., Hidalgo-Álvarez, R., de Vicente, J., 2015. Simulations of polydisperse magnetorheological fluids: A structural and kinetic investigation. Journal of Rheology 59, 475–498.

[13] Fu, Y., Yao, J., Zhao, H., Zhao, G., Qiu, Y., 2019. Simulation of a bidisperse magnetorheological fluid using the combination of a two-component lattice Boltzmann method and a discrete element approach. Soft Matter 15, 6867–6877.

[14] Ghaffari, A., Hashemabadi, S.H., Ashtiani, M., 2015. A review on the simulation and modeling of magnetorheological fluids. Journal of Intelligent Material Systems and Structures 26, 881–904.

[15] de Graaf, M.C., 2019. A new concept in bearing technology: Magnetorheological texturing (master thesis) , 86.

[16] Groot, R.D., Warren, P.B., 1997. Dissipative particle dynamics: Bridging the gap between atomistic and mesoscopic simulation. Journal of Chemical Physics 107, 4423–4435.

[17] Han, K., Feng, Y.T., Owen, D.R., 2010a. Modelling of magnetorheological fluids with combined lattice Boltzmann and discrete element approach. Communications in Computational Physics 7, 1095–1117.

[18] Han, K., Feng, Y.T., D.R.J.Owen, 2010b. Three-dimensional modelling and simulation of magnetorheological fluids. International , 1102–1119.

[19] Ji, D., Luo, Y., Ren, H., Wei, D., Shao, J., 2019. Numerical Simulation and Experimental Analysis of Microstructure of Magnetorheological Fluid. Journal of Nanomaterials 2019.

[20] Kargulewicz, M., Iordanoff, I., Marrero, V., Tichy, J., 2012. Modeling of magnetorheological fluids by the discrete element method. Journal of Tribology 134, 1–9.

[21] Keaveny, E.E., Maxey, M.R., 2008. Modeling the magnetic interactions between paramagnetic beads in magnetorheological fluids. Journal of Computational Physics 227, 9554–9571.

[22] Krieger, I.M., 1972. Rheology of monodisperse latices. Advances in Colloid and Interface Science 3, 111–136.

[23] Kroupa, M., Vonka, M., Soos, M., Kosek, J., 2016. Utilizing the Discrete Element Method for the Modeling of Viscosity in Concentrated Suspensions. Langmuir 32, 8451–8460.

[24] Kumar, S., Sehgal, R., Wani, M.F., Sharma, M.D., 2021. Stabilization and tribological properties of magnetorheological (MR) fluids: A review. Journal of Magnetism and Magnetic Materials 538, 168295.

[25] Lagger, H.G., Bierwisch, C., Korvink, J.G., Moseler, M., 2014. Discrete element study of viscous flow in magnetorheological fluids. Rheologica Acta 53, 417–443.

[26] Lagger, H.G., Breinlinger, T., Korvink, J.G., Moseler, M., Di Renzo, A., Di Maio, F., Bierwisch, C., 2015. Influence of hydrodynamic drag model on shear stress in the simulation of magnetorheological fluids. Journal of Non-Newtonian Fluid Mechanics 218, 16–26.

[27] Leps, T., Hartzell, C., 2021. High fidelity, discrete element method simulation of magnetorheological fluids using accurate particle size distributions in LIGGGHTS extended with mutual dipole method. Materials Research Express 8, 085701.

[28] Lu, H., Hua, D., Wang, B., Yang, C., Hnydiuk-Stefan, A., Królczyk, G., Liu, X., Li, Z., 2021. The Roles of Magnetorheological Fluid in Modern Precision Machining Field: A Review. Frontiers in Materials 8, 1–11.

[29] Lucieer, B.T.D., 2020. Experimental investigation of self-healing magnetorheological surface textures .

[30] Manikas, K., Vogiatzis, G.G., Hütter, M., Anderson, P.D., 2021. Structure formation in suspensions under uniform electric or magnetic field. Multiscale and Multidisciplinary Modeling, Experiments and Design 4, 77–97.

[31] Morillas, J.R., De Vicente, J., 2020. Magnetorheology: a review. Soft Matter 16, 9614–9642.

[32] Olabi, A.G., Grunwald, A., 2007. Design and application of magnetorheological fluid. Materials and Design 28, 2658–2664.

[33] Paulick, M., Morgeneyer, M., Kwade, A., 2015. A new method for the determination of particle contact stiffness. Granular Matter 17, 83–93.

[34] Pei, L., Pang, H., Chen, K., Xuan, S., Gong, X., 2018. Simulation of the optimal diameter and wall thickness of hollow Fe3O4 microspheres in magnetorheological fluids. Soft Matter 14, 5080–5091.

[35] Phu, D.X., Choi, S.B., 2019. Magnetorheological fluid based devices reported in 20132018: Mini-review and comment on structural configurations. Frontiers in Materials 6.

[36] Ruiz-López, J.A., Wang, Z.W., Fernández-Toledano, J.C., Hidalgo-Alvarez, R., de Vicente, J., 2016. Start-up rheometry of highly polydisperse magnetorheological fluids: experiments and simulations. Rheologica Acta 55, 245–256.

[37] Ruiz-López, J.A., Wang, Z.W., Hidalgo-Alvarez, R., de Vicente, J., 2017. Simulations of model magnetorheological fluids in squeeze flow mode. Journal of Rheology 61, 871–881.

[38] Sand, A., Stener, J.F., Toivakka, M.O., Carlson, J.E., Pålsson, B.I., 2016. A Stokesian dynamics approach for simulation of magnetic particle suspensions. Minerals Engineering 90, 70–76.

[39] Singh, N., Awasthi, R.K., 2021. Influence of texture geometries on the performance parameters of hydrodynamic journal bearing. Proceedings of the Institution of Mechanical Engineers, Part J: Journal of Engineering Tribology 235, 2056–2072.

[40] Son, K.J., 2018. A discrete element model for the influence of surfactants on sedimentation characteristics of magnetorheological fluids. Korea Australia Rheology Journal 30, 29–39.

[41] Tala-Ighil, N., Fillon, M., Maspeyrot, P., 2011. Effect of textured area on the performances of a hydrodynamic journal bearing. Tribology International 44, 211–219.

[42] Thiagarajan, S., Koh, A.S., 2021. Performance and Stability of Magnetorheological FluidsA Detailed Review of the State of the Art. Advanced Engineering Materials 2001458, 1–24.

[43] van Der Meer, G.T.D., 2020. Modelling self-healing magnetic textures in hydrodynamic bearings. Thesis Topology .

[44] Volkova, O., Cutillas, S., Bossis, G., 1999. Shear banded flows and nematic-to-isotropic transition in er and mr fluids. Physical Review Letters 82, 233–236.

[45] Xuan, T., Li, J., Li, B., Fan, W., 2021. Effects of the non-uniform magnetic field on the shear stress and the microstructure of magnetorheological fluid. Journal of Magnetism and Magnetic Materials 535, 168066.

[46] Yang, Z.J., 1990. Magnetic Fields Around Permanent Magnets, University of Oslo, Oslo.

[47] Yang, Z.J., Johansen, T.H., Bratsberg, H., Helgesen, G., Skjeltorp, A.T., 1990. Potential and force between a magnet and a bulk Y1Ba 2Cu3O7-δ superconductor studied by a mechanical pendulum. Superconductor Science and Technology 3, 591–597.

[48] Zhang, Y., Li, D., Zhang, Z., 2019. The study of magnetorheological fluids sedimentation behaviors based on volume fraction of magnetic particles and the mass fraction of surfactants. Materials Research Express 6.

[49] Zhou, J., Mo, J., Shao, C., Li, Z., 2015. Effects of magnetized walls on the particle structure and the yield stress of magnetorheological fluids. Journal of Magnetism and Magnetic Materials 389, 124–129.

[50] Zhu, H.P., Zhou, Z.Y., Yang, R.Y., Yu, A.B., 2007. Discrete particle simulation of particulate systems: Theoretical developments. Chemical Engineering Science 62, 3378–3396.

A

Lubrication model

Lubrication force

When particles move relatively to each other, they move the fluid around them. When particles are far apart, this does not effect them. However, when particles are close together, the fluid gets squeezed out or sucked in between them. This results in a lubrication force acting between the particles. These forces are described by equations (A.1a) and (A.1b). Between a wall and a particle the same occurs, these forces are described by equations (A.1c) and (A.1d).

$$\boldsymbol{F}_{ab}^{\mathsf{nl}} = -6\pi\eta^{\mathsf{f}} R \boldsymbol{v}_{ab} \cdot \hat{\boldsymbol{r}}_{ab} \left(\frac{1}{4}\epsilon_{ab}^{-1} - \frac{9}{40}\log\epsilon_{ab} - \frac{3}{112}\epsilon_{ab}\log\epsilon_{ab} \right) \tag{A.1a}$$

$$\boldsymbol{F}_{ab}^{\mathsf{sl}} = -6\pi\eta^{\mathsf{f}} R \boldsymbol{v}^{\mathsf{s}} \left(-\frac{1}{6}\log\epsilon_{ab} \right) \tag{A.1b}$$

$$\boldsymbol{F}_{ad}^{\mathsf{nl}} = -6\pi\eta^{\mathsf{f}} R \boldsymbol{v}_{ad} \cdot \hat{\boldsymbol{r}}_{ad} \left(\epsilon_{ad}^{-1} - \frac{1}{5}\log\epsilon_{ad} - \frac{1}{21}\epsilon_{ad}\log\epsilon_{ad} \right) \tag{A.1c}$$

$$\boldsymbol{F}_{ad}^{\mathsf{sl}} = -6\pi\eta^{\mathsf{f}} R \boldsymbol{v}^{\mathsf{s}} \left(-\frac{8}{15}\log\epsilon_{ad} - \frac{64}{375}\epsilon_{ad}\log\epsilon_{ad} \right) \tag{A.1d}$$

where

$$\epsilon_{ab} = \frac{\boldsymbol{r}_{ab} - 2R}{R} \qquad \boldsymbol{v}_{ab} = \boldsymbol{v}_a - \boldsymbol{v}_b \qquad \boldsymbol{v}^{\mathsf{s}} = \boldsymbol{v}_{ab} - (\boldsymbol{v}_{ab} \cdot \hat{\boldsymbol{r}}_{ab})\,\hat{\boldsymbol{r}}_{ab}$$

$$\epsilon_{ad} = \frac{\boldsymbol{r}_{ad} - 2R}{R} \qquad \boldsymbol{v}_{ad} = \boldsymbol{v}_a - \boldsymbol{v}_d \qquad \boldsymbol{v}^{\mathsf{s}} = \boldsymbol{v}_{ad} - (\boldsymbol{v}_{ad} \cdot \hat{\boldsymbol{r}}_{ad})\,\hat{\boldsymbol{r}}_{ad}$$

These equations diverge when $\epsilon \to 0$, which results in infinitely high forces. In order to overcome this, equations (A.1) are multiplied by a correction factor, given in equation (A.2) [61]. Furthermore, for $\epsilon < \epsilon_{\mathsf{c}}$, ϵ_{c} is used in equations (A.1) instead of ϵ. Particles are considered to be within critical relative distance when $\epsilon \leq 0.1 = \epsilon_{\mathsf{c}}$ [20].

$$f^* = \frac{\epsilon}{3\epsilon_{\mathsf{c}}} \left(\left(1 + \frac{\epsilon}{6\epsilon_{\mathsf{c}}} \right) \ln\left(1 + \frac{6\epsilon_{\mathsf{c}}}{\epsilon} \right) - 1 \right) \tag{A.2}$$

B

Inclusion model

The inclusion model provides a way to include multipoles into the dipole methods by looking at the solution of a two-body problem. In this model only the interaction between two particles in a uniform magnetic field is taken into account. An illustration of the problem is shown in figure B.1, where $\boldsymbol{H}^{\text{ext}} = H^{\perp}\hat{\boldsymbol{x}} + H^{\parallel}\hat{\boldsymbol{z}}$. The solution to this problem is given by the sum of spherical harmonics defined as equations (B.1a) and (B.1b), where $\left(r_a^{\mathsf{p}}, \theta_a^{\mathsf{p}}, \phi_a^{\mathsf{p}}\right)$ are the spherical coordinates of particle a.

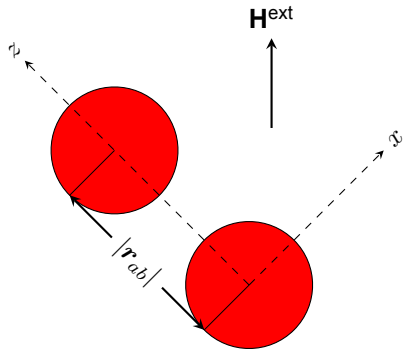


Figure B.1: The two body problem [29].

$$\Phi^{\mathsf{f}} = -H^{\perp}x - H^{\parallel}z + \sum_{a=1}^{2}\sum_{l=0}^{\infty}\sum_{m=0}^{1} B_{lm,a}\frac{P_l^m\left(\cos\theta_a^{\mathsf{p}}\right)}{\left(r_a^{\mathsf{p}}\right)^{l+1}}\cos m\phi_a^{\mathsf{p}} \tag{B.1a}$$

$$\Phi_a^{\mathsf{p}} = \sum_{l=0}^{\infty}\sum_{m=0}^{1} A_{lm,a}\left(r_a^{\mathsf{p}}\right)^l P_l^m\left(\cos\theta_a^{\mathsf{p}}\right)\cos m\phi_a^{\mathsf{p}} \tag{B.1b}$$

where associated Legendre polynomial $P_l^m\left(\cos\theta_a^{\mathsf{p}}\right)$ of equation (B.2) is used.

$$P_l^m\left(\cos\theta_a^{\mathsf{p}}\right) = (-1)^m\left(1-\cos^2\theta_a^{\mathsf{p}}\right)^{m/2}\frac{\mathsf{d}^m}{\mathsf{d}\left(\cos\theta\right)^m}P_l\left(\cos\theta_a^{\mathsf{p}}\right) \tag{B.2}$$

To find a solution for the general problem described by equations (2.6a) to (2.8c), the coordinate systems of the two particles need to be combined to a single system using the Hobson formula of equations (B.3) and (B.4).

$$\frac{P_l^m\left(\cos\theta_1^{\mathsf{p}}\right)}{\left(r_1^{\mathsf{p}}\right)^{l+1}} = \sum_{s=m}^{\infty}(-1)^{s+m}(-1)^{l+s}\binom{l+s}{s+m}\frac{\left(r_2^{\mathsf{p}}\right)^s}{|\boldsymbol{r}_{12}|^{l+s+1}}P_s^m\left(\cos\theta_2^{\mathsf{p}}\right) \tag{B.3}$$

$$\frac{P_l^m\left(\cos\theta_2^{\mathsf{p}}\right)}{\left(r_2^{\mathsf{p}}\right)^{l+1}} = \sum_{s=m}^{\infty}(-1)^{s+m}\binom{l+s}{s+m}\frac{\left(r_1^{\mathsf{p}}\right)^s}{|\boldsymbol{r}_{12}|^{l+s+1}}P_s^m\left(\cos\theta_1^{\mathsf{p}}\right) \tag{B.4}$$

Substituting equations (B.1a) to (B.4) into the boundary conditions of equations (2.8a) to (2.8c), assuming a finite number of multipoles N^{m} and eliminating$A_{lm,a}$ results in the linear systems of equations (B.5a) to (B.5g). These two $2N^{\mathsf{m}} \times 2N^{\mathsf{m}}$ systems needs to be solved for B in order to determine the magnetic field generated by the particles by substituting equations (B.1a) and (B.1b) into the general problem of equations (2.6a) to (2.7b). The resulting magnetic field can then be used in equations (2.9) and (2.10) to determine the force acting on each particle.The computational time of this model mainly depends on the number of multipoles that are taken into account. For closely spaced particles where $\frac{\mu^{\mathsf{pr}}}{\mu^{\mathsf{fr}}} \geq 50$ it is known that a 100 multipoles are not enough to approximate the exact solution [62]. Therefore, to get a good approximation of the exact solution, more multipoles are required which significantly increases the computational time of the model.

$$\begin{pmatrix}\boldsymbol{X} & \boldsymbol{\Delta}_m\\ \boldsymbol{\Gamma}_m & \boldsymbol{X}\end{pmatrix}\begin{pmatrix}\boldsymbol{B}_{m,1}\\ \boldsymbol{B}_{m,2}\end{pmatrix} = \begin{pmatrix}\boldsymbol{q}_m\\ \boldsymbol{q}_m\end{pmatrix} \tag{B.5a}$$

$$\boldsymbol{B}_{m,i} = \begin{pmatrix}B_{1m,1} & B_{2m,1} & \dots & B_{N^{\mathsf{m}}m,1}\end{pmatrix}^\top \tag{B.5b}$$

$$\boldsymbol{q}_0 = \begin{pmatrix}H^{\parallel}R^3\left(1-\mu^{\mathsf{pr}}\right) & 0 & \dots & 0\end{pmatrix}^\top \tag{B.5c}$$

$$\boldsymbol{q}_1 = \begin{pmatrix}-H^{\perp}R^3\left(1-\mu^{\mathsf{pr}}\right) & 0 & \dots & 0\end{pmatrix}^\top \tag{B.5d}$$

$$X_{ij} = \delta_{ij}\left(i\mu^{\mathsf{pr}}+i+1\right) \tag{B.5e}$$

$$\Delta_{ij,m} = (-1)^{i+m}\left(i\mu^{\mathsf{pr}}-i\right)\binom{i+j}{j+m}\frac{R^{2i+1}}{|\boldsymbol{r}_{12}|^{i+j+1}} \tag{B.5f}$$

$$\Gamma_{ij,m} = (-1)^{i+i}\,\Delta_{ij,m} \tag{B.5g}$$

An attempt was made to model the interaction between particle using the inclusion model. However, the model did not result in accurate magnetic interaction forces and proved to be very slow. Therefore it was decided to abandon the inclusion model and focus on the fixed and mutual dipole methods.

C

External magnetic field

C.1. Original model

This section will elaborate the original model used to determine the magnetic field generated by a homogeneously polarized along the x-direction bar magnet with dimensions $2L_x'^{\text{pm}} \times 2L_y'^{\text{pm}} \times 2L_z'^{\text{pm}}$. The model is shortly discussed in section 2.1.4 and the same equations will be used. The magnetic field of a bar magnet is given by equations (C.1) after integration of equation (2.23a) where $\boldsymbol{B}^{\text{rem}} = B^{\text{rem}}\boldsymbol{e}_x$.

$$B'_x(x_0, y_0, z_0) = \frac{B^{\text{rem}}}{4\pi} \int I_1(x_0, y_0, z_0) \tag{C.1a}$$

$$B'_y(x_0, y_0, z_0) = \frac{B^{\text{rem}}}{4\pi} \int I_2 \tag{C.1b}$$

$$B'_z(x_0, y_0, z_0) = \frac{B^{\text{rem}}}{4\pi} \int I_3(x_0, y_0, z_0) \tag{C.1c}$$

$$\boldsymbol{B}'(\boldsymbol{r}_0) = \begin{bmatrix} B'_x(\boldsymbol{r}_0) \\ B'_y(\boldsymbol{r}_0) \\ B'_z(\boldsymbol{r}_0) \end{bmatrix} \tag{C.1d}$$

where the integral factors $\int I_1$, $\int I_2$ and $\int I_3$ are given by equations (C.2).

$$\begin{aligned} \int I_1(x_0, y_0, z_0) = &- F_1(-x_0, y_0, z_0) - F_1(-x_0, y_0, -z_0) - F_1(-x_0, -y_0, z_0) - F_1(-x_0, -y_0, -z_0) \\ &- F_1(x_0, y_0, z_0) - F_1(x_0, y_0, -z_0) - F_1(x_0, -y_0, z_0) - F_1(x_0, -y_0, -z_0) \end{aligned} \tag{C.2a}$$

$$\int I_2(x_0, y_0, z_0) = \ln \frac{F_2(-x_0, -y_0, z_0) F_2(x_0, y_0, z_0)}{F_2(-x_0, y_0, z_0) F_2(x_0, -y_0, z_0)} \tag{C.2b}$$

$$\int I_3(x_0, y_0, z_0) = \ln \frac{F_3(-x_0, y_0, -z_0) F_3(x_0, y_0, z_0)}{F_3(-x_0, y_0, z_0) F_3(x_0, y_0, -z_0)} \tag{C.2c}$$

(C.2d)

and F_1, F_2 and F_3 are given by equations (C.3).

$$F_1(x,y,z) = \text{atan}\left(\frac{\left(L'_y+y\right)\left(L'_z+z\right)}{\left(L'_x+x\right)\sqrt{\left(L'_x+x\right)^2+\left(L'_y+y\right)^2+\left(L'_z+z\right)^2}}\right) \tag{C.3a}$$

$$F_2(x,y,z) = \frac{\sqrt{\left(L'_x+x\right)^2+\left(L'_y+y\right)^2+\left(L'_z+z\right)^2}+L'_z+z}{\sqrt{\left(L'_x+x\right)^2+\left(L'_y+y\right)^2+\left(L'_z-z\right)^2}-L'_z+z} \tag{C.3b}$$

$$F_3(x,y,z) = \frac{\sqrt{\left(L'_x+x\right)^2+\left(L'_y+y\right)^2+\left(L'_z+z\right)^2}+L'_y+y}{\sqrt{\left(L'_x+x\right)^2+\left(L'_y-y\right)^2+\left(L'_z+z\right)^2}-L'_y+y} \tag{C.3c}$$

C.2. Rotated model

In the discrete element models described in this report the bar magnet is homogeneously polarized along the z-direction. Therefore the model discussed in appendix C.1 is rotated $-\frac{\pi}{2}$ rad around the y axis. This is done using equations (C.4). It should be noted that the dimension of the magnet are rotated as well where $L'_x = L_z$, $L'_y = L_y$ and $L'_z = L_x$.

$$\boldsymbol{R} = \begin{bmatrix} \cos -\frac{\pi}{2} & 0 & \sin -\frac{\pi}{2} \\ 0 & 1 & 0 \\ -\sin -\frac{\pi}{2} & 0 & \cos -\frac{\pi}{2} \end{bmatrix} \tag{C.4a}$$

$$\boldsymbol{B}(\boldsymbol{r}_0) = \boldsymbol{R}\boldsymbol{B}'\left(\begin{bmatrix} \boldsymbol{R}^\top \boldsymbol{r}_0 \cdot \boldsymbol{e}_x \\ \boldsymbol{R}^\top \boldsymbol{r}_0 \cdot \boldsymbol{e}_y \\ \boldsymbol{R}^\top \boldsymbol{r}_0 \cdot \boldsymbol{e}_z \end{bmatrix}\right) \tag{C.4b}$$

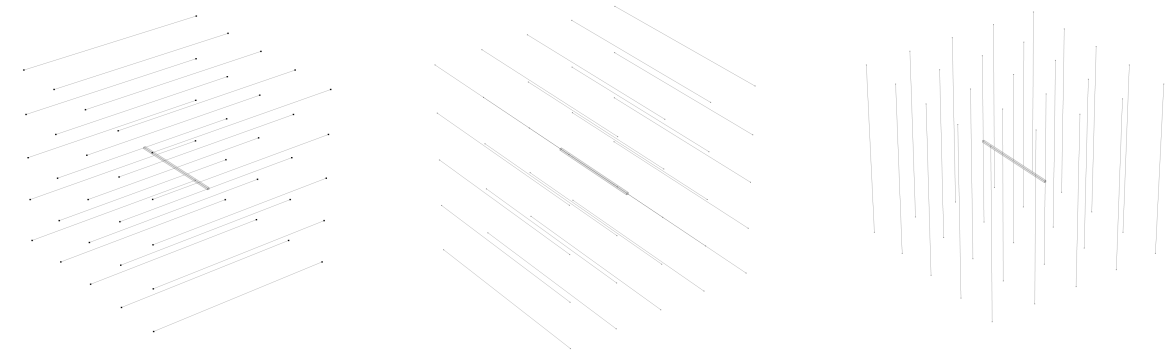
C.3. Verification

The model for the homogeneously polarized bar magnet has been verified using a COMSOL model. The COMSOL model consist of a bar magnet with the parameters shown in table C.1.

Parameter	Description	Value
L_x	x dimension	1×10^{-6} m
L_y	y dimension	50×10^{-6} m
L_z	z dimension	1×10^{-6} m
$B^{\text{rem}}\boldsymbol{e}_z$	Remanence	$750\mu_0$ T

Table C.1: Parameters used in the COMSOL model for the bar magnet

The COMSOL model and the model of appendix C.2 are compared by determining the magnetic field strength $\boldsymbol{H}^{\text{ext}}$ along 75 lines. The lines are build in the geometry section so that the mesh can follow the lines and the resulting magnetic field along this lines are smooth. An overview of the constructed lines is shown in figure C.1.



(a) Lines constructed parallel to the x axis **(b)** Lines constructed parallel to the y axis **(c)** Lines constructed parallel to the z axis

Figure C.1: Constructed lines in the COMSOL model

The bar magnet is placed inside a box with a relative permeability of 1 and the outer surfaces of this box have an infinite element domain.

From the overview of the comparison of the two models along the lines given in figures C.2 to C.4, it can be concluded that the two models result in the same magnetic field strengths. The only differences can be found when a line intersects with the magnet, this is not a problem since the inside of the magnet is not of interest.

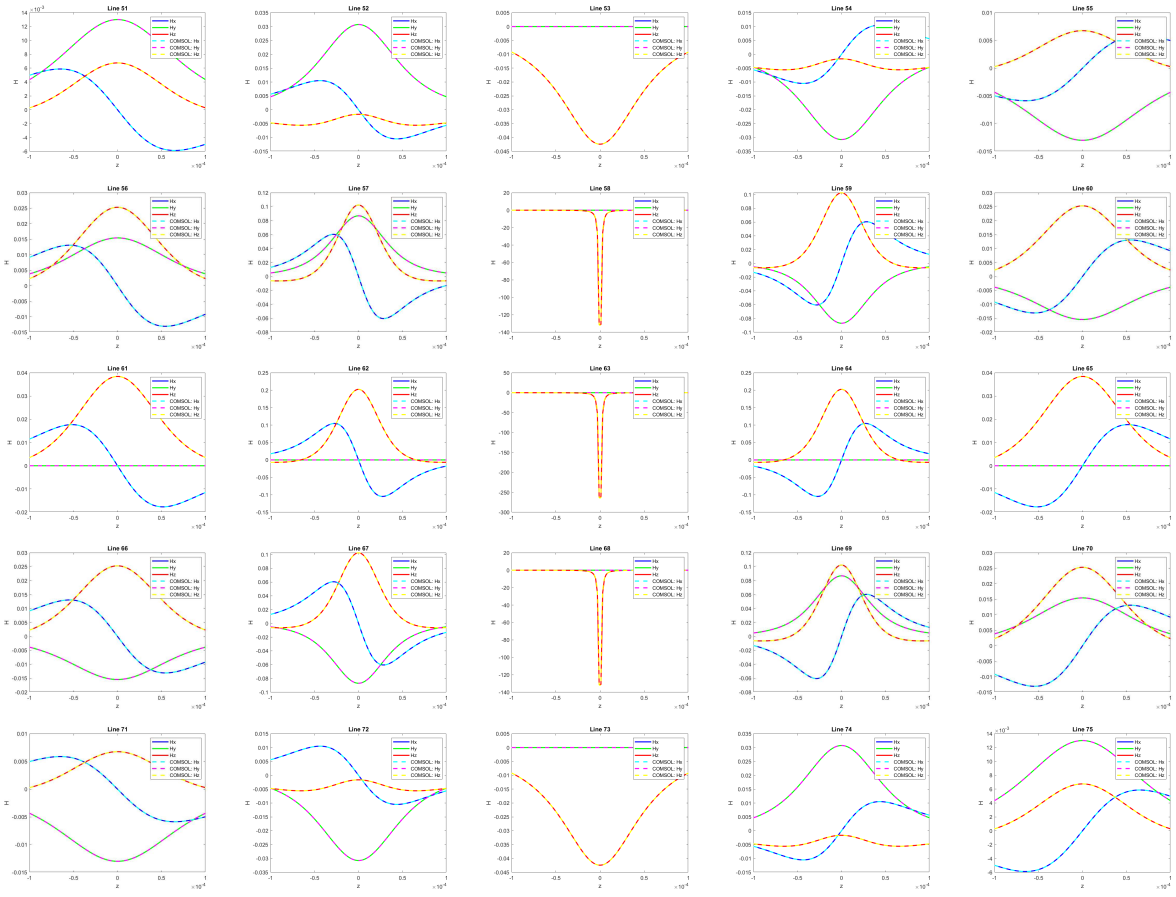


Figure C.2: Comparison of the two models over the constructed lines in x direction

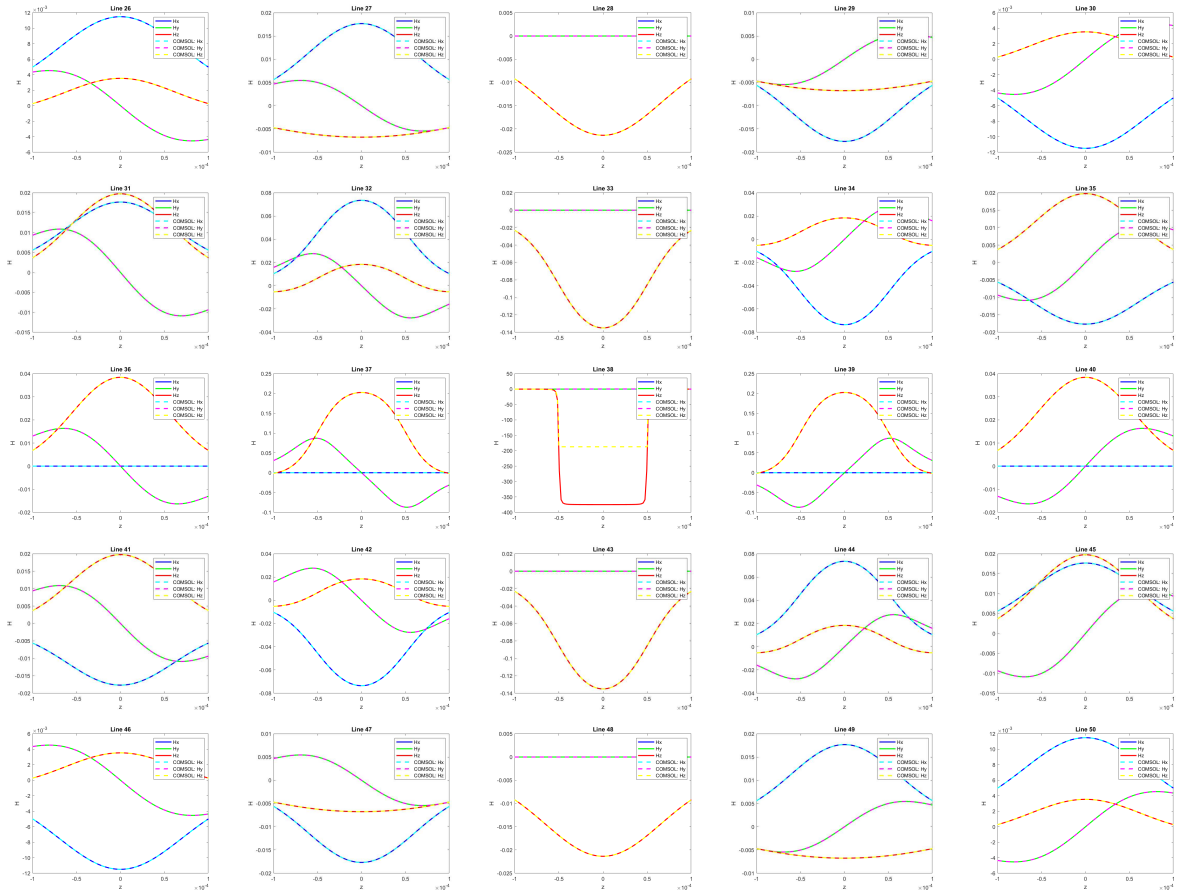


Figure C.3: Comparison of the two models over the constructed lines in y direction

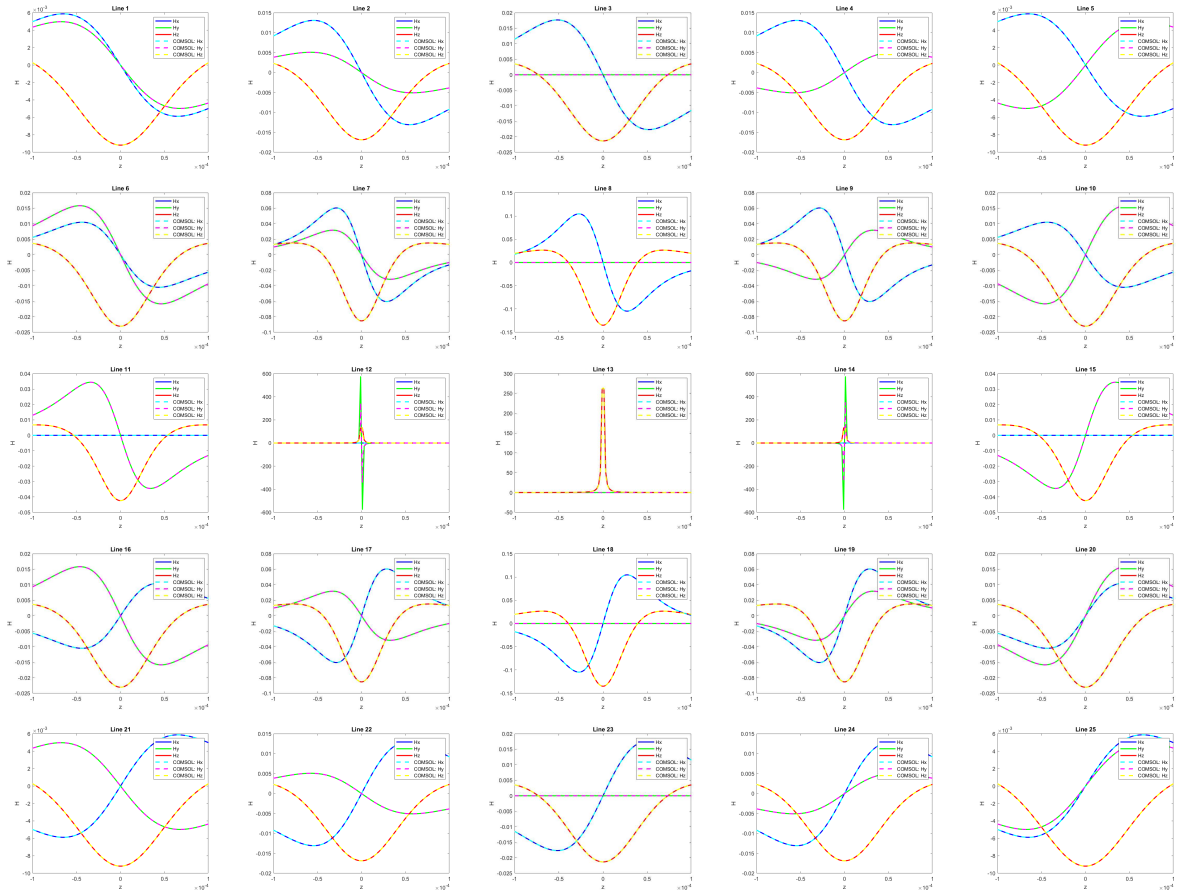
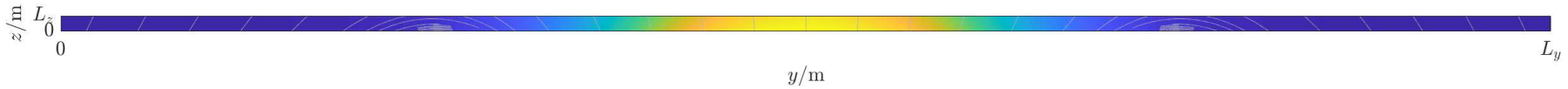


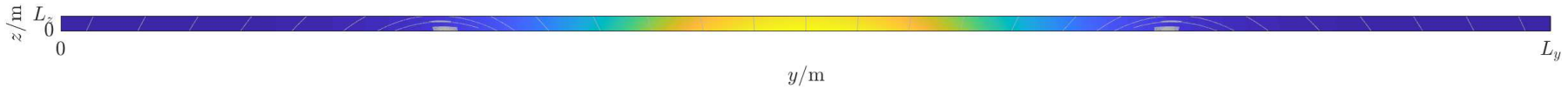
Figure C.4: Comparison of the two models over the constructed lines in z direction

C.4. Magnetic fields of the bar magnets

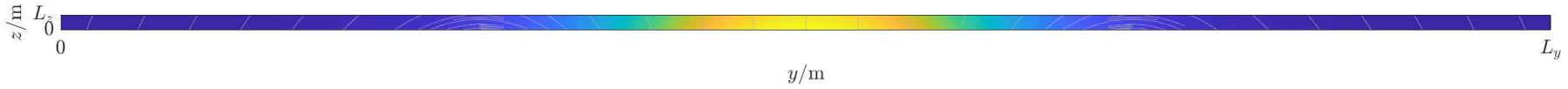
C.4.1. Bar magnet 1



C.4.2. Bar magnet 2



C.4.3. Bar magnet 3



D

MATLAB model

A model to simulate the behavior of an MRF was first made in MATLAB. This allowed to quickly check if the particles behaved as they should. A quick overview of the model will be given.

The model uses the same forces as described in chapter 2. Furthermore, the lubrication model is added and the collision force is critically damped in order to increase stability and decrease the computational time of the simulation. The contact stiffness of the Hertzian contact model is given by equation (D.1) [26]. It is expected that this does not influence the steady state behavior of the MRF. The model uses the fixed dipole method with cut-off distances and periodic boundary conditions are included.

$$k_{ab}^{\mathsf{c}} = \frac{\mathsf{d}\left\|\boldsymbol{F}_{ab}^{\mathsf{c}}\right\|}{\mathsf{d}\xi} = \frac{E}{(1-\nu^2)}\sqrt{R^{\mathsf{eff}}\xi} \tag{D.1}$$

The ODE is solved using the ode23s solver, as the particles collision results in a stiff system. In order to improve stability of the simulation, at each time step the Jacobian is determined. Even though the ode23s function enables for a variable time step, simulating the MRF using this model proved to be very slow. Therefore, the model was simplified and converted to Fortran.

E

Single core Fortran code

E.1. Source code

E.1.1. Fixed dipole method in a uniform magnetic field

The most basic model is given by the model where the fixed dipole method is used for the magnetic interaction force and the MRF is subjected to a uniform magnetic field. The source code files are given below.

param.f90

```
module mod_param
    !This file is used to load all the parameters

    !Define pi=3.14....
    real(8),parameter :: pi = acos(-1.) !pi

    !General parameter
    real(8),parameter :: mu0 = pi*4.D-07 !Permeabillity of vaccuum
    integer,parameter :: n = 1000 !Number of particles

    !Fluid parameters
    real(8),parameter :: mu_f = mu0, & !Permeabillity
                         rho_f = 1000., & !Density
                         nu_f = 0.2, & !Viscosity
                         sr = 0. !The applied shear rate

    !Particle parameters
    real(8),parameter :: mu_p = 2000.*mu0, & !Permeabillity
                         a = 2.5D-06, & !Radius
                         rho_p = 7.8D3, & !Density
                         E = 500.D6, & !Youngs modulus
                         nu = 0.3, & !Poison ratio
                         Vp = 4./3.*pi*a**3, & !Volume
                         m_p = Vp*rho_p, & !Mass
                         beta = (mu_p/mu0 - mu_f/mu0) / (mu_p/mu0+2.*mu_f/mu0) !Contrast factor

    !Simulation domain
    real(8),parameter :: Srz = 75.D-06, & !Film height, z-dimension
                         S_frac = 1.5, & !y-dimension = 1.5*x-dimension
                         vfrac = 0.2, & !Volume fraction
                         Srx = sqrt(n*Vp/(vfrac*Srz*(S_frac))), & !x-dimension
                         Sry = S_frac*Srx !y-dimension
    real(8),parameter,dimension(3) :: Sredge = (/Srx, Sry, Srz/) !Edges of the simulation domain

    !Magnetic field parameters
    real(8),parameter,dimension(3) :: H0 = (/0.,0.,0.1/)/mu0 !Uniform magnetic field
    real(8),parameter,dimension(3) :: mag0_p = 4.*pi*a**3*beta*H0 !Magnetic moment

    !Integration parameters
    real(8),parameter :: tstart = 0., & !Start time of the simulation
                         tfinal = 7.5D-3, & !End time of the simulation
                         dt = 5.D-8, & !Time step
                         lambda = 0.65 !Emperical parameter fo rthe modified velocity-Verlet algorithm
    integer,parameter :: steps = nint((tfinal-tstart)/dt),  & !Number of steps
                         runs = 1, & !Number of runs
                         N_frames = 60, & !Number of frames (positions + velocities) to save
                         intsave=nint(real(steps)/real(N_frames)) !Interval for which to save the frames
```

```
48
49      !Wall parameters
50      real(8),parameter,dimension(2) :: Pw = (/0.D0, Srz/), & !Positions of the walls
51                                        Nw = (/1.,-1./) !Normal vector from the wall into the fluid
52  end module
```

init_part.f90 [2]

```
1   module mod_initpart
2       !This file is used to generate the particles in the simulation domain
3
4       use mod_param !Load the parameters
5       implicit none
6       public init_particles !Make sure the subroutine init_particles can be found
7
8   contains
9       subroutine init_particles(coords) !Subroutine to generate particles
10          integer :: values(1:8),nseed !Random number generation variables
11          integer :: p,pp,i,j !Used to loop over the particles and the walls
12          integer, dimension(:), allocatable :: seed !The seed
13          real(8),dimension(n,3),intent(out) :: coords !Coordinates of the particles, only as output
14          real(8) :: rn, & !Normal distance between particles
15                     rx, & !x-distance between particles
16                     ry, & !y-distance between particles
17                     rz    !z-distance between particles
18
19          !Generate random seed
20          call date_and_time(VALUES = values)
21          call random_seed(size=nseed)
22          allocate(seed(1:nseed))
23          seed(:) = values(8)
24          call random_seed(put=seed)
25
26          !Generation of particles
27          do p = 1,n !Loop over the particles
28              !Generate a random x,y and z position
29  111         call random_number(rx)
30              call random_number(ry)
31              call random_number(rz)
32              coords(p,1) = Srx*rx
33              coords(p,2) = Sry*ry
34              coords(p,3) = Srz/2.D0 + 0.99D0*(2.D0*(rz-0.5D0))*((Srz/2.D0) - a)
35
36              !First conditions for which the particle is accepted,
37              !If the particle is not accepted, the positions are regenerated
38              if ((coords(p,1) < 0.D0) .or. (coords(p,1) > Srx  ) .or. &
39                      (coords(p,2) < 0.D0) .or. (coords(p,2) > Sry  ) .or. &
40                      (coords(p,3) < a ) .or. (coords(p,3) > Srz-a)) then
41                  goto 111
42              end if
43
44              !Make sure that particles do not overlap, otherwise the particle is regenerated
45              do pp=1,p-1 !Loop over the other generated particles
46                  do i=-1,1 !Periodic boundary conditions in x direction
47                      do j=-1,1 !Periodic boundary conditions in x direction
48                          !Distance between the particles
49                          rn = sqrt((coords(p,1)-coords(pp,1)+i*Srx)**2 + &
50                              (coords(p,2)-coords(pp,2)+j*Sry)**2 + &
51                              (coords(p,3)-coords(pp,3))**2)
52                          if (rn< 1.025D0*2.D0*a) then
53                              goto 111
54                          endif
55                      enddo
56                  enddo
57              enddo
58          enddo
59      end subroutine
60  end module
```

pbc2d.f90

```
1   module mod_pbc2d
2       !This file is used to determine the position of the particles given the PBCs
3
4       use mod_param !Load the parameters
5
6       implicit none
7       public pbc2dr, pbc2d !Make sure the subroutines can be found
8   contains
9       subroutine pbc2dr(r) !subroutine to determine the distance between the
10                           !a particle and the correct image of a particle
11
12          integer :: i !Used to loop over the 2 dimensions of the PBC
13          real(8),dimension(3):: r !Distance vector between 2 particles
```

```

        do i=1,2 ! Loop over the PBC dimensions
            !Minimum image convention
            if (r(i) > Sredge(i)/2.D0) then
                r(i) = r(i) - Sredge(i)
            elseif (r(i) < -Sredge(i)/2.D0) then
                r(i) = r(i) + Sredge(i)
            endif
        enddo
    end subroutine

    subroutine pbc2d(r) !Subroutine to determine the position of the particles,
                        !given the PBCs

        integer :: i !Used to loop over the 2 dimensions of the PBC
        real(8),dimension(3):: r !Distance vector between 2 particles

        do i=1,2 ! Loop over the PBC dimensions
            !Check if a particle has crossed a boundary and correct its position
            if (r(i) > Sredge(i)) then
                r(i) = r(i) - Sredge(i)
            elseif (r(i) < 0.D0) then
                r(i) = r(i) + Sredge(i)
            endif
        enddo
    end subroutine
end module
```

force.f90

```
module mod_force
    !This file is used to determine the forces and accelerations acting on the particles

    use mod_param !Load the parameters
    use mod_pbc2d !Load the file used for PBCs

    implicit none
    public force !Make sure the subroutines can be found

contains
    subroutine force(coords,vel,acc,tauint,tauext) !subroutine to determine the forces and accelerations

        integer :: p,pp,i !Used to loop over the particles and walls
        real(8),dimension(n,3),intent(in) :: coords,vel !Input arrays for the positions and velocities
        real(8),dimension(n,3),intent(out) :: acc !Output array for the acceleration
        real(8),intent(out):: tauint, tauext !Output values for the shear stresses
        real(8),dimension(n,3) :: Ft = 0. !Total force matrix
        real(8),dimension(n,3) :: mag_p !Magnetic moment of each particle
        real(8),dimension(3) :: Fm, Fc, Fd,r,ru !Forces and distance vectors
        real(8) :: Ftauint, Ftauext !Force vectors used to determine the shear stresses
        real(8) :: rn,epsc !Distance and overlap of particles

        !Reset the total force
        Ft = 0.

        !Reset the shear stresses
        tauint = 0.
        tauext = 0.

        !Set the magnetization
        do p =1,n
            mag_p(p,:) = mag0_p
        enddo

        do p=1,n !Loop over the particles to determine the forces

            !Reset the external shear force
            Ftauext = 0.

            do pp=p+1,n !Loop over the other particles

                !Reset the force between the particles in y direction
                Ftauint = 0.

                !Distance between the particles
                r = coords(p,:) - coords(pp,:)

                !Fix the distance according to the PBCs
                call pbc2dr(r)

                !Norm of the vector r
                rn = norm2(r)

```

```
54                !Unit vector
55                ru = r/rn
56
57                !Magnetic force
58                if (rn<6.*a) then !Make sure that the particles are within cut-off distance
59
60                    !Magnetic force acting on p due to pp
61                    Fm = 3.*mu0/(4.*pi*rn**4) * &
62                    (dot_product(mag_p(p,:),mag_p(pp,:))*ru + &
63                     dot_product(mag_p(pp,:),ru)*mag_p(p,:)  + &
64                     dot_product(mag_p(p,:) ,ru)*mag_p(pp,:) - &
65                    5.*dot_product(mag_p(p,:),ru)*dot_product(mag_p(pp,:),ru)*ru)
66
67                    !Addition to the total force
68                    Ft(p,:)  = Ft(p,:)  + Fm
69                    Ft(pp,:) = Ft(pp,:) - Fm
70
71                    !Addition to the internal shear stress force
72                    Ftauint = Fm(2)
73
74                endif
75
76                !Contact force
77                if (rn<2.*a) then !Make sure the particles overlap
78                    !Overlap
79                    epsc = 2.*a-rn
80
81                    !Contact force on p due to pp
82                    Fc = (2./3.)*E/(1.-nu**2)*sqrt(a/2.)*epsc**(3./2.)*ru
83
84                    !Addition to the total force
85                    Ft(p,:)  = Ft(p,:)  + Fc
86                    Ft(pp,:) = Ft(pp,:) - Fc
87
88                    !Addition to the internal shear stress force
89                    Ftauint = Ftauint + Fc(2)
90                endif
91
92                !Determine the internal shear stress
93                tauint = tauint - 1./(Srx*Sry*Srz)*r(3)*Ftauint
94            enddo
95
96            !Wall forces
97            do i =1,2 !Loop over the walls
98
99                !Vector between particle and wall
100               r = coords(p,:) - (/coords(p,1), coords(p,2), Pw(i)/)
101
102               !Distance between particle and wall
103               rn = norm2(r)
104
105               if (rn<a) then !If the wall and particle overlap
106                   !Overlap wall and particle
107                   epsc = a-rn
108
109                   !Contact force from the wall on particle
110                   Fc = (2./3.)*E/(1.-nu**2)*sqrt(a)*epsc**(3./2.)*(/0.D0,0.D0,Nw(i)/)
111
112                   !Add it to the total force
113                   Ft(p,:) = Ft(p,:) + Fc
114
115                   !Addition to the external stress force
116                   Ftauext = Fc(2)
117               endif
118           enddo
119
120           !Fluid drag - couette flow
121           Fd = 6.*pi*a*nu_f*((/0.D0, sr*coords(p,3), 0.D0/) - vel(p,:))
122
123           !Addition to the total force
124           Ft(p,:) = Ft(p,:) + Fd
125
126           !Addition to the external shear stress force
127           Ftauext = Ftauext + Fd(2)
128
129           !Determine the external shear stress
130           tauext = tauext + 1./(Srx*Sry*Srz)*coords(p,3)*Ftauext
131       enddo
132
133       !The accleration of all particles
134       acc = Ft/m_p
135
136   end subroutine
137end module
```

main.f90

```
program MRFSIM
    !This is the main file of the model used to simulate a MRF

    use mod_param !Load the parameters
    use mod_pbc2d !Load the file used for PBCs
    use mod_initpart !Load the file to generate particles
    use mod_force !Load the file to determine the acceleration of particles

    implicit none

    real(8),dimension(n,3) :: coords,vel, velp, acc = 0.0 !Arrays used for the position,
                                                          !(Prediction) velocity and acceleration
    real(8),dimension(0:steps) :: time,test,t !Arrays to track the time spent, the estimated time left,
                                              !and the time in the simulation
    real(8),dimension(n) :: Ek !Used to track the kinetic energy of the particles
    real(8) :: startsim, finishsim, start,finish1,finish2,perc !Used to time and track the simulation
    real(8) :: tauint,tauext !Internal and external shear stresses
    integer :: step,p,run,frame !Used for the steps, particles, runs and the saved frames
    character (len=255) :: file_Ek, file_Ekmax, file_tau, file_meanz,file_frame,dir_frame !Data headers
    character (len=26),dimension(runs) :: header_Ek,header_Ekmax, header_tauint,header_tauext, &
                                          header_meanz !Data headers for each run
    real(8),dimension(runs) :: tc,Ekc,Ekmaxc,tauintc,tauextc,meanzc !Used to put the data of all runs
                                                                    !into one file
    character(len = 255) :: dirname,cwd !Directory name, currecnt directory

    !Open a output file and write some parameters to it
    open(unit=1,file='output.txt',status='replace')
    write(1,*) 'Particle diameter', 2.*a
    write(1,*) 'Simulation region', Srx, Sry, Srz
    write(1,*) 'Shear rate', sr

    !Start a timer for the simulation
    call cpu_time(startsim)

    do run=1,runs !Loop over the number of runs

        !Reset the frames, time, estimated time, in simulation time
        frame = 0
        time = 0.
        test = 0.
        t    = 0.

        !Generate particles
        call init_particles(coords)

        !Reset velocity
        vel =0.

        !Determine kinetic energy
        do p=1,n
            Ek(p) = 0.5D0*m_p*norm2(vel(p,:))**2
        enddo

        !Open dat file and write the total kinetic energy
        write(file_Ek,"('Ek',i0,'.dat')") run
        open(unit=2,file=trim(file_Ek))
        write(2,*) t(0), sum(Ek)


        !Open dat file and write the maximum kinetic energy
        write(file_Ekmax,"('Ekmax',i0,'.dat')") run
        open(unit=3,file=trim(file_Ekmax))
        write(3,*) t(0), maxval(Ek)

        !Determine the initial accelerations
        call force(coords,vel,acc,tauint,tauext)

        !Open dat file and write the shear stresses
        write(file_tau,"('tau',i0,'.dat')") run
        open(unit=4,file=trim(file_tau))
        write(4,*) t(0), tauint, tauext

        !Open dat file and write the mean z-height
        write(file_meanz,"('meanz',i0,'.dat')") run
        open(unit=5,file=trim(file_meanz))
        write(5,*) t(0), sum(coords(:,3))/n

        !Open dat file and write the positions and velocities
        write(dir_frame,"('./out/frames',i0,'/')") run
        write(file_frame,"('frame',i0,'.dat')") frame
        open(unit=6,file=trim(dir_frame)//trim(file_frame))
        do p = 1,n !Loop over the particles
```

```
83              write(6,*) t(0), coords(p,1), coords(p,2), coords(p,3), vel(p,1), vel(p,2), vel(p,3)
84          enddo
85          close(6) !Close this frame dat file
86
87          !update frame counter
88          frame = frame + 1
89
90          !Write the estimation of the time left to the output file
91          write(1,10) "run ",run," of ",runs,": Estimated time left: ",0.,"/ min @ ",0.,"%"
92          10 format(a,I1,a,I1,a,f8.2,a,f6.2,a) !Format for the estimated time left
93
94          ! ========== Main loop =========
95          do step=1,steps
96
97              !Start times
98              call cpu_time(start)
99
100             !Update positions
101             coords = coords + dt*vel + 0.5D0*dt**2*acc
102
103             !Periodic boundary conditions
104             do p=1,n
105                 call pbc2d(coords(p,:))
106             enddo
107
108             !Prediction for the velocity
109             velp = vel + lambda*dt*acc
110
111             !Update velocity with old accelerations
112             vel = vel + 0.5D0*dt*acc
113
114             !Update acceleration and append to data file
115             call force(coords,velp,acc,tauint,tauext)
116
117             !Update velocity with new acceleration
118             vel = vel + 0.5D0*dt*acc
119
120             !Determine kinetic energy
121             do p=1,n
122                 Ek(p) = 0.5D0*m_p*norm2(vel(p,:))**2
123             enddo
124
125             !New time
126             t(step) = t(step-1) + dt
127
128             !Write the total kinetic energy to dat file
129             write(2,*) t(step), sum(Ek)
130
131             !Write the maximum kinetic energy to dat file
132             write(3,*) t(step), maxval(Ek)
133
134             !Write the shear stress to dat file
135             write(4,*) t(step), tauint, tauext
136
137             !Write the mean z-heigth to dat file
138             write(5,*) t(step), sum(coords(:,3))/n
139
140             if (mod(step,intsave) .eq. 0) then !Only save the positions and velocities when
141                                                !this condition holds
142
143                 !Open the right frame file
144                 write(file_frame,"('frame',i0,'.dat')") frame
145                 open(unit=6,file=trim(dir_frame)//trim(file_frame))
146
147                 !Write data to file
148                 do p = 1,n
149                     write(6,*) t(step), coords(p,1), coords(p,2), coords(p,3), vel(p,1), vel(p,2), vel(p,3)
150                 enddo
151
152                 !Close file
153                 close(6)
154
155                 !update frame counter
156                 frame = frame + 1
157             endif
158
159             !Time when verlet is done for this step
160             call cpu_time(finish1)
161
162             !Time elapsed for this verlet-step
163             time(step) = finish1-start
164
165             !Estimation of the time left
166             test(step) = sum(time)/(step-1.D0)*(steps-step)/60.D0
```

```
167
168             !Percentage done
169             perc = real(step)/real(steps)*100.D0
170
171
172             if (mod(step ,intsave) .eq. 0) then !Only write the time left when
173                                                 !this condition holds
174
175                 !Print progress to console
176                 write(1,10) "run ",run," of ",runs,": Estimated time left: ",test(step),"/ min @ ",perc,"%"
177             endif
178
179             !Real time for this step
180             call cpu_time(finish2)
181             time(step) = finish2-start
182
183         enddo
184
185         !Close the data files for this run
186         close(2)
187         close(3)
188         close(4)
189         close(5)
190
191     enddo
192
193     !Time when the main loop stops
194     call cpu_time(finishsim)
195
196     !Write the total simulation time to output file
197     write(1,*) 'Time for ', runs, ' runs is: ', finishsim-startsim
198
199     !Make headers for the total dat files
200     do run=1,runs
201         write(header_Ek(run),"('Ek',i0)") run
202         write(header_Ekmax(run),"('Ekmax',i0)") run
203         write(header_tauint(run),"('tauint',i0)") run
204         write(header_tauext(run),"('tauext',i0)") run
205         write(header_meanz(run),"('meanz',i0)") run
206     enddo
207
208     !Open all the total dat files and write the headers to them
209     open(unit=6+4*(runs-1),file='./out/Ek.dat')
210     write(6+4*(runs-1),*) "          t                              ",header_Ek(1:runs),"average"
211     open(unit=7+4*(runs-1),file='./out/Ekmax.dat')
212     write(7+4*(runs-1),*) "          t                              ",header_Ekmax(1:runs),"average"
213     open(unit=8+4*(runs-1),file='./out/tau.dat')
214     write(8+4*(runs-1),*) "          t                          ",header_tauint(1:runs), "average                    ",&
215      header_tauext(1:runs),"average"
216     open(unit=9+4*(runs-1),file='./out/meanz.dat')
217     write(9+4*(runs-1),*) "          t                              ",header_meanz(1:runs),"average"
218
219     !Combine all the runs into one file
220     do step=1,steps !Loop over all the steps
221         do run=1,runs !Loop over the runs
222
223             !Read the total kinetic energy from this run at this step
224             write(file_Ek,"('Ek',i0,'.dat')") run
225             open(unit=2+4*(run-1),file=trim(file_Ek))
226             read(2+4*(run-1),*) tc(run),Ekc(run)
227
228             !Read the maximum kinetic energy from this run at this step
229             write(file_Ekmax,"('Ekmax',i0,'.dat')") run
230             open(unit=3+4*(run-1),file=trim(file_Ekmax))
231             read(3+4*(run-1),*) tc(run),Ekmaxc(run)
232
233             !Read the shear stresses from this run at this step
234             write(file_tau,"('tau',i0,'.dat')") run
235             open(unit=4+4*(run-1),file=trim(file_tau))
236             read(4+4*(run-1),*) tc(run),tauintc(run), tauextc(run)
237
238             !Read the mean z-height from this run at this step
239             write(file_meanz,"('meanz',i0,'.dat')") run
240             open(unit=5+4*(run-1),file=trim(file_meanz))
241             read(5+4*(run-1),*) tc(run),meanzc(run)
242
243         enddo
244
245         !Write the data of all the runs to the total files
246         write(6+4*(runs-1),*) tc(1),Ekc     ,sum(Ekc)/runs
247         write(7+4*(runs-1),*) tc(1),Ekmaxc  ,maxval(Ekmaxc)
248         write(8+4*(runs-1),*) tc(1),tauintc ,sum(tauintc)/runs, tauextc, sum(tauextc)/runs
249         write(9+4*(runs-1),*) tc(1),meanzc  ,sum(meanzc)/runs
250     enddo
```

```
251
252     !Deleta all the individual run files
253     do run=1,runs
254         close(2+4*(run-1),status='delete')
255         close(3+4*(run-1),status='delete')
256         close(4+4*(run-1),status='delete')
257         close(5+4*(run-1),status='delete')
258     enddo
259 end program
```

E.1.2. Mutual dipole method

The code is transformed to use the mutual dipole method by adding some lines of code to force.f90 from appendix E.1.1.

The following variables need to be added below rule 21.

```
        integer :: k !Used to loop over the itterations of the mutual dipole model
        real(8),dimension(3) :: Hj !Magnetic field generated by all other particles
```

Next to this, after the magnetic moment of each particle is initiated, the mutual dipole method needs to be added. The following lines of code should therefore be placed between rule 33 and 35.

```
        do k=2,4 !Loop over the mutual dipole itteractions
            do p=1,n !Loop over the particles

                !Reset the magnetic field produced by particles
                Hj = (/0.,0.,0./)

                do pp =1,p-1 !Loop over all other particles for which
                             !the magnetic moment is already updated

                    !Distance between the particles
                    r = coords(pp,:) - coords(p,:)

                    !Fix the distance according to the boundary conditions
                    call pbc2dr(r)

                    !Norm of the vector r
                    rn = norm2(r)

                    !Unit vector
                    ru = r/rn

                    !Add the magnetic field produced by this particle to
                    !the total magnetic field produced by the particles
                    Hj = Hj + 1./(4.*pi*rn**3) * (3.*ru* dot_product(mag_p(pp,:),ru) - mag_p(pp,:))
                enddo

                do pp = p+1,n!Loop over all other particles for which
                             !the magnetic moment is not yet updated

                    !Distance between the particles
                    r = coords(pp,:) - coords(p,:)

                    !Fix the distance according to the boundary conditions
                    call pbc2dr(r)

                    !Norm of the vector r
                    rn = norm2(r)

                    !Unit vector
                    ru = r/rn

                    !Add the magnetic field produced by this particle to
                    !the total magnetic field produced by the particles
                    Hj = Hj + 1./(4.*pi*rn**3) * (3.*ru* dot_product(mag_p(pp,:),ru) - mag_p(pp,:))
                enddo

                !Update the magnetic moment of particle p
                mag_p(p,:) = 4.*pi*a**3*beta*(H0 + Hj)
            enddo
        enddo
```

E.1.3. Non-uniform magnetic field

The current model can only be used to determine the behavior of a MRF in a uniform magnetic field. Replacing the uniform field to a non-uniform magnetic field is thus required. Firstly, the file param.f90 is changed to include the parameters of the bar magnet. The following lines of code are used to replace lines 35 and 37.

```
    !Magnet parameters
    real(8),parameter :: w = 2.5D-3, & !0.5 times the x dimension of the magnet
                         h =  2.5D-3, & !0.5 times the y dimension of the magnet
                         l = 1.D-3, & !0.5 times the z dimension of the magnet
                         B_r = 1.45 !remanence of the magnet

    !Displacement of the bar manget, to make sure the center of the magnet is at the
    !correct location
    real(8),dimension(3),parameter :: mdisp = (/-Srx/2., -Sry/2. - h, l + 2.D-3/)
```

The magnetic field generated by the permanent magnet is computed by field.f90.

field.f90

```
module mod_field
    !This file is used to determine the magnetic field and the gradient force at a particles position

    use mod_param !Load the parameter

    implicit none
    public field, Bs, Brs !Make sure the subroutines can be found

contains
    subroutine field !subroutine used to create data for a vector plot of the field

        real(8) :: x,y,z !x,y and positions
        integer :: xs,ys,zs !Used as percentages for the simulation domain
        real(8) :: Brx, Bry, Brz, Brxn, Bryn, Brzn !Magnetic field values

        !Open the data file
        open(2,file='./out/field.out')

        do zs = 0,100 !Loop over the z-dimension
            do ys = 0,100 !Loop over the y-dimension

                        !The x-dimension is not of interest
                        x = 0.D0

                        !Determine the real location based on the percentages
                        y = ys*Sry/100.D0
                        z = zs*Srz/100.D0

                        !Determine the magnetic field at this location (note the displacement)
                        call Brs(x+mdisp(1),y+mdisp(2),z+mdisp(3),Brx,Bry,Brz)

                        !Normalize the vector
                        Brxn = Brx /(sqrt(Brx**2+Bry**2+Brz**2))
                        Bryn = Bry /(sqrt(Brx**2+Bry**2+Brz**2))
                        Brzn = Brz /(sqrt(Brx**2+Bry**2+Brz**2))

                        !Write data to dat file
                        write(2,*) x,y,z, Brx,Bry,Brz, Brxn,Bryn,Brzn
            enddo
        enddo
    end subroutine

    subroutine Bs(x,y,z,Bx,By,Bz) !Model for the non-rotated B field
        implicit none
        real(8),intent(in)  :: x,y,z !The position of interest (input)
        real(8),intent(out) :: Bx,By,Bz !The magnetic field (output)
        real(8) :: I1,I2,I3 !The intergral factors
        real(8) :: m !The magnetization of the magnet

        !Determine Integral factors
        I1 = F1f(-x,y,z)+F1f(-x,y,-z)+F1f(-x,-y,z)+F1f(-x,-y,-z)+F1f(x,y,z)+&
             F1f(x,y,-z)+F1f(x,-y,z)+F1f(x,-y,-z)
        I2 = dlog((F2f(-x,-y,z)*F2f(x,y,z))/(F2f(-x,y,z)*F2f(x,-y,z)))
        I3 = dlog((F3f(-x,y,-z)*F3f(x,y,z))/(F3f(-x,y,z)*F3f(x,y,-z)))

        !Determine the magnetization using the remanence
        m = 1.D0/mu0*B_r

        !Determine the magnetic field at this location
        Bx = mu0*m/(4.D0*pi)*(-I1)
        By = mu0*m/(4.D0*pi)*  I2
        Bz = mu0*m/(4.D0*pi)*  I3
    end subroutine

    subroutine Brs(x,y,z,Brx,Bry,Brz) !Model used to rotate the B field around the y-axis
        implicit none
        real(8),intent(in)  :: x,y,z !The position of interest (input)
        real(8),intent(out) :: Brx,Bry,Brz !The rotated magnetic field (output)
        real(8) :: th = -pi/2.D0 !Rotation angle
```

```
70          real(8) :: xr,yr,zr !rotated positions
71          real(8) :: Bx, By, Bz !The original magnetic field
72
73          !Rotation martix transposed times position
74          xr = dcos(th)*x + 0.D0*y + (-dsin(th))*z
75          yr = 0.D0    *x + 1.D0*y +   0.D0     *z
76          zr = dsin(th)*x + 0.D0*y +   dcos(th) *z
77
78          !The original magnetic field at these positions
79          call Bs(xr,yr,zr,Bx,By,Bz)
80
81          !Rotation of the magnetic field
82          Brx =   dcos(th) *Bx + 0.D0*By + dsin(th)*Bz
83          Bry =   0.D0     *Bx + 1.D0*By + 0.D0    *Bz
84          Brz = (-dsin(th))*Bx + 0.D0*By + dcos(th)*Bz
85      end subroutine
86
87      subroutine magms(x,y,z,magmx,magmy,magmz) !Determines the magnetization of a particle in
88                                                !the rotated B field
89          implicit none
90          real(8),intent(in) :: x,y,z !The position of interest (input)
91          real(8),intent(out) :: magmx, magmy, magmz !The magnetic moment (output)
92          real(8) :: Brx, Bry, Brz !The rotated magnetic field
93
94          !The rotated magnetic field at x,y,z
95          call Brs(x,y,z,Brx,Bry,Brz)
96
97          !The magnetic moment
98          magmx = 4.*pi*a**3*beta*Brx/mu0
99          magmy = 4.*pi*a**3*beta*Bry/mu0
100         magmz = 4.*pi*a**3*beta*Brz/mu0
101     end subroutine
102
103     subroutine gmBs(x,y,z,gmBx,gmBy,gmBz) !Determines the gradient of magm dot B
104         implicit none
105         real(8),intent(in) :: x,y,z !The position of interest (input)
106         real(8),intent(out) :: gmBx, gmBy, gmBz !The gradient force (output)
107         real(8) :: fBrx,fBry,fBrz,bBrx,bBry,bBrz,  &
108                    fmagmx,fmagmy,fmagmz,bmagmx,bmagmy,bmagmz !Variables used to determine the gradient
109         real(8) :: h !step size
110
111         !Step size
112         h = epsilon(h)
113
114         !gradient force in x direction using central difference
115         call   Brs(x+h,y,z,fBrx  ,fBry  ,fBrz)
116         call magms(x+h,y,z,fmagmx,fmagmy,fmagmz)
117         call   Brs(x-h,y,z,bBrx  ,bBry  ,bBrz)
118         call magms(x-h,y,z,bmagmx,bmagmy,bmagmz)
119         gmBx = ((fBrx*fmagmx+fBry*fmagmy+fBrz*fmagmz) - &
120                 (bBrx*bmagmx+bBry*bmagmy+bBrz*bmagmz)) / (2.D0*h)
121
122         !gradient force in y direction using central difference
123         call   Brs(x,y+h,z,fBrx  ,fBry  ,fBrz)
124         call magms(x,y+h,z,fmagmx,fmagmy,fmagmz)
125         call   Brs(x,y-h,z,bBrx  ,bBry  ,bBrz)
126         call magms(x,y-h,z,bmagmx,bmagmy,bmagmz)
127         gmBy = ((fBrx*fmagmx+fBry*fmagmy+fBrz*fmagmz) - &
128                 (bBrx*bmagmx+bBry*bmagmy+bBrz*bmagmz)) / (2.D0*h)
129
130         !gradient force in z direction using central difference
131         call   Brs(x,y,z+h,fBrx  ,fBry  ,fBrz)
132         call magms(x,y,z+h,fmagmx,fmagmy,fmagmz)
133         call   Brs(x,y,z-h,bBrx  ,bBry  ,bBrz)
134         call magms(x,y,z-h,bmagmx,bmagmy,bmagmz)
135         gmBz = ((fBrx*fmagmx+fBry*fmagmy+fBrz*fmagmz) - &
136                 (bBrx*bmagmx+bBry*bmagmy+bBrz*bmagmz)) / (2.D0*h)
137     end subroutine
138
139     function F1f(x,y,z) !Function used in the integral factors
140         implicit none
141         real(8) :: x,y,z !The position of interest
142         real(8) :: F1f !Factor for the magnetic field
143         F1f = datan((h+y)*(w+z)/((l+x)*sqrt((l+x)**2+(h+y)**2+(w+z)**2)))
144         return
145     end function
146
147     function F2f(x,y,z) !Function used in the integral factors
148         implicit none
149         real(8) :: x,y,z !The position of interest
150         real(8) :: F2f !Factor for the magnetic field
151         F2f = (sqrt((l+x)**2 + (h+y)**2 + (w+z)**2)+w+z)/(sqrt((l+x)**2 + (h+y)**2 + (w-z)**2)-w+z)
152         return
153     end function
```

```
154
155     function F3f(x,y,z) !Function used in the integral factors
156         implicit none
157         real(8) :: x,y,z !The position of interest
158         real(8) :: F3f !Factor for the magnetic field
159         F3f = (sqrt((l+x)**2 + (h+y)**2 + (w+z)**2)+h+y)/(sqrt((l+x)**2 + (h-y)**2 + (w+z)**2)-h+y)
160         return
161     end function
162 end module
```

The magnetic moment of a particle and the magnetic gradient force are then used in force.f90. In order to incorporate the following needs to be added between line 5 and 7:

```
    use mod_field !Load the file to determine the magnetic field
```

line 19 should be replaced by:

```
          real(8),dimension(3) :: Fm, Fc, Fd,r,ru, Fme !Forces and distance vectors
```

line 32 needs to be replaced by:

```
          !Determine the magnetization of the particle
          call magms(coords(p,1)+mdisp(1),coords(p,2)+mdisp(2),coords(p,3)+mdisp(3), &
                     mag_p(p,1),mag_p(p,2),mag_p(p,3))
```

and the following should be added between line 127 and 129:

```
          !External magnetic field force
          call gmBs(coords(p,1)+mdisp(1),coords(p,2)+mdisp(2),coords(p,3)+mdisp(3),Fme(1),Fme(2),Fme(3))

          !Addition to the total force
          Ft(p,:) = Ft(p,:) + Fme

          !Addition to the external shear stress force
          Ftauext = Ftauext + Fme(2)
```

In case the mutual dipole method is used, it needs to modified in order to incorporate the non-uniform magnetic field. Firstly, line 18 of the force.f90 should be replaced by:

```
      real(8),dimension(n,3) :: mag_p,mag_pe !(External) magnetic moment of each particle
```

then line 32 needs to be replaced by:

```
          !Determine the magnetization of the particle
          call magms(coords(p,1)+mdisp(1),coords(p,2)+mdisp(2),coords(p,3)+mdisp(3), &
                     mag_pe(p,1),mag_pe(p,2),mag_pe(p,3))
          mag_p(p,:) = mag_pe(p,:)
```

The mutual dipole method as shown below should then be placed between line 33 and 35:

```
      do k=2,4 !Loop over the mutual dipole itteractions
          do p=1,n !Loop over the particles

              !Reset the magnetic field produced by particles
              Hj = (/0.,0.,0./)

              do pp =1,p-1 !Loop over all other particles for which
                           !the magnetic moment is already updated

                  !Distance between the particles
                  r = coords(pp,:) - coords(p,:)

                  !Fix the distance according to the boundary conditions
                  call pbc2dr(r)

                  !Norm of the vector r
                  rn = norm2(r)

                  !Unit vector
                  ru = r/rn

                  !Add the magnetic field produced by this particle to
                  !the total magnetic field produced by the particles
                  Hj = Hj + 1./(4.*pi*rn**3) * (3.*ru* dot_product(mag_p(pp,:),ru) - mag_p(pp,:))
              enddo

              do pp = p+1,n!Loop over all other particles for which
                           !the magnetic moment is not yet updated

                  !Distance between the particles
                  r = coords(pp,:) - coords(p,:)

                  !Fix the distance according to the boundary conditions
```

```
                    call pbc2dr(r)

                    !Norm of the vector r
                    rn = norm2(r)

                    !Unit vector
                    ru = r/rn

                    !Add the magnetic field produced by this particle to
                    !the total magnetic field produced by the particles
                    Hj = Hj + 1./(4.*pi*rn**3) * (3.*ru* dot_product(mag_p(pp,:),ru) - mag_p(pp,:))
                enddo

                !Update the magnetic moment of particle p
                mag_p(p,:) = 4.*pi*a**3*beta*Hj + mag_pe(p,:)
            enddo
        enddo
```

Furthermore, in order to save the field produced by the bar magnet, the following should be added between line 7 and 9 of main.f90:

```
    use mod_field !Load the file to determine the magnetic field
```

and the following lines should be added between line 30 and 32:

```
    !Make the data for the magnetic field
    call field
```

Both single core Fortran models can be altered like this to simulate a MRF in a non-uniform magnetic field generated by a bar magnet.

E.1.4. Compiling and running the code

The Fortran code is compiled using a Makefile, for which GFortran should be installed on your machine. A guide for the installation of GFortran on Windows, Linux, macOS or OpenBSD can be found here [63]. The models are compiled by running the Makefile given below [2]:

Makefile

```
1  FC=gfortran #fortran compiler
2  FFLAGS= -O3 -std=legacy -mcmodel=medium #Activate flags
3  SRC=param.f90 init_part.f90 pbc2d.f90 force.f90 main.f90 #Files to use for compilation (use
       the correct order)
4  OBJ=${SRC:.f90=.o} #substitute .f90 with .o
5  NAME=SCFf
6
7  %.o: %.f90 #wildcard rule, creation of *.o depends on *.f90
8          $(FC) $(FFLAGS) -o $@ -c $<
9
10 $(NAME): $(OBJ)
11         $(FC) $(FFLAGS) -o $@ $(OBJ)
12
13 clean: #cleans all the old compilation files
14         @rm -f *.mod *.o $(NAME)
```

The Makefile is executed using the command:

```
make
```

The compilation is cleared by using the command:

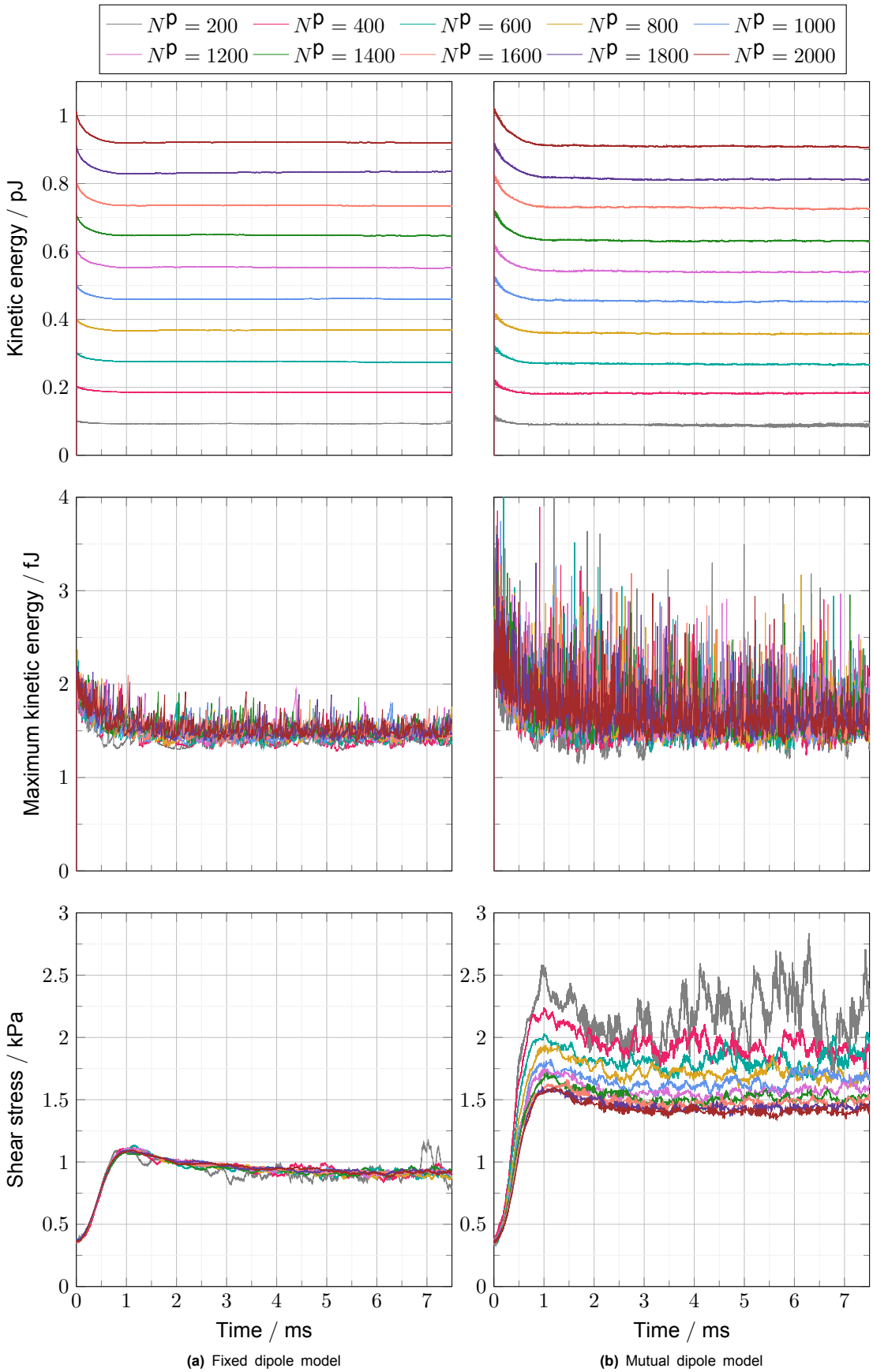
```
make clean
```

It should be noted, that for a non-uniform magnetic field, line 3 should change to:

```
SRC=param.f90 init_part.f90 pbc2d.f90 field.f90 force.f90 main.f90 #Files to use for compilation (use the
    correct order)
```

This Makefile will generate a executable that can be used to run the simulation. In order to make sure that the simulation does not give an error, the directories /out and /out/frames1,/out/frames2,...,/out/frames(runs) should already exist in the working directory.

E.2. Number of particles



(a) Fixed dipole model

(b) Mutual dipole model

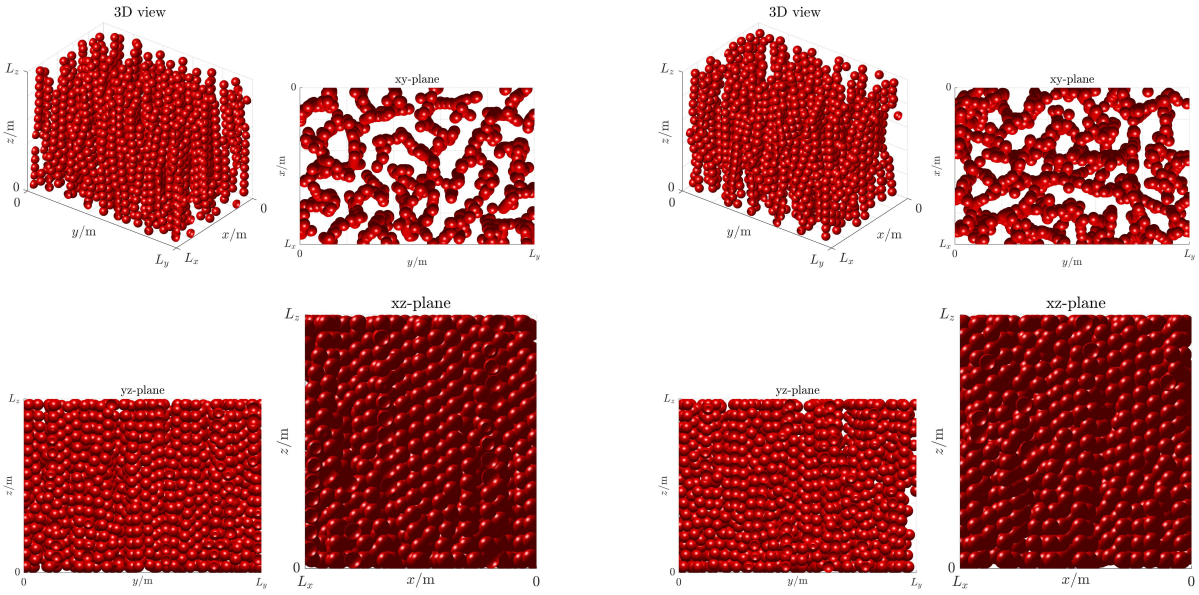
Figure E.1: Mean data over time for different number of particles N^{p} for 3 runs of the Fortran models

E.3. Results

E.3.1. Steady state structures

Uniform magnetic field

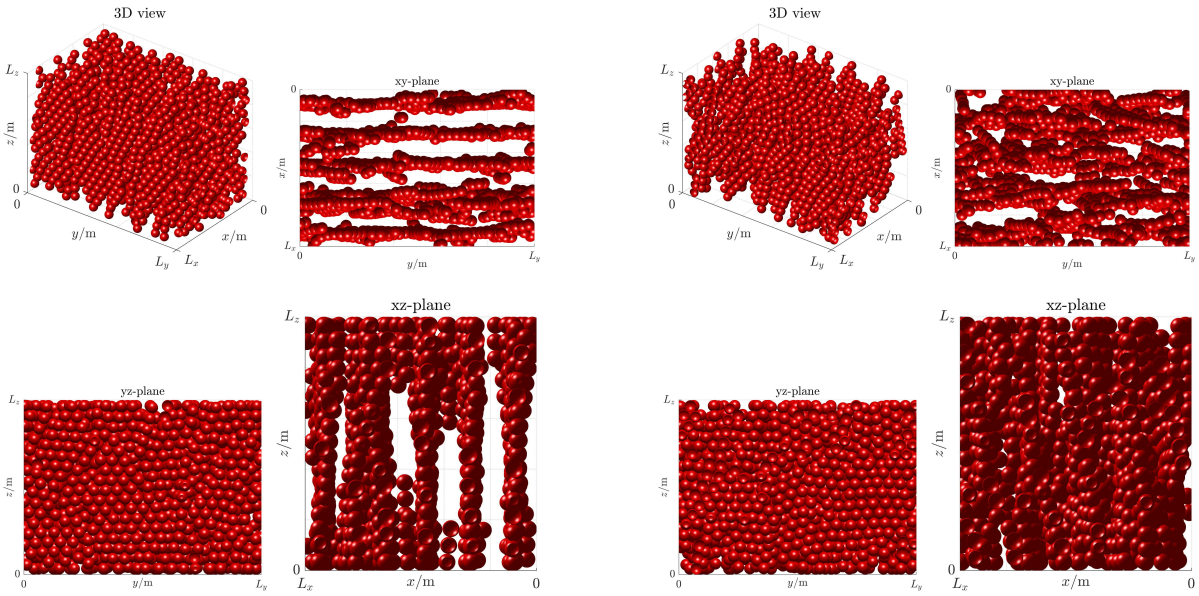
$\dot{\gamma} = 0\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.2: Steady state structures for the single core Fortran models where $\dot{\gamma} = 0\,\mathrm{s}^{-1}$

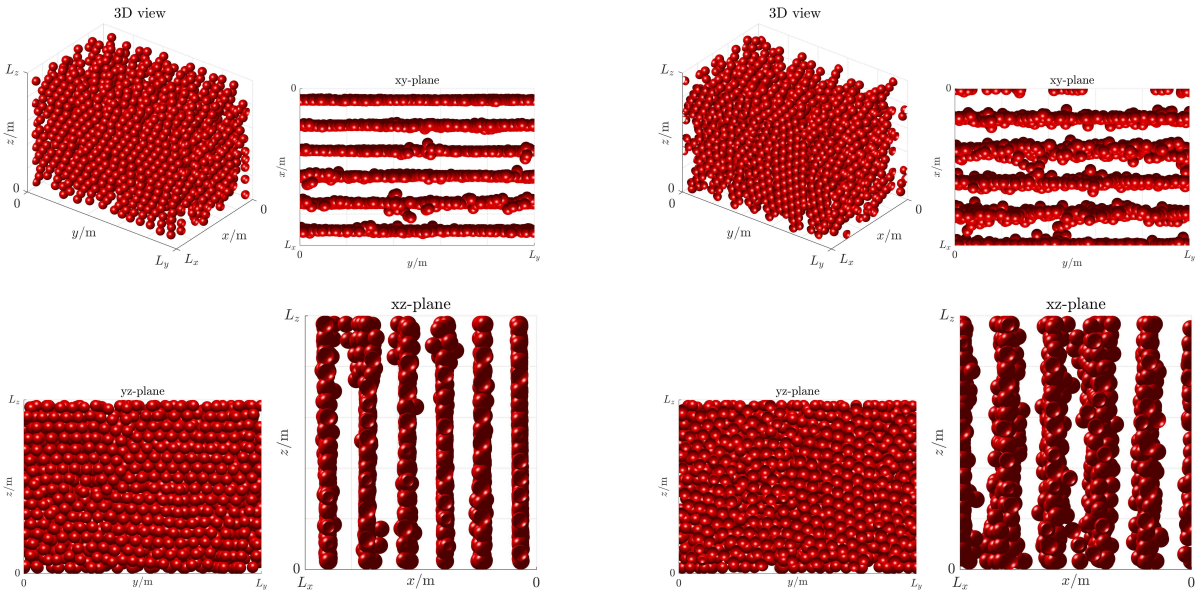
$\dot{\gamma} = 500\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.3: Steady state structures for the single core Fortran models where $\dot{\gamma} = 500\,\mathrm{s}^{-1}$

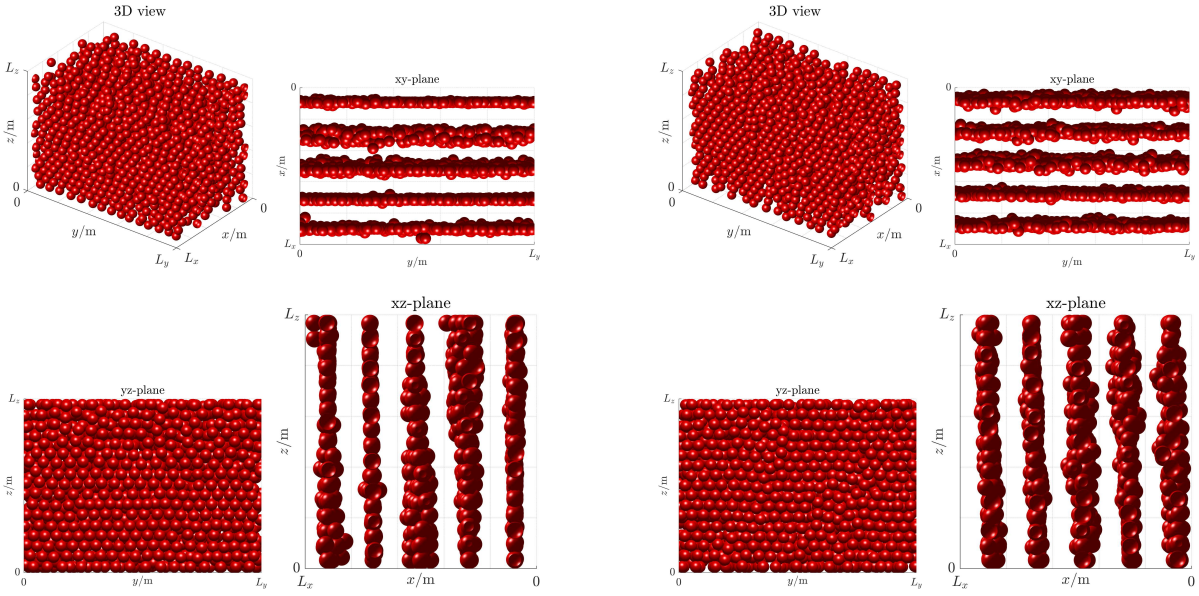
$\dot{\gamma} = 1000\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.4: Steady state structures for the single core Fortran models where $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$

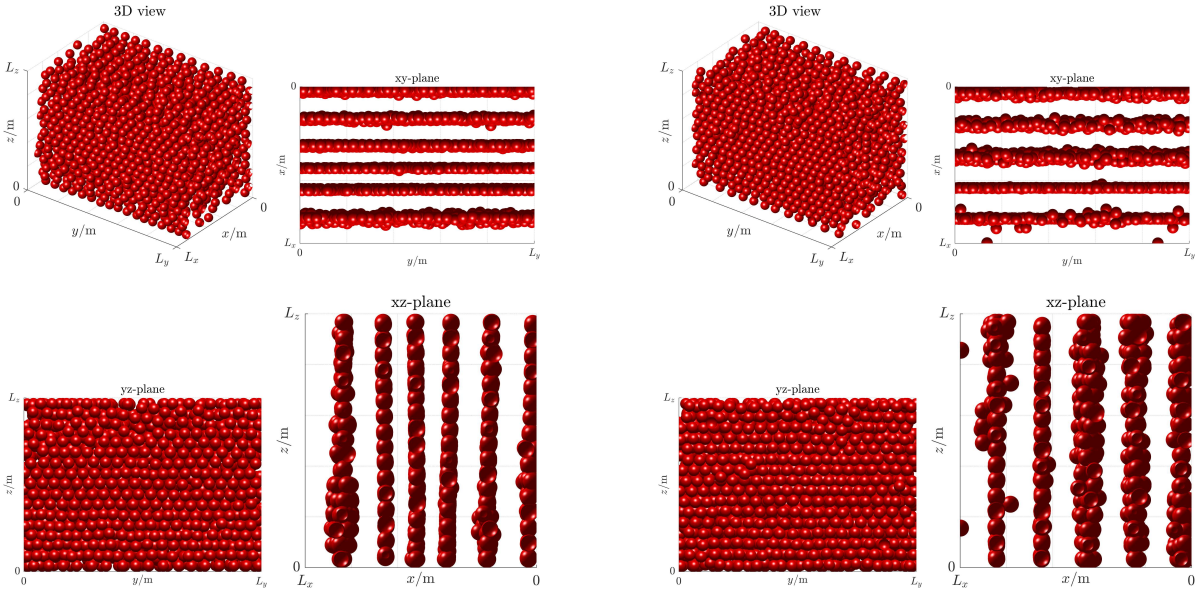
$\dot{\gamma} = 2500\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.5: Steady state structures for the single core Fortran models where $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$

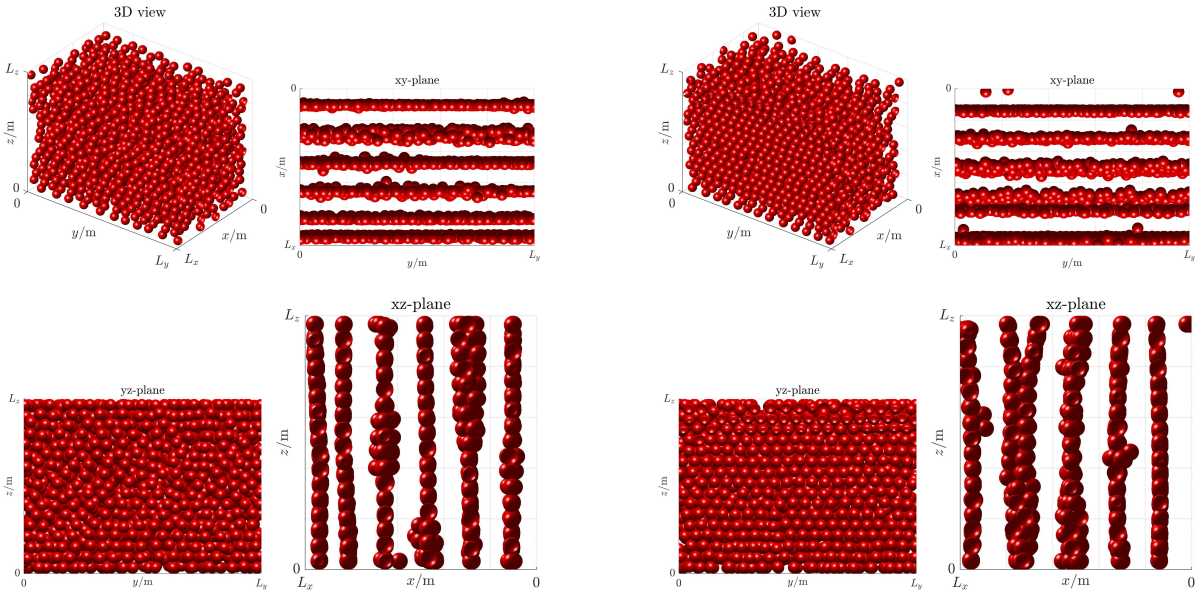
$\dot{\gamma} = 5000\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.6: Steady state structures for the single core Fortran models where $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$

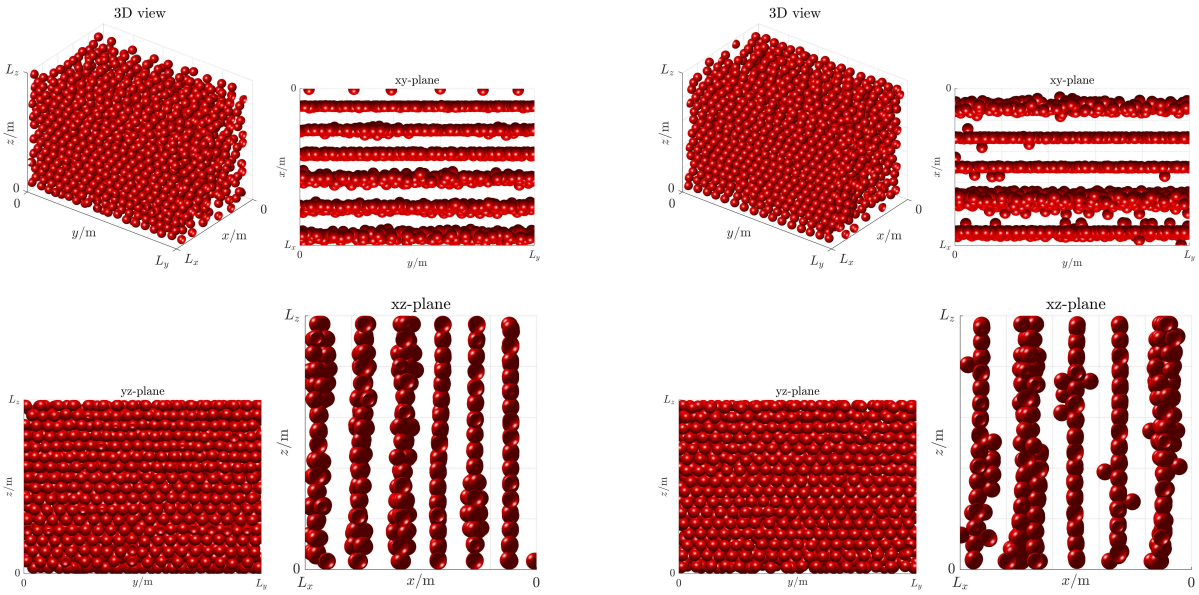
$\dot{\gamma} = 7500\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.7: Steady state structures for the single core Fortran models where $\dot{\gamma} = 7500\,\mathrm{s}^{-1}$

$\dot{\gamma} = 10\,000\,\mathrm{s}^{-1}$



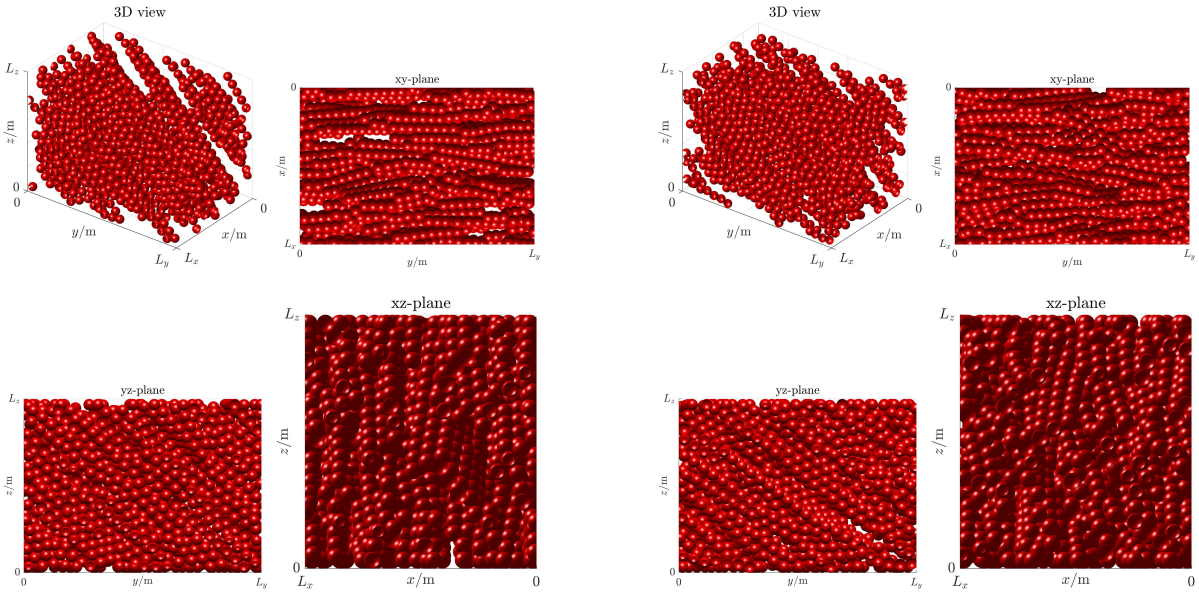
(a) Fixed dipole method

(b) Mutual dipole method

Figure E.8: Steady state structures for the single core Fortran models where $\dot{\gamma} = 10\,000\,\mathrm{s}^{-1}$

Non-uniform: Front of the bar magnet

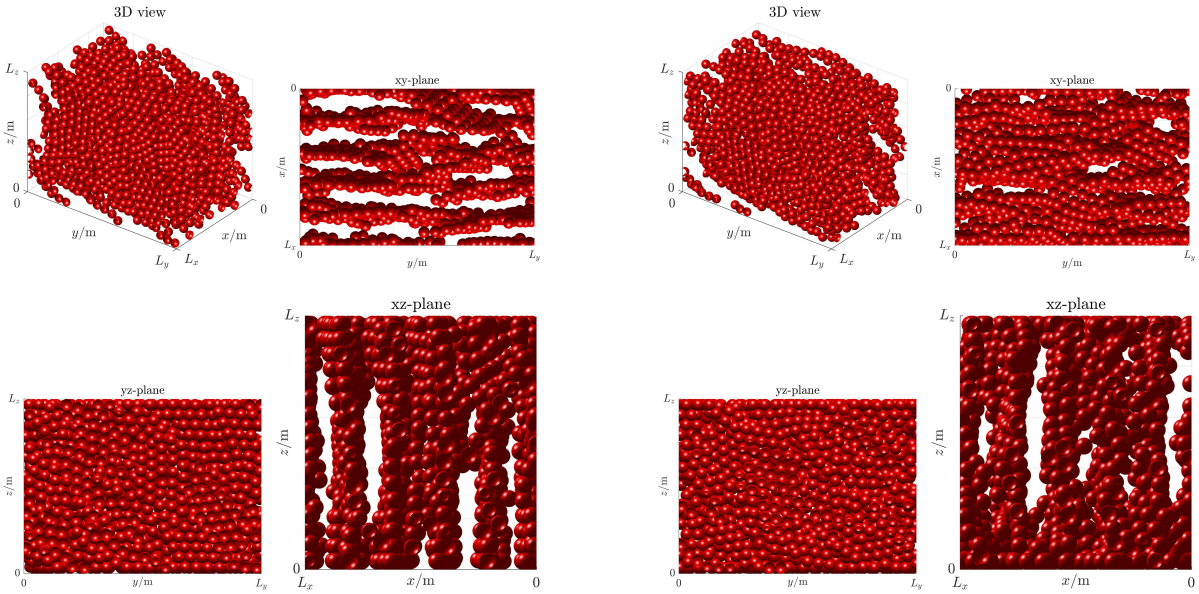
$\dot{\gamma} = 0\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.9: Steady state structures for the single core Fortran models where $\dot{\gamma} = 0\,\mathrm{s}^{-1}$

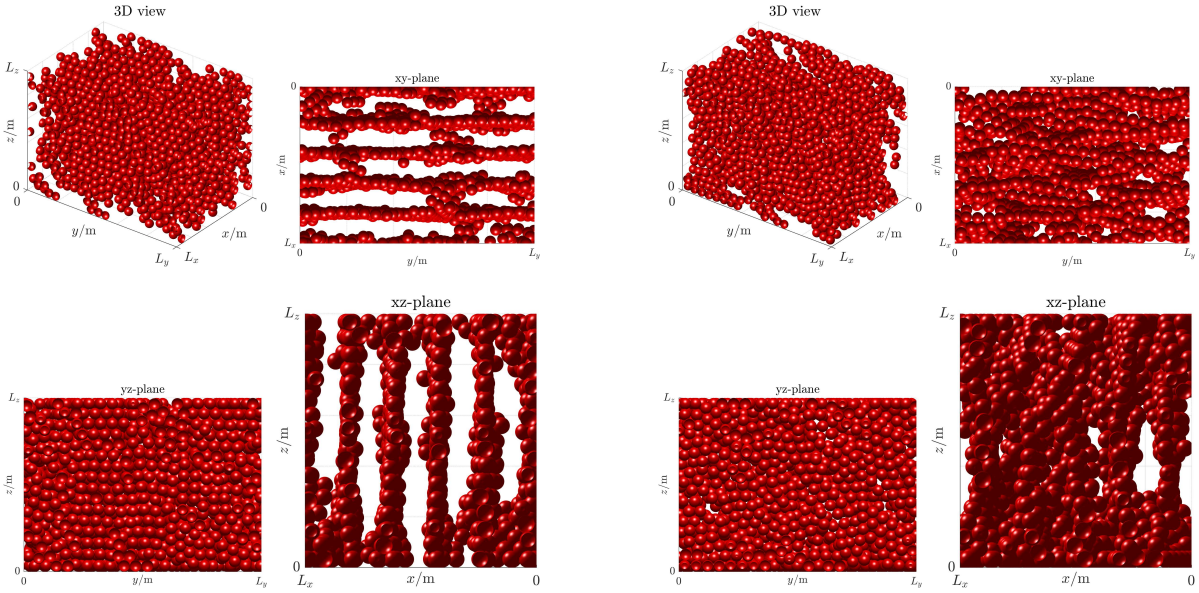
$\dot{\gamma} = 500\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.10: Steady state structures for the single core Fortran models where $\dot{\gamma} = 500\,\mathrm{s}^{-1}$

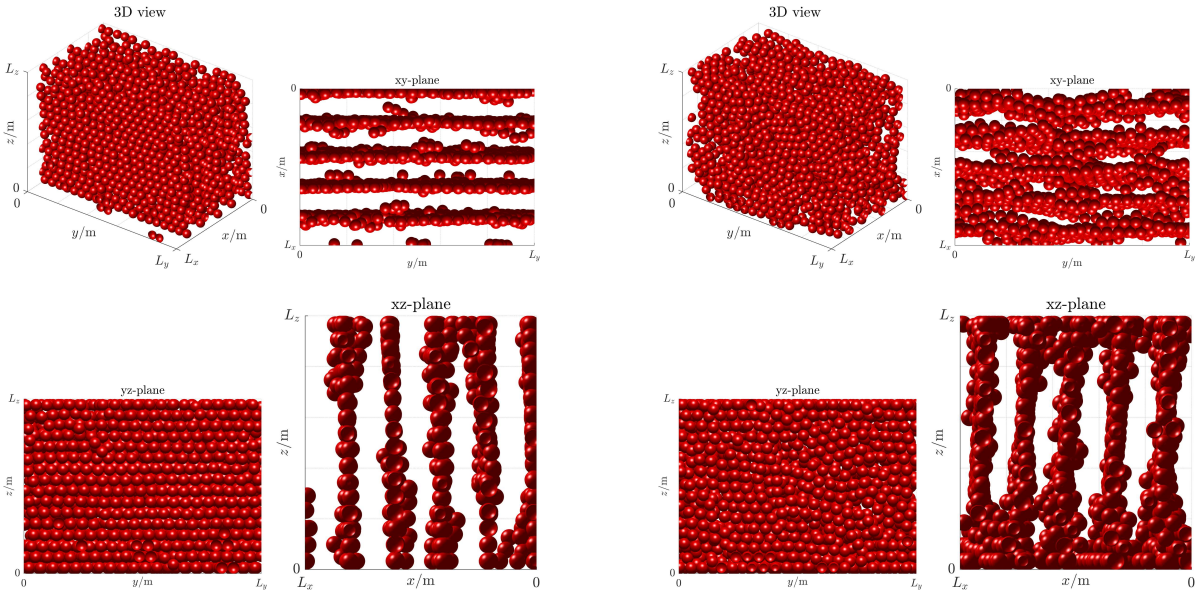
$\dot{\gamma} = 1000\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.11: Steady state structures for the single core Fortran models where $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$

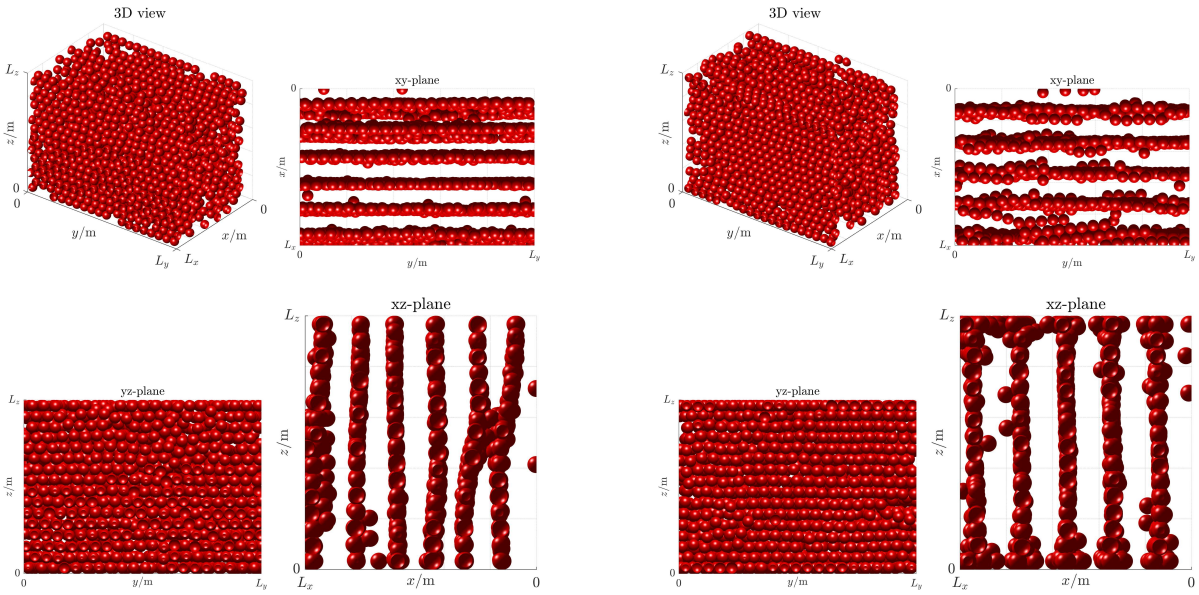
$\dot{\gamma} = 2500\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.12: Steady state structures for the single core Fortran models where $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$

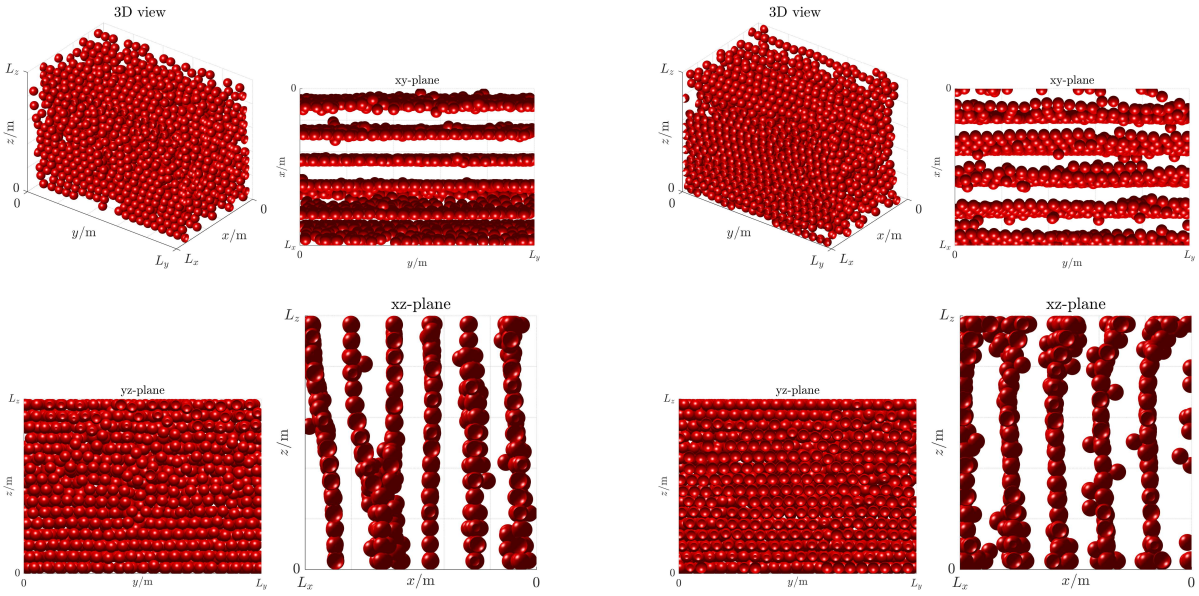
$\dot{\gamma} = 5000\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.13: Steady state structures for the single core Fortran models where $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$

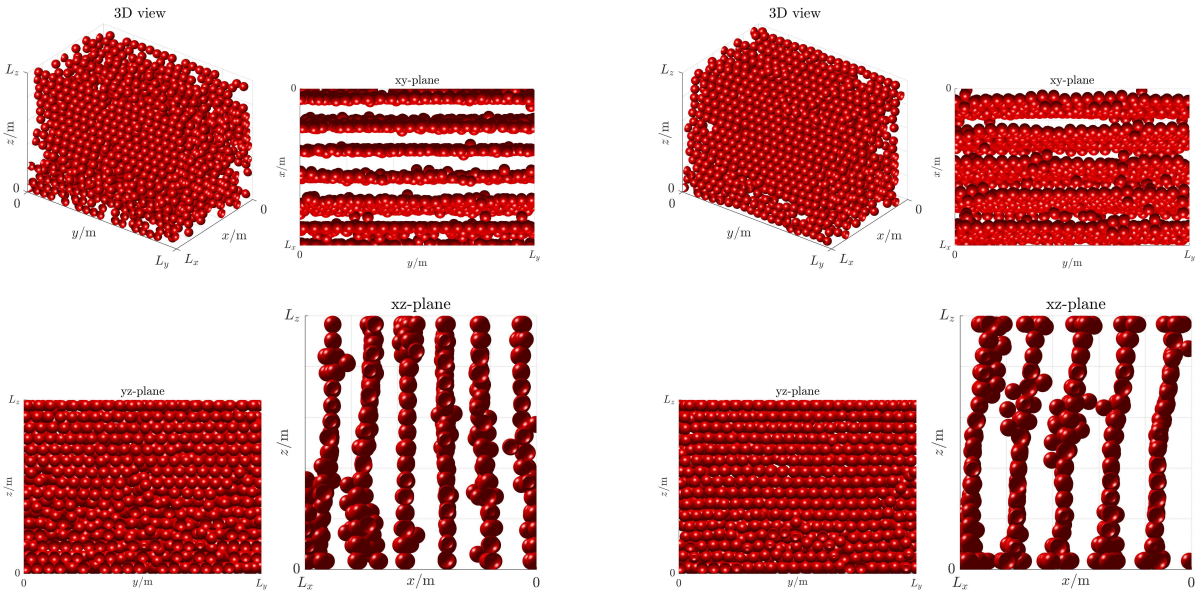
$\dot{\gamma} = 7500\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.14: Steady state structures for the single core Fortran models where $\dot{\gamma} = 7500\,\mathrm{s}^{-1}$

$\dot{\gamma} = 10\,000\,\mathrm{s}^{-1}$



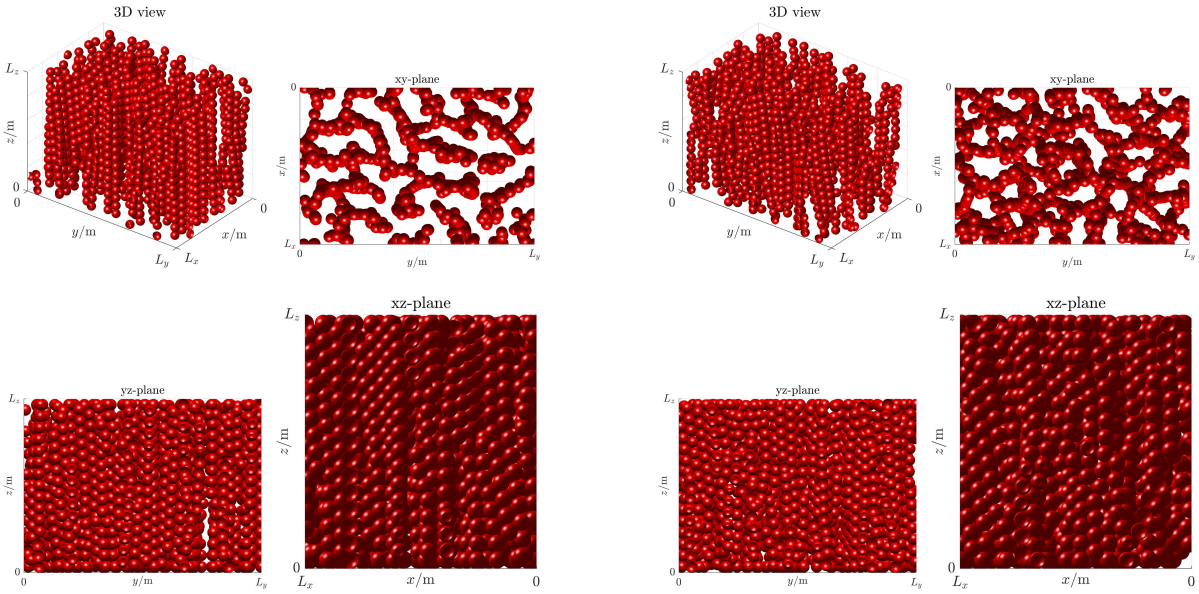
(a) Fixed dipole method

(b) Mutual dipole method

Figure E.15: Steady state structures for the single core Fortran models where $\dot{\gamma} = 10\,000\,\mathrm{s}^{-1}$

Non-uniform: Center of the bar magnet

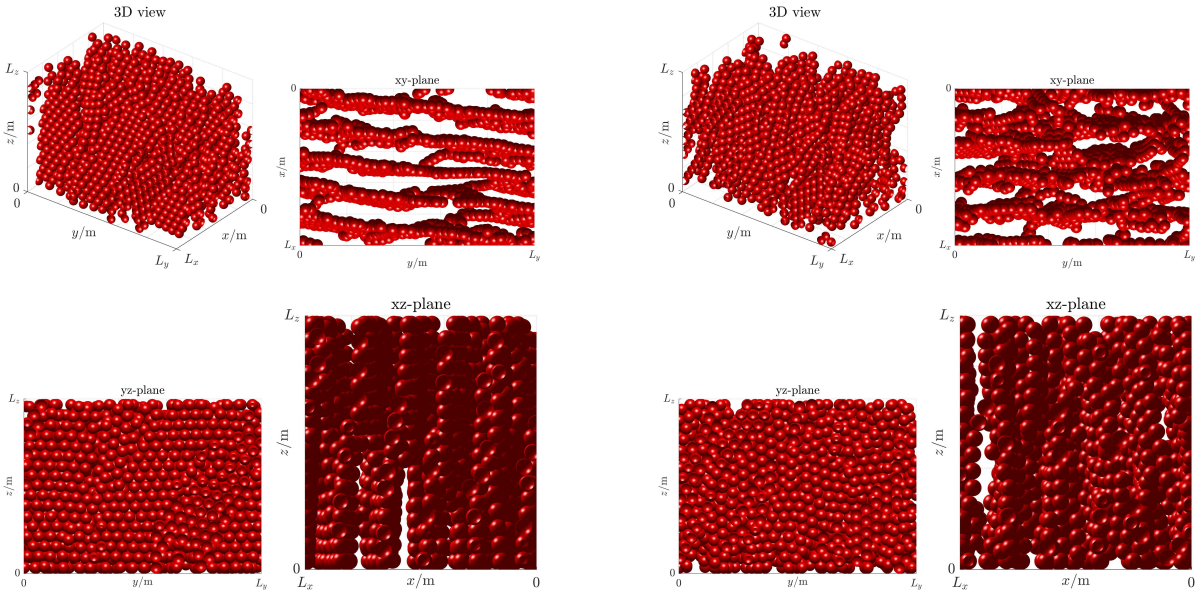
$\dot{\gamma} = 0\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.16: Steady state structures for the single core Fortran models where $\dot{\gamma} = 0\,\mathrm{s}^{-1}$

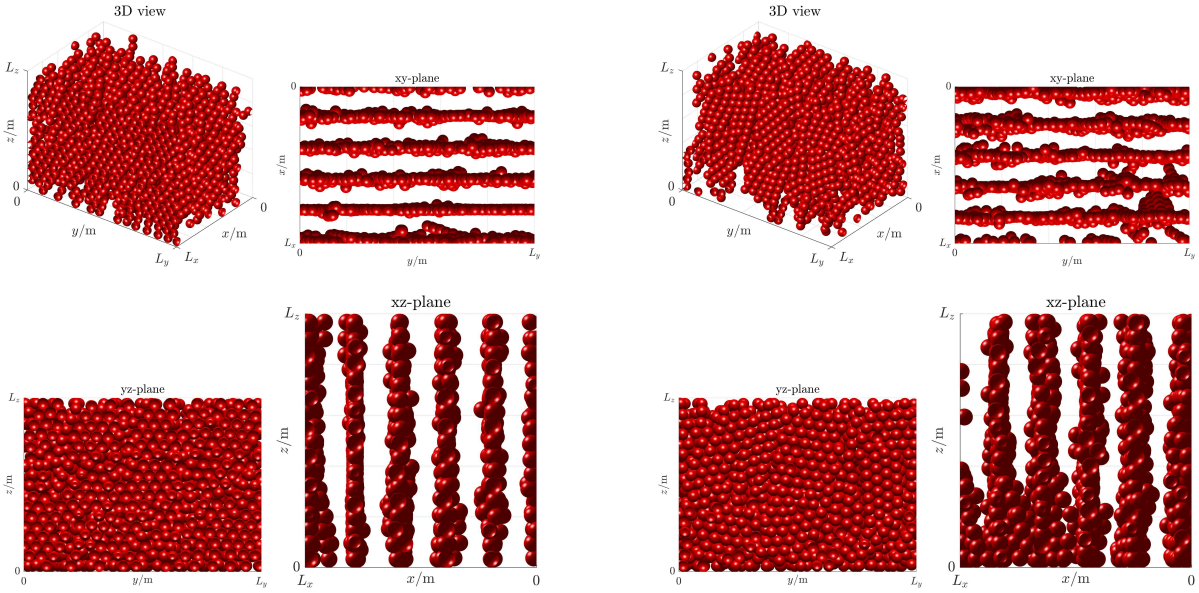
$\dot{\gamma} = 500\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.17: Steady state structures for the single core Fortran models where $\dot{\gamma} = 500\,\mathrm{s}^{-1}$

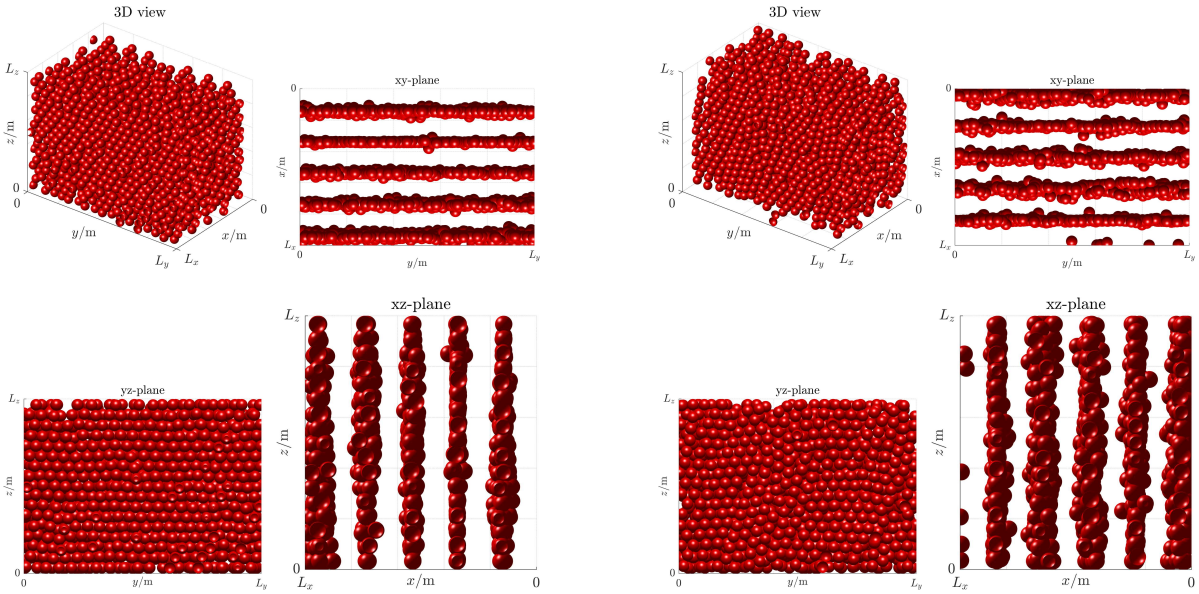
$\dot{\gamma} = 1000\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.18: Steady state structures for the single core Fortran models where $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$

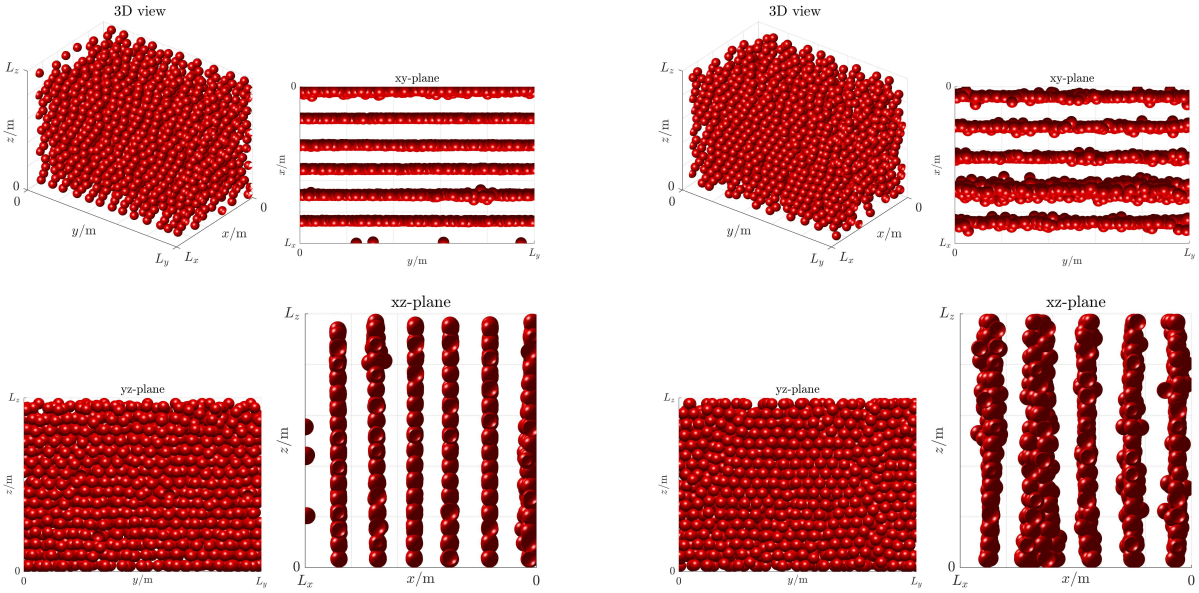
$\dot{\gamma} = 2500\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.19: Steady state structures for the single core Fortran models where $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$

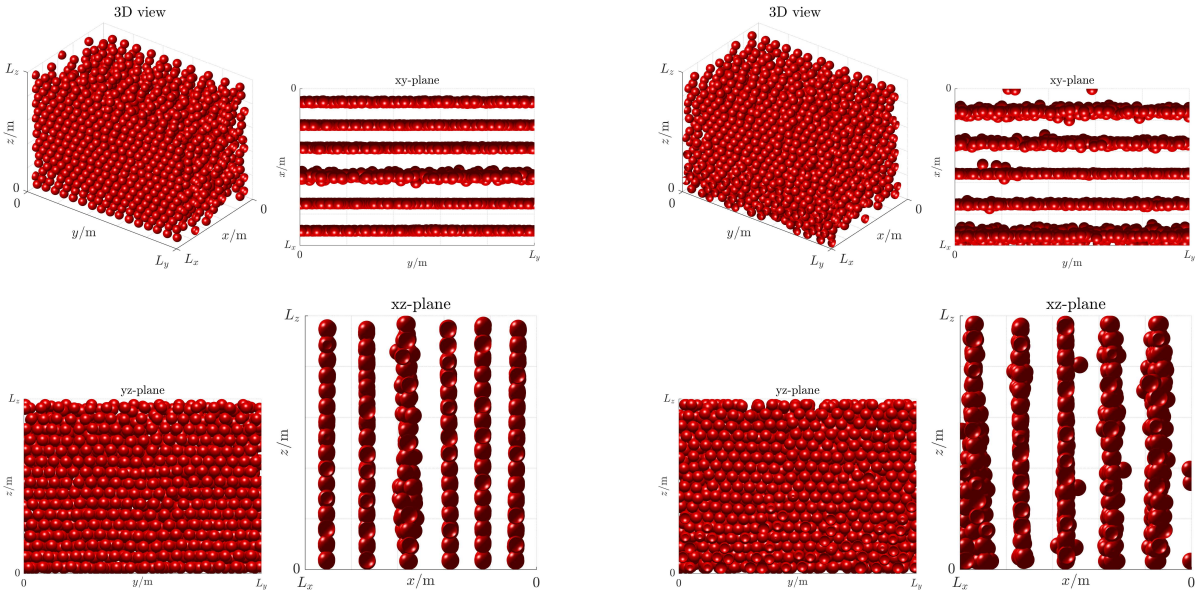
$\dot{\gamma} = 5000\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.20: Steady state structures for the single core Fortran models where $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$

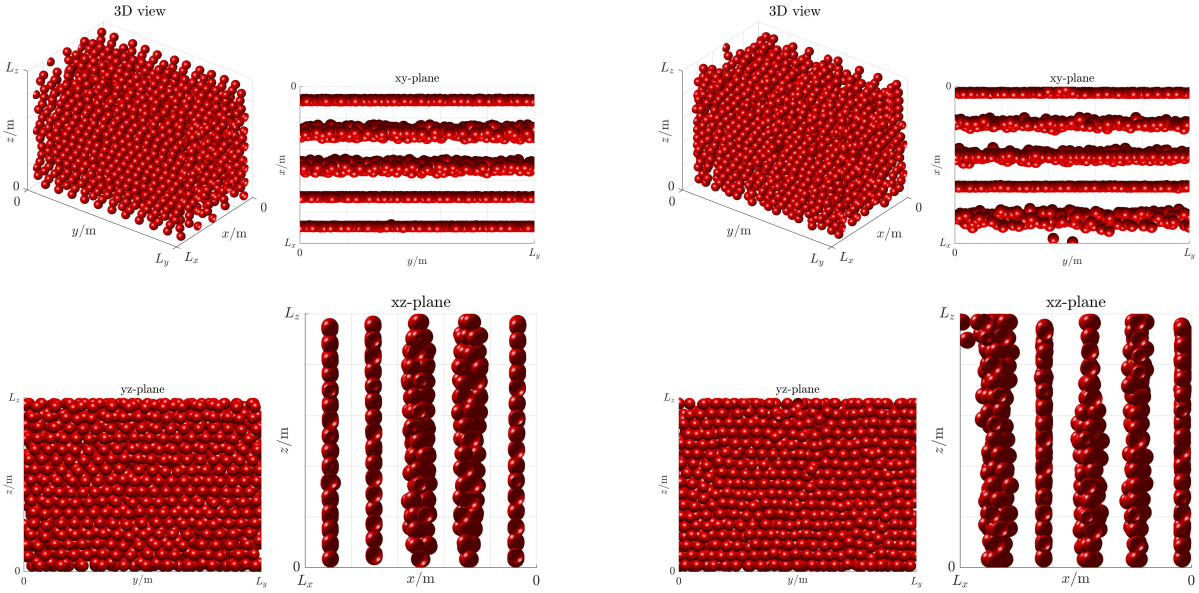
$\dot{\gamma} = 7500\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.21: Steady state structures for the single core Fortran models where $\dot{\gamma} = 7500\,\mathrm{s}^{-1}$

$\dot{\gamma} = 10\,000\,\mathrm{s}^{-1}$

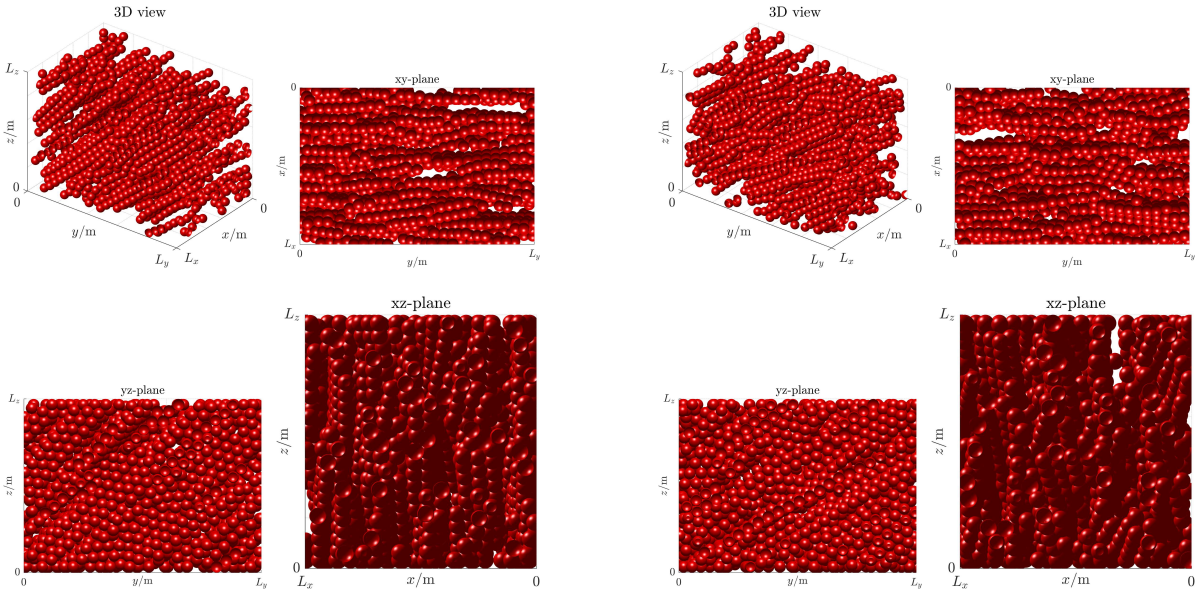


(a) Fixed dipole method

(b) Mutual dipole method

Figure E.22: Steady state structures for the single core Fortran models where $\dot{\gamma} = 10\,000\,\mathrm{s}^{-1}$

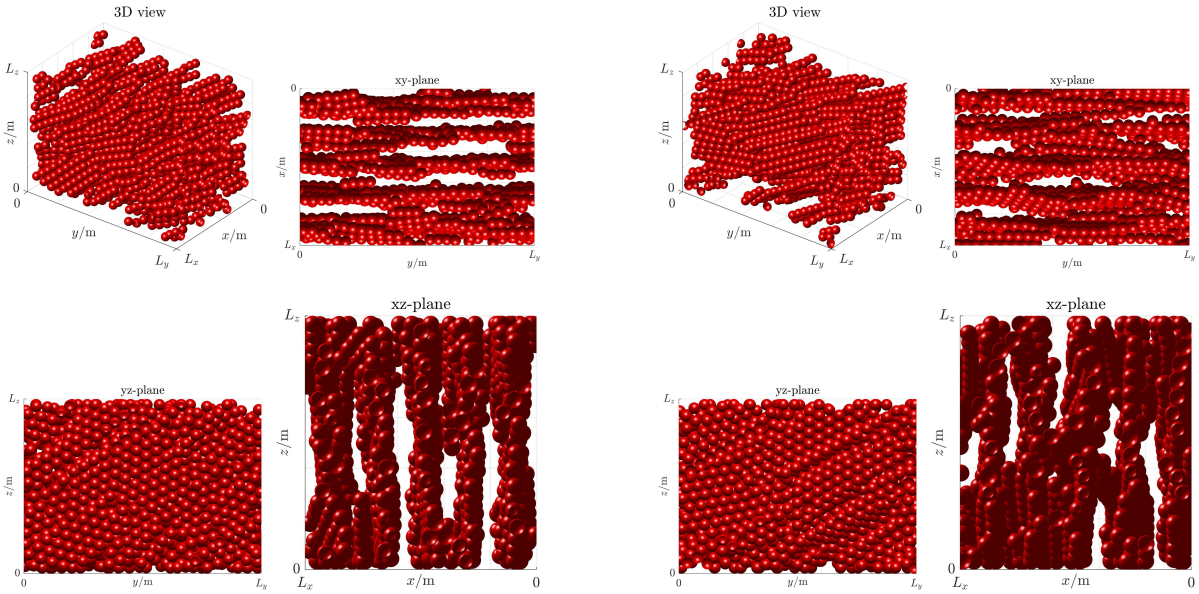
Non-uniform: Back of the bar magnet
$\dot{\gamma} = 0\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.23: Steady state structures for the single core Fortran models where $\dot{\gamma} = 0\,\mathrm{s}^{-1}$

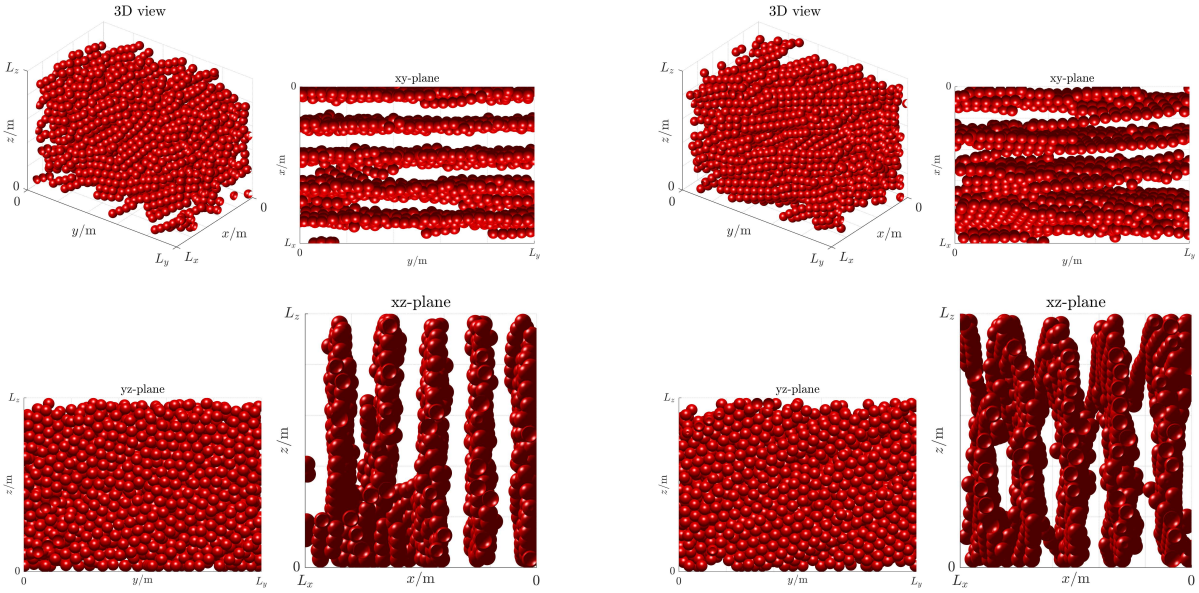
$\dot{\gamma} = 500\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.24: Steady state structures for the single core Fortran models where $\dot{\gamma} = 500\,\mathrm{s}^{-1}$

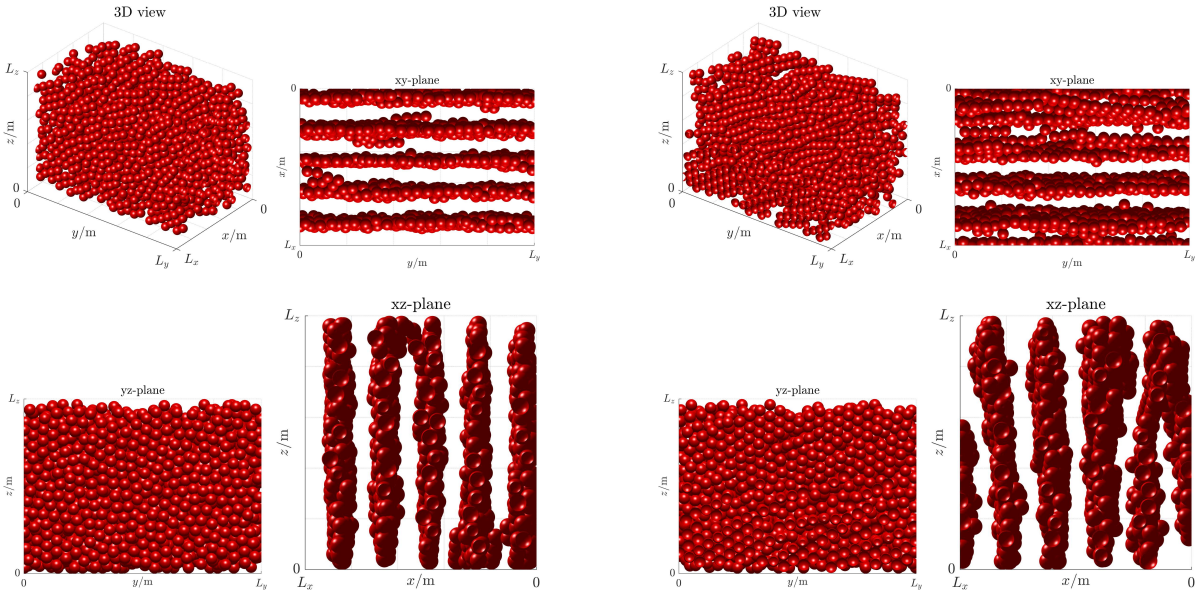
$\dot{\gamma} = 1000\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.25: Steady state structures for the single core Fortran models where $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$

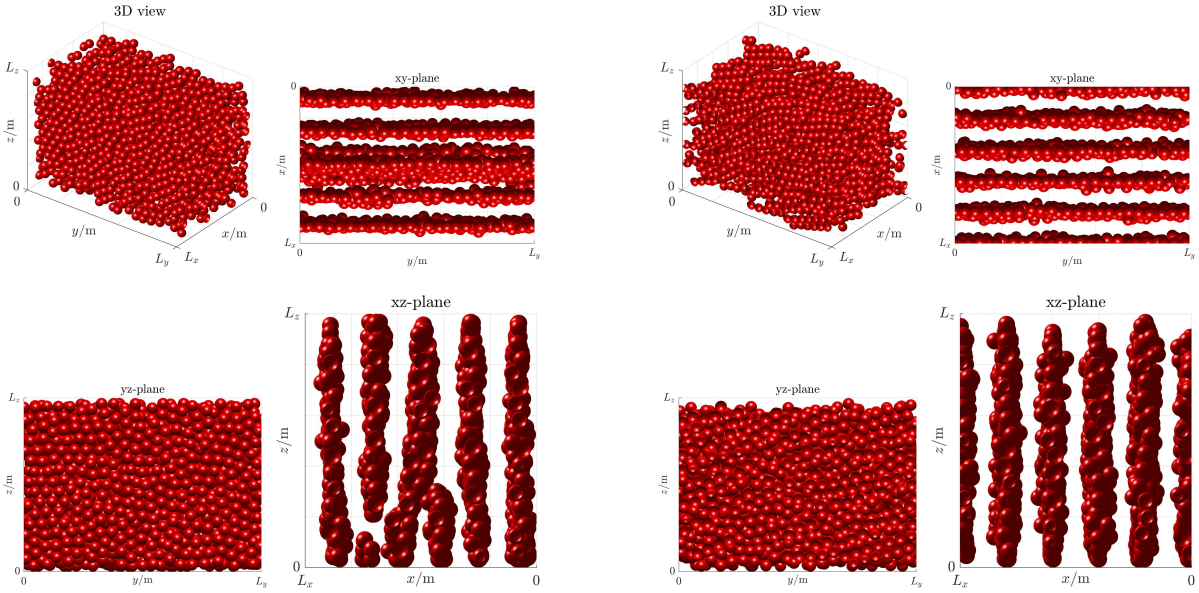
$\dot{\gamma} = 2500\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.26: Steady state structures for the single core Fortran models where $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$

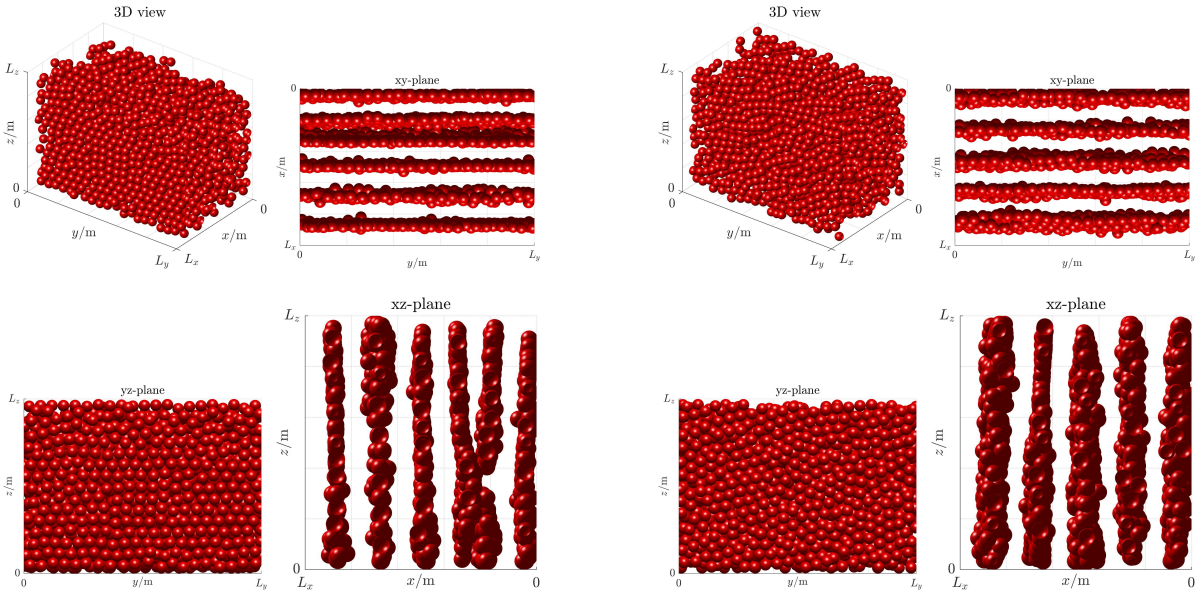
$\dot{\gamma} = 5000\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.27: Steady state structures for the single core Fortran models where $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$

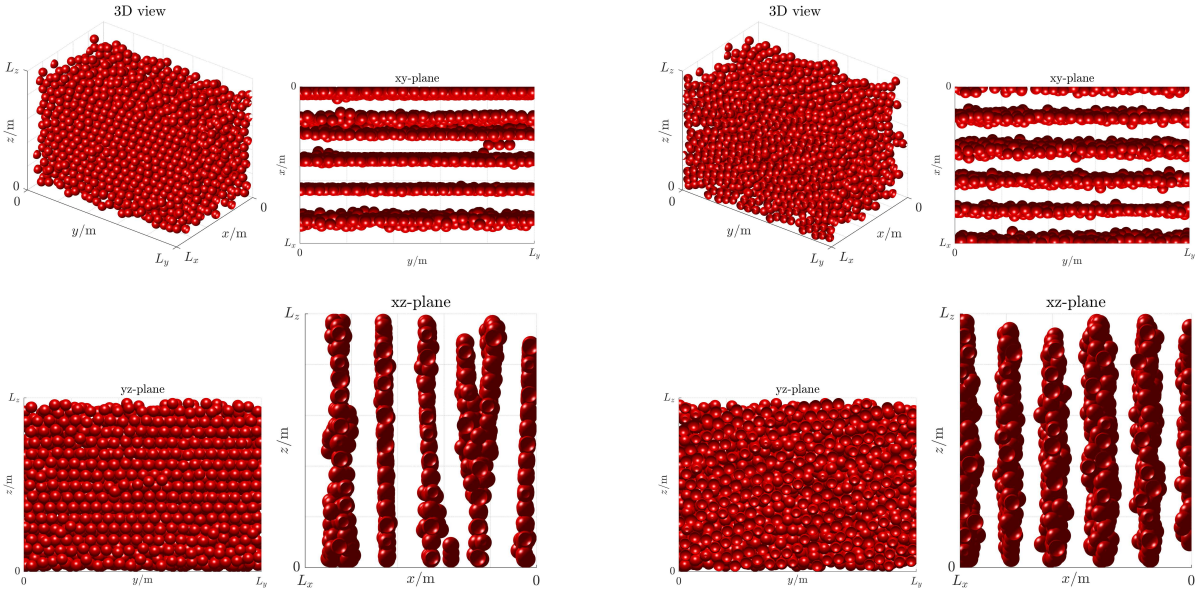
$\dot{\gamma} = 7500\,\mathrm{s}^{-1}$



(a) Fixed dipole method **(b)** Mutual dipole method

Figure E.28: Steady state structures for the single core Fortran models where $\dot{\gamma} = 7500\,\mathrm{s}^{-1}$

$\dot{\gamma} = 10\,000\,\mathrm{s}^{-1}$



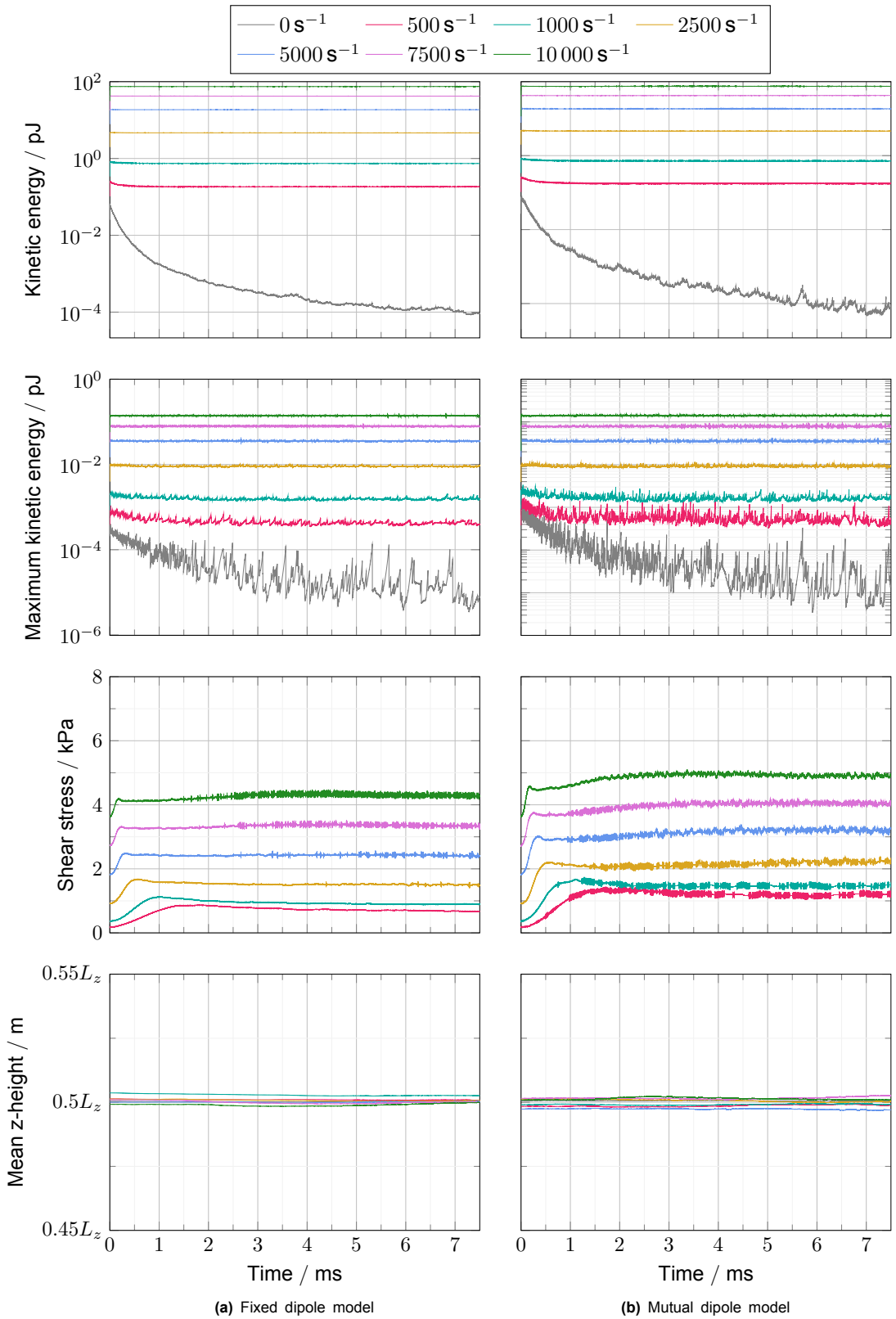
(a) Fixed dipole method

(b) Mutual dipole method

Figure E.29: Steady state structures for the single core Fortran models where $\dot{\gamma} = 10\,000\,\mathrm{s}^{-1}$

E.3.2. Dynamic evolution

Uniform magnetic field

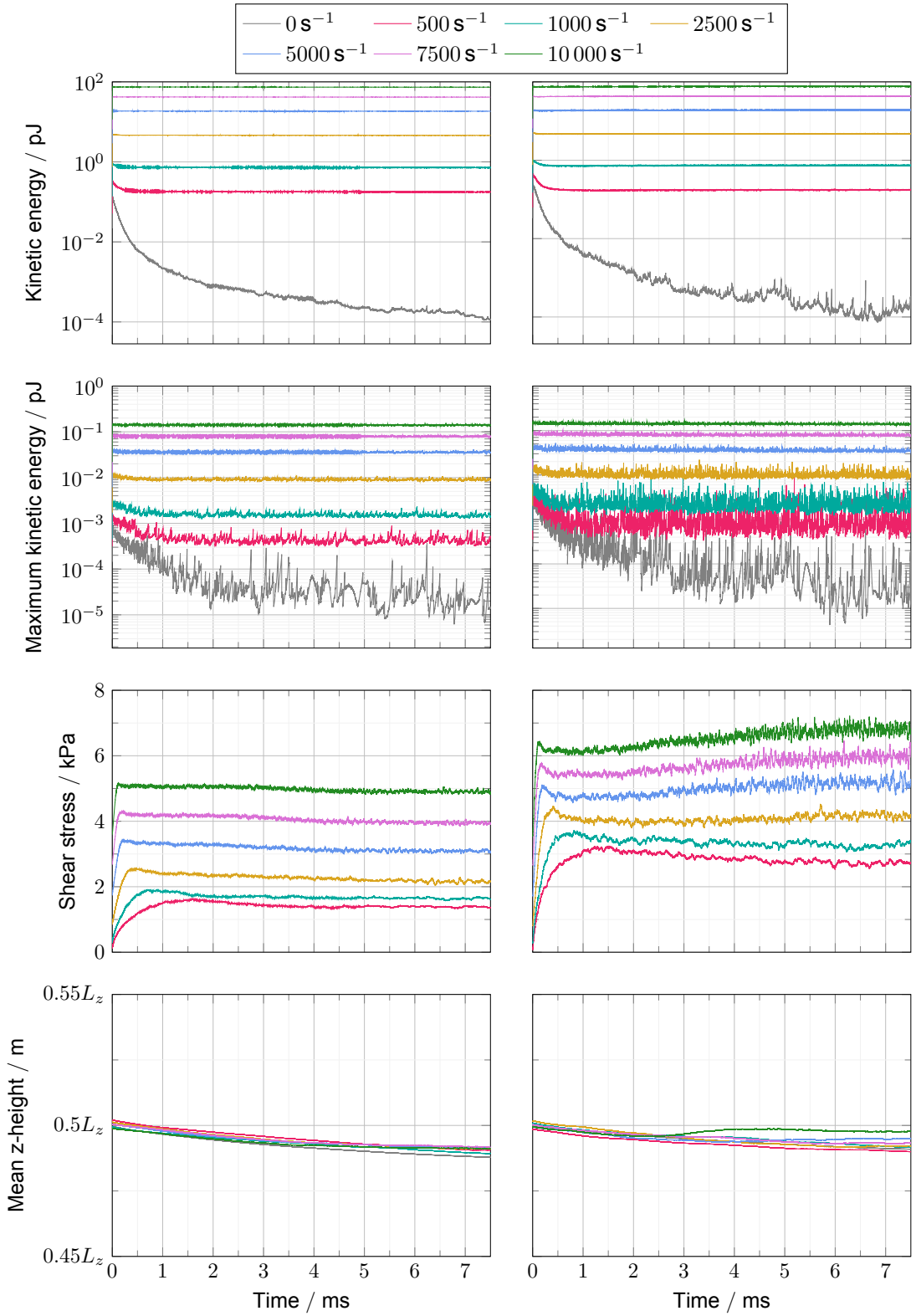


(a) Fixed dipole model

(b) Mutual dipole model

Figure E.30: Dynamic evolution in a uniform magnetic field for the single core Fortran models

Non-uniform: Front of the bar magnet

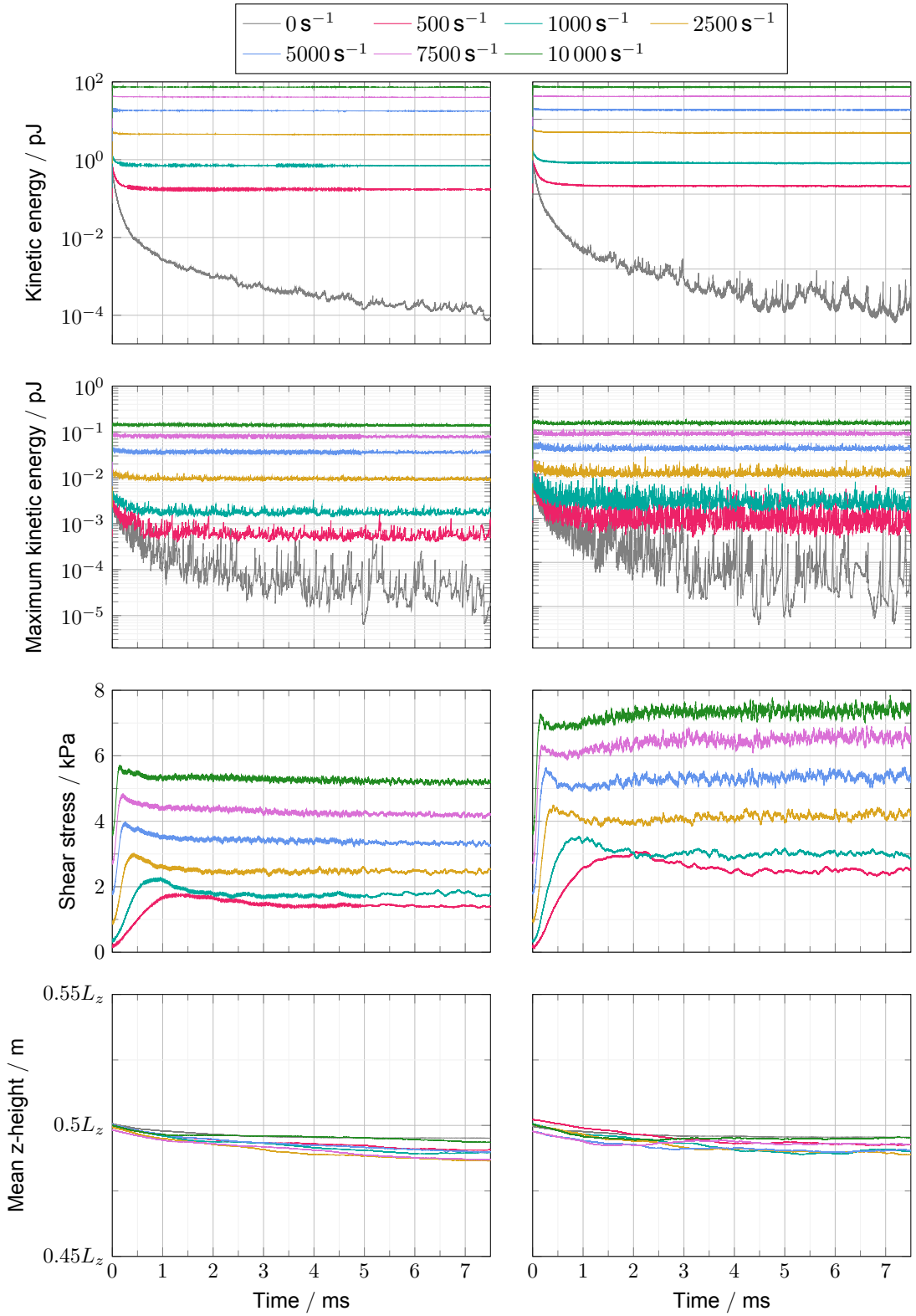


(a) Fixed dipole model

(b) Mutual dipole model

Figure E.31: Dynamic evolution at the front of the bar magnet for the single core Fortran models

Non-uniform: Center of the bar magnet

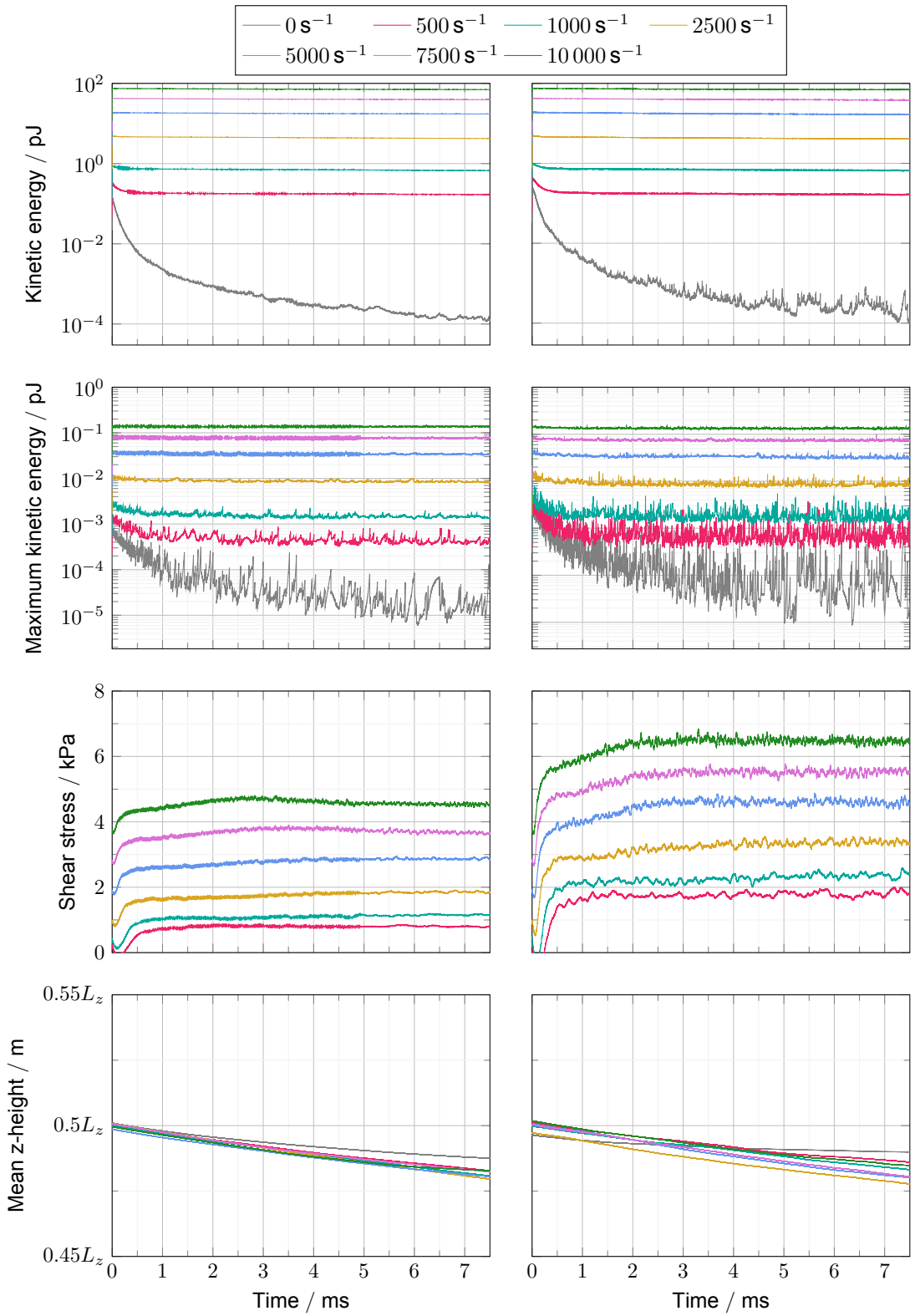


(a) Fixed dipole model

(b) Mutual dipole model

Figure E.32: Dynamic evolution at the center of the bar magnet for the single core Fortran models

Non-uniform: Back of the bar magnet

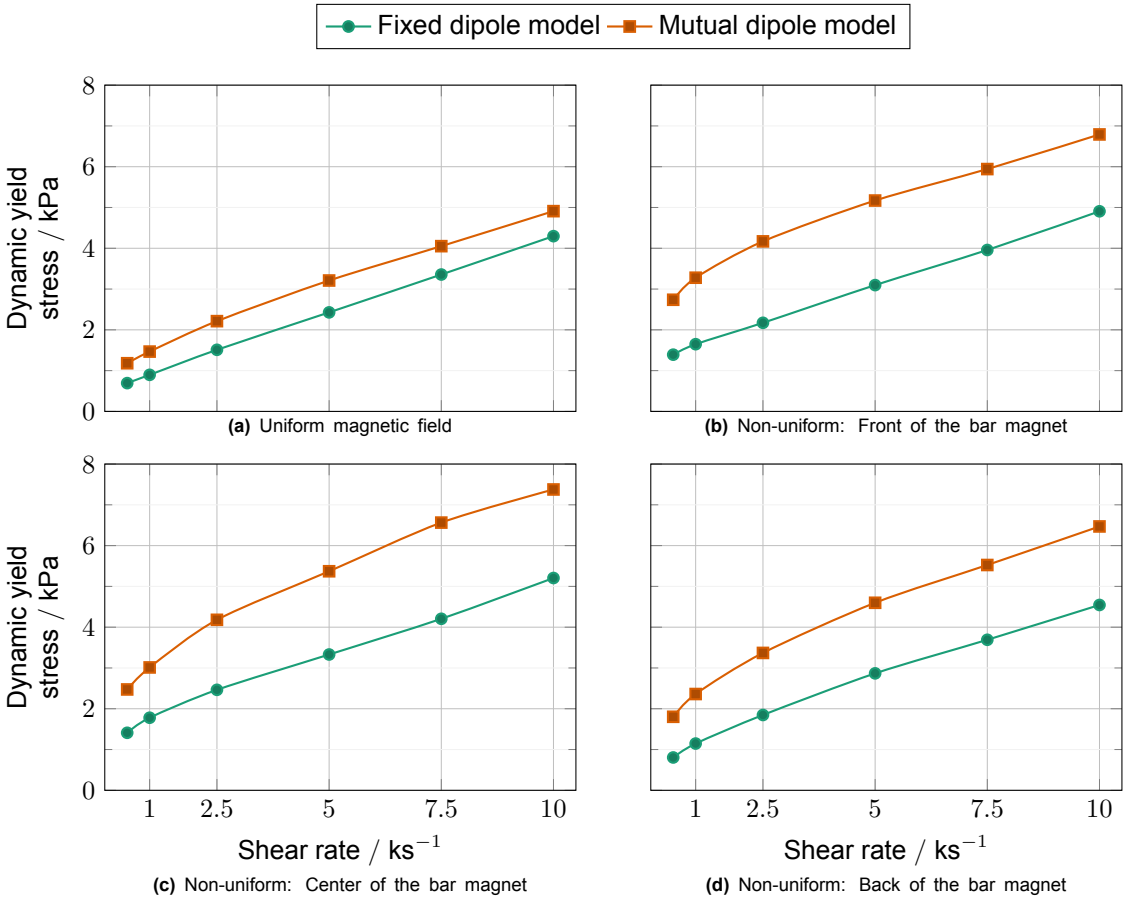


(a) Fixed dipole model

(b) Mutual dipole model

Figure E.33: Dynamic evolution at the back of the bar magnet for the single core Fortran models

E.3.3. Rheological model



(a) Uniform magnetic field

(b) Non-uniform: Front of the bar magnet

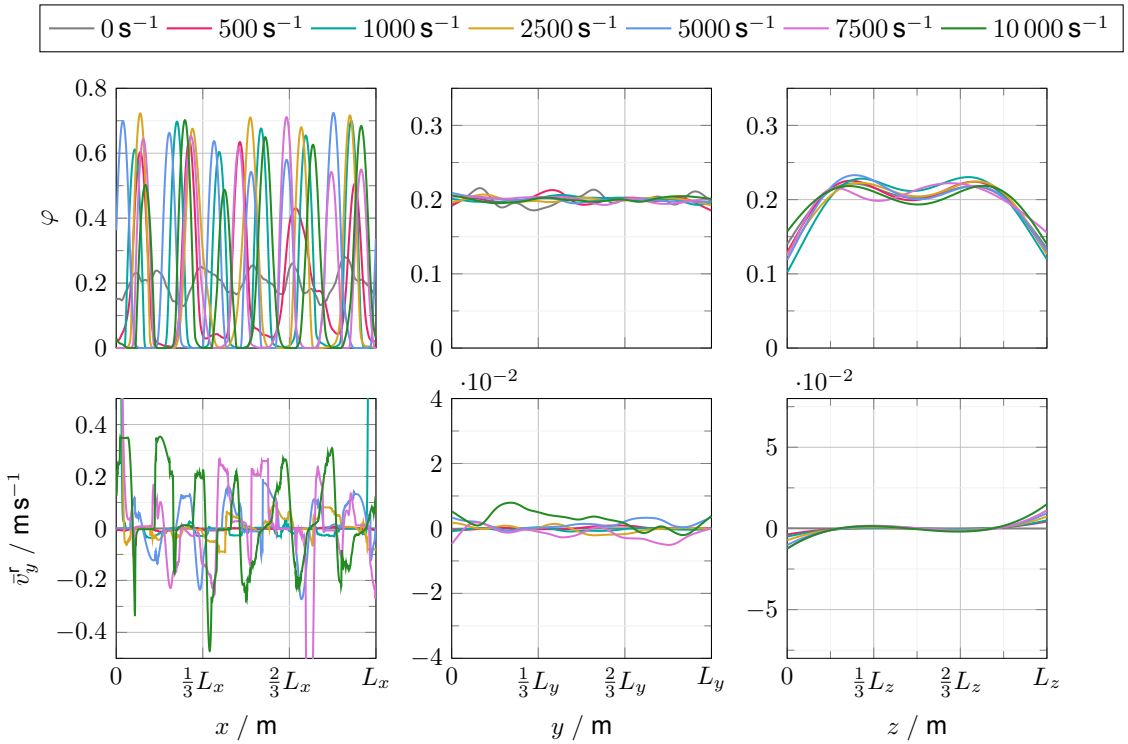
(c) Non-uniform: Center of the bar magnet

(d) Non-uniform: Back of the bar magnet

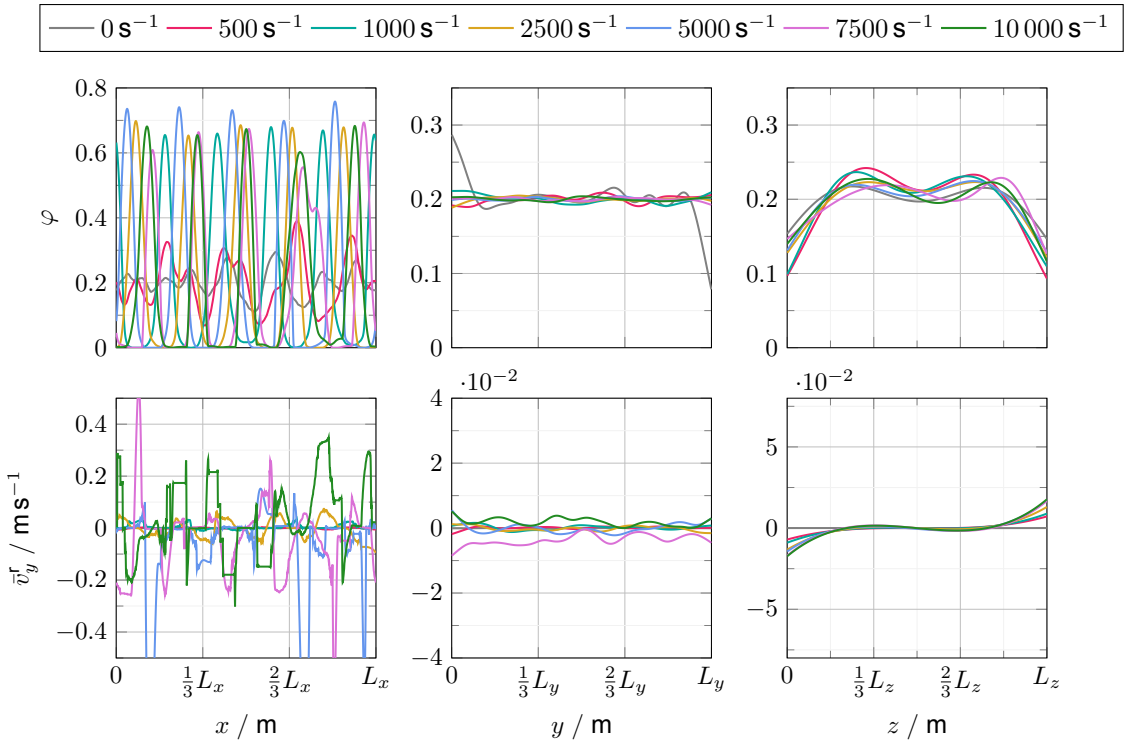
Figure E.34: Rheological models

E.3.4. Slab volume fractions and velocities

Uniform magnetic field



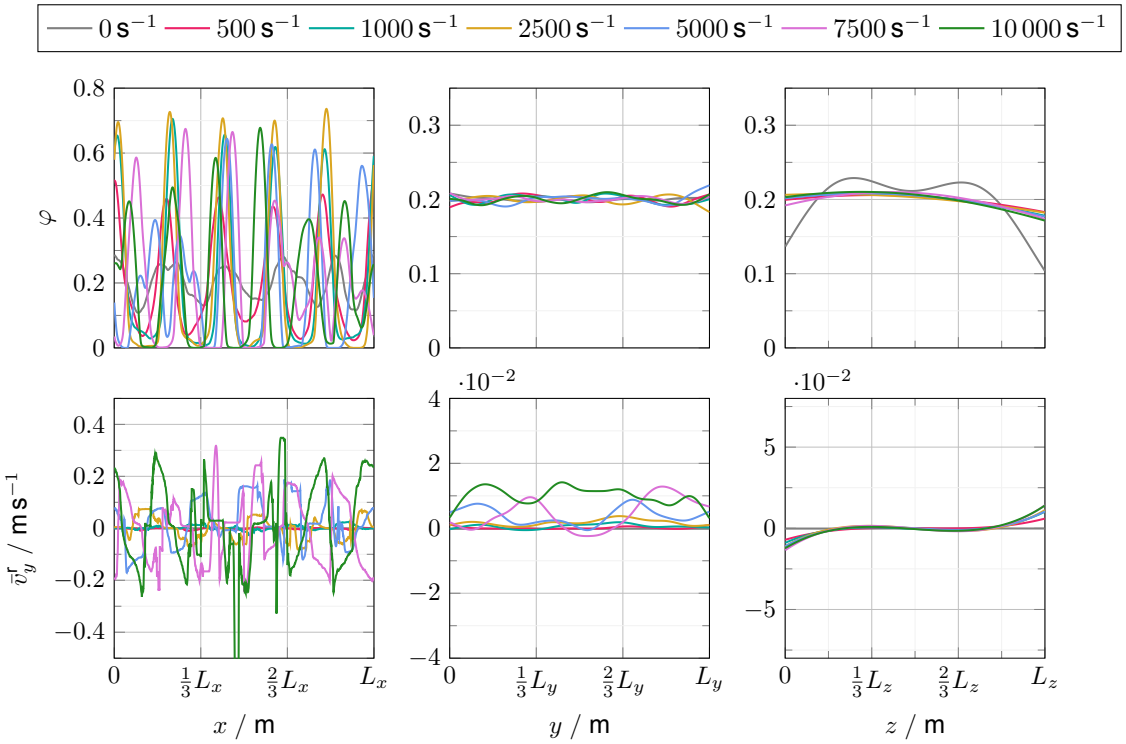
(a) Fixed dipole method



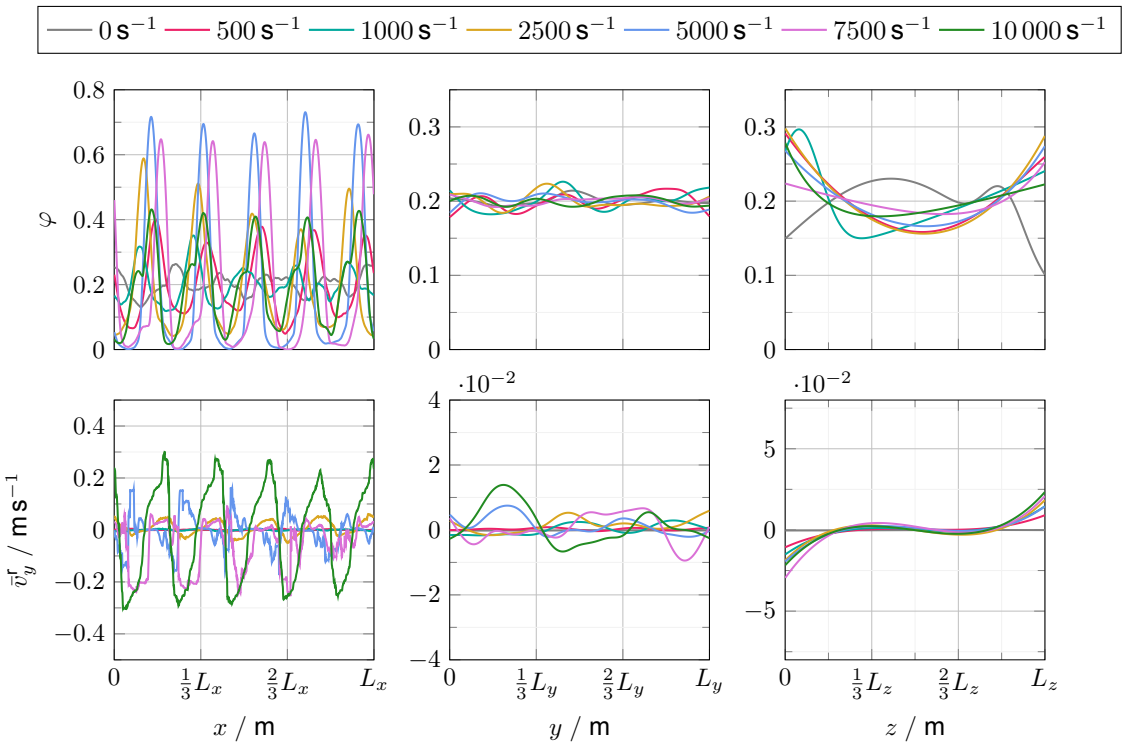
(b) Mutual dipole method

Figure E.35: Slab profiles for the single core Fortran models

Non-uniform: Front of the bar magnet



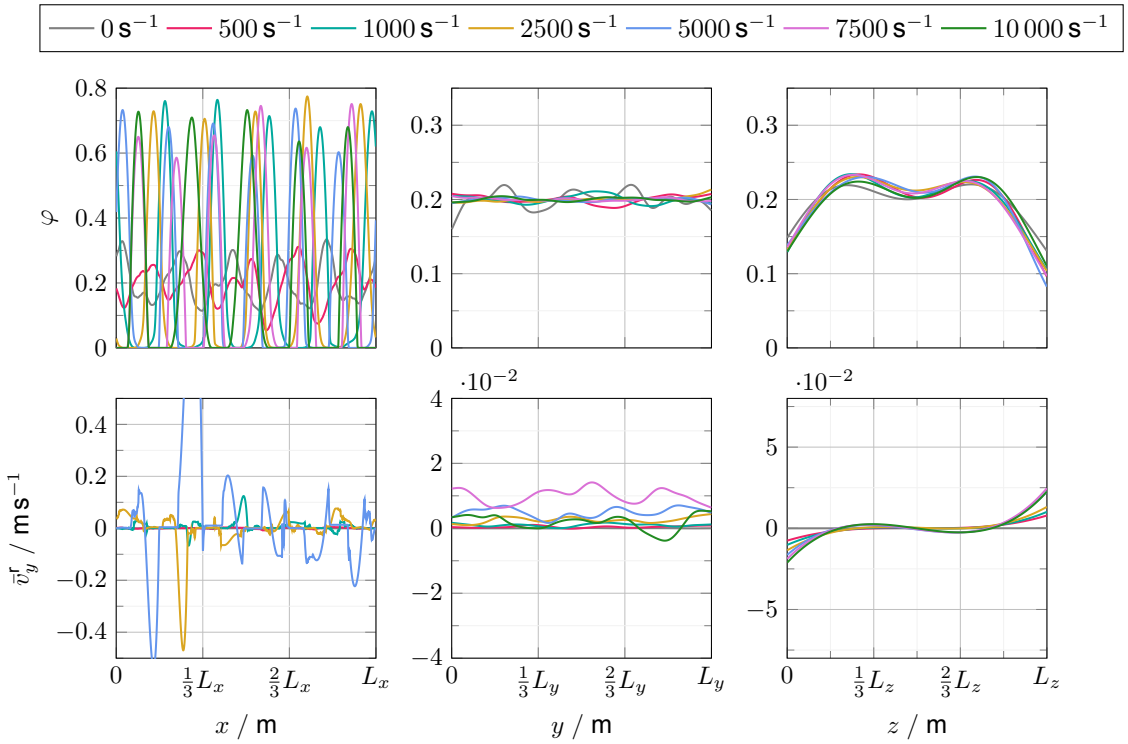
(a) Fixed dipole method



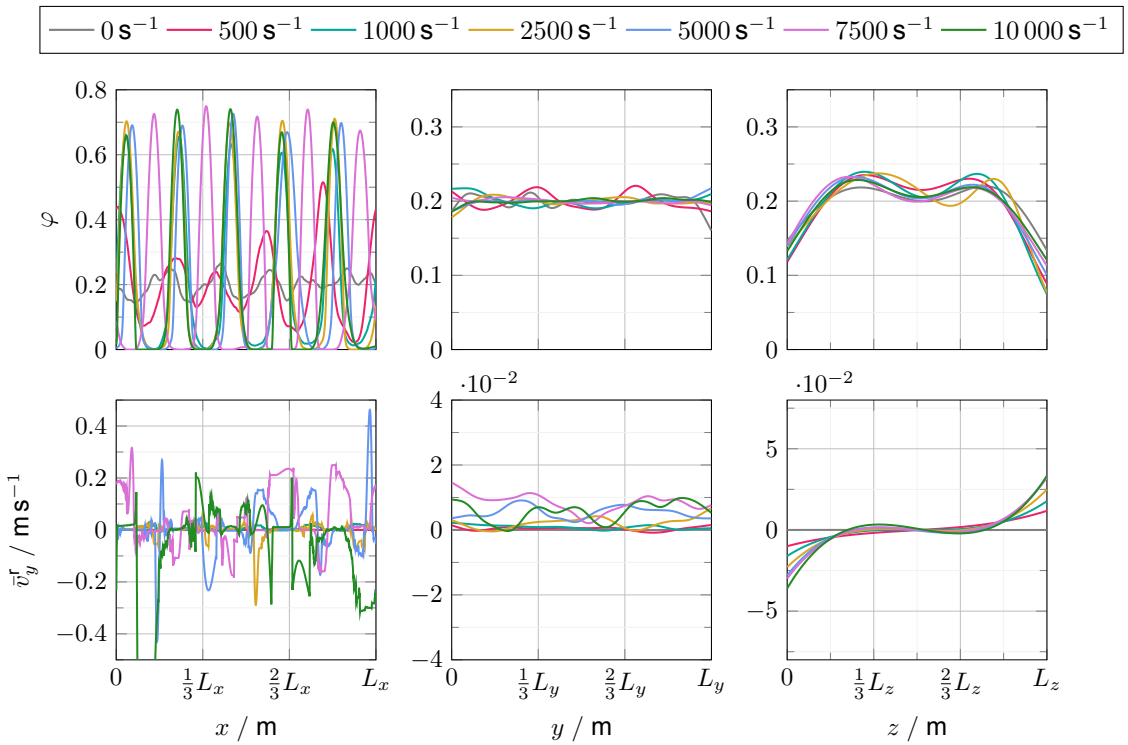
(b) Mutual dipole method

Figure E.36: Slab profiles for the single core Fortran models

Non-uniform: Center of the bar magnet



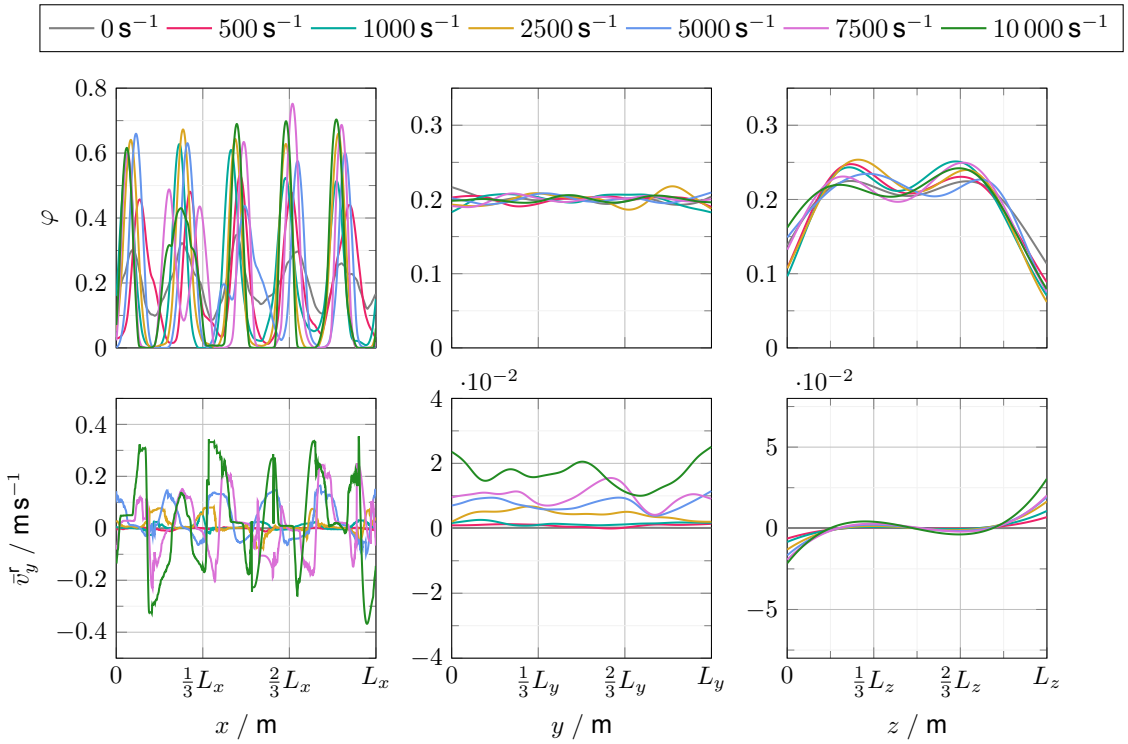
(a) Fixed dipole method



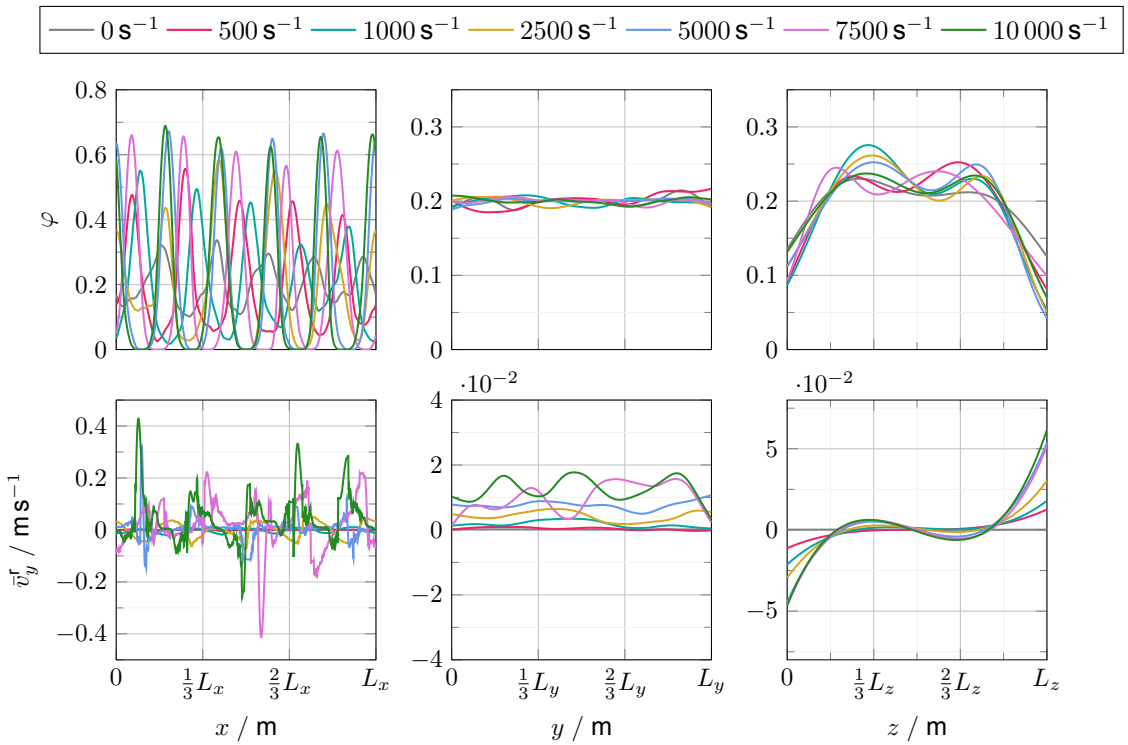
(b) Mutual dipole method

Figure E.37: Slab profiles for the single core Fortran models

Non-uniform: Back of the bar magnet



(a) Fixed dipole method



(b) Mutual dipole method

Figure E.38: Slab profiles for the single core Fortran models

F

Parallel Fortran model

F.1. Source code

F.1.1. Uniform magnetic field

param.f90

```
module mod_param
    !This file is used to load all the parameters

    !Define pi=3.14....
    real(8),parameter :: pi = acos(-1.)

    !MPI parameters
    integer, parameter :: ndims = 2 !2D parallelization (Columns in z-direction)
    integer, dimension(ndims), parameter :: dims = (/3,160/) !Parallelization of 3x160 processes
    integer,parameter :: nprocs = dims(1)*dims(2) !Number of processes

    !General parameters
    real(8),parameter :: mu0 = pi*4.D-07 !Permeabillity of vaccuum
    real(8),parameter :: vfrac = 0.2 !Volume fraction

    !Fluid parameters
    real(8),parameter :: mu_f = mu0, & !Permeabillity
                         rho_f = 1000., & !Density
                         nu_f = 0.2, & !Viscosity
                         sr = 0. !The applied shear rate

    !Particle parameters
    real(8),parameter :: mu_p = 2000.*mu0, & !Permeabillity
                         a = 2.5D-06, & !Radius
                         rho_p = 7.8D3, & !Density
                         E = 500.D6, & !Youngs modulus
                         nu = 0.3, & !Poison ratio
                         Vp = 4./3.*pi*a**3, & !Volume
                         m_p = Vp*rho_p, & !Mass
                         beta = (mu_p/mu0 - mu_f/mu0) / (mu_p/mu0+2.*mu_f/mu0) !Contrast factor

    !Simulation domain
    real(8),parameter :: Srx = dims(1)*2.*3.*a, &   !x dimension of the simulation domain
                         Sry = 8000.*a, &           !y dimension of the simulation domain
                         Srz = 75.D-06              !z dimension of the simulation domain
    real(8),parameter,dimension(3) :: Sredge = (/Srx, Sry, Srz/) !Edges vector of the simulation domain

    !Number of particles
    integer,parameter :: n = nint((Srx*Sry*Srz*vfrac)/Vp), &                  !Number of particles
                         nmax = max( min ( nint(4.*n/(dims(1)*dims(2))) , n),1) !Maximum number of
                                                                              !particles per process

    !Magnetic field parameters
    real(8),parameter :: H0xU = 0, &    !Magnetic field strength x-direction
                         H0yU = 0, &    !Magnetic field strength y-direction
                         H0zU = 0.1/mu0 !Magnetic field strength z-direction

    !Integration parameters
    real(8),parameter :: tstart = 0., & !Start time of the simulation
                         tfinal = 7.5D-3, & !End time of the simulation
                         dt = 5.D-8, & !Time step
                         lambda = 0.65 !Emperical parameter fo rthe modified velocity-Verlet algorithm
    integer,parameter :: steps = nint((tfinal-tstart)/dt),  & !Number of steps
```

```
                        N_frames = 60, & !Number of frames (positions + velocities) to save
                        intsave=nint(real(steps)/real(N_frames)) !Interval for which to save the frames

    !Wall parameters
    real(8),parameter,dimension(2) :: Pw = (/0.D0, Srz/), & !Positions of the walls
                                      Nw = (/1.,-1./) !Normal vector from the wall into the fluid

end module
```

common.f90

```
module mod_common
    !This file is used to define common variables for all the processes and routines

    use mod_param !Load the parameters

    !Make a struct called ap
    type particle
        real(8) ::  x,y,z, &            !Positions
                    vx,vy,vz,&          !Veloccities
                    vpx,vpy,vpz, &      !Prediction velocities
                    ax,ay,az,&          !Accelerations
                    magmx,magmy,magmz, & !Magnetic moments
                    magfx,magfy,magfz, & !Magnetic interaction forces
                    colfx,colfy,colfz, & !Collision forces
                    hydfx,hydfy,hydfz    !Hydrodynamic forces
        integer :: mslv                  !Particle number
        integer, dimension(0:8) :: nb    !Used for communicating with the neighbors
    end type particle
    type(particle),dimension(nmax) :: ap !Set the dimension for this struct

    !Extra common variables
    integer :: pmax, &      !Used to determine the number of particles in a process
               frame = 0, & !Used to determine which frame needs to be saved
               step  = 0    !Used for the step of the simulation
    real(8) :: t = tstart, &    !Time variable, starts at 0
               Ftauext,  &      !External shear force
               Ftauint,  &      !Internal shear force
               Frtauint, &      !Internal shear force times distance
               Frtauext         !External shear force times z-location

end module mod_common

module mod_common_mpi
    !This file is used to define common variables for all the processes and routines

    use mpi !Load the mpi module

    use mod_param !Load the parameters
    implicit none

    !MPI communication parameters
    integer :: COMM_CART               !Used for communication of the processes in the cartesion grid
    logical :: periods(ndims),  &      !Used for the periodic boundary conditions
               reorder                 !Used to order the processes
    integer :: status(MPI_STATUS_SIZE) !Used to communicate the status of the mpi
    integer :: ierr,     &             !Used to return the error message
               my_rank,  &             !The rank of the current process
               numprocs                !Number of processes

    !These variables are used to determine the neighbors of each process
    integer :: right, &
               rightfront, &
               front, &
               leftfront, &
               left, &
               leftback, &
               back, &
               rightback

    integer, dimension(0:8) :: neighbor !Neighbor vector, to store the neighbors of each process
    integer, dimension(1:2) :: coords   !Used to determine the (x,y) coordinates of a process
    integer, dimension(0:8,1:2) :: coordsnb !Used to store the (x,y) coordinates of neighboring processes

    !Variables for the boundaries of each process
    real(8) :: boundleftmyid, &
               boundrightmyid, &
               boundfrontmyid, &
               boundbackmyid

end module mod_common_mpi
```

initmpi.f90

```
module mod_initmpi
    !This file is used to initialize the MPI module, divide the domain and assign processes

    use mpi             !Load the mpi module
    use mod_param       !Load the parameters
    use mod_common      !Load the common variables
    use mod_common_mpi  !Load the common mpi variables

    implicit none

    public initmpi !Make sure the subroutine can be found

contains
    subroutine initmpi !subroutine to initialize the mpi
        implicit none

        real(8)    :: x,y,z,dx,dy,dz !used to determine the location of a process
        real(8)    :: x_corner,y_corner,z_corner !Used to determine the corner location
                                                 !of a process
        integer :: nx,ny,nz !Used for the size of the domain
        !These variables are used for the location of each process
        integer :: coordsleft(1:ndims) , &
                   coordsright(1:ndims), &
                   coordsfront(1:ndims), &
                   coordsback(1:ndims)
        !These are used for the coordinates of the neighboring processes
        integer :: coordsneighbor(1:ndims)

    !--Who am I? --- get the rank of this process and store it in my_rank
    call MPI_COMM_RANK( MPI_COMM_WORLD, my_rank, ierr )

    !Determine the number of processes numprocs
    call MPI_COMM_SIZE( MPI_COMM_WORLD, numprocs, ierr )

    !The domain decomposition
    nx=dims(1) !Division in x direction
    ny=dims(2) !Division in y direction
    nz=1       !Division in z direction

    !Set the boundary conditions
    periods = .true.

    !Create the cartesian domain decomposition
    call MPI_CART_CREATE(MPI_COMM_WORLD,ndims,dims,periods,reorder,COMM_CART,ierr)

    !Determine the processes again
    call MPI_COMM_SIZE(COMM_CART,numprocs,ierr)

    !Determine the rank of each process again
    call MPI_COMM_RANK(COMM_CART,my_rank,ierr)

    !Determine the coordinates of each process
    call MPI_CART_COORDS(COMM_CART,my_rank,2,coords,ierr)

    !Determine the boundaries of each process
    boundleftmyid  = (coords(1)  )*Srx/(1.*dims(1)) !left boundary of subdomain
    boundrightmyid = (coords(1)+1)*Srx/(1.*dims(1)) !right boundary of subdomain
    boundfrontmyid = (coords(2)  )*Sry/(1.*dims(2)) !front boundary of subdomain
    boundbackmyid  = (coords(2)+1)*Sry/(1.*dims(2)) !back boundary of subdomain


    !The coordinates of my left-bottom corner in the total domain:
    dx=Srx/dims(1)
    dy=Sry/dims(2)
    dz=Srz
    x_corner= float( nx/dims(1) * coords(1)  ) * dx
    y_corner= float( ny/dims(2) * coords(2)  ) * dy
    z_corner= 0.

    !The centroid of the process in the total domain
    x=x_corner+(float(1)-0.5)*dx
    y=y_corner+(float(1)-0.5)*dy
    z=z_corner+(float(1)-0.5)*dz

    !Determine the ranks horizontal and vertical neighbors
    call MPI_CART_SHIFT(comm_cart,0,1,left,right,ierr) !Determine the ranks of the left and right neighbor
    call MPI_CART_SHIFT(comm_cart,1,1,front,back,ierr) !Determine the ranks of the front and back neighbor

    !Determine the rank of the rightfront neighbor
    call MPI_CART_COORDS(comm_cart,right,ndims,coordsright,ierr) !Coordinates right neighbor
    call MPI_CART_COORDS(comm_cart,front,ndims,coordsfront,ierr) !Coordinates front neighbor
    coordsneighbor(1) = coordsright(1) !Store the right coordinates
    coordsneighbor(2) = coordsfront(2) !Store the front coordinates
    call MPI_CART_RANK(comm_cart,coordsneighbor,rightfront,ierr) !Determine the rank of the rightfront
```

```
                                                        !neighbor

    !Determine the rank of the rightback neighbor
    call MPI_CART_COORDS(comm_cart,back,ndims,coordsback ,ierr)  !Coordinates back neighbor
    coordsneighbor(1) = coordsright(1) !Store the right coordinates
    coordsneighbor(2) = coordsback(2)  !Store the back coordinates
    call MPI_CART_RANK(comm_cart,coordsneighbor,rightback,ierr)  !Determine the rank of the rightback
                                                                 !neighbor

    !Determine the rank of the leftfront neighbor
    call MPI_CART_COORDS(comm_cart,left ,ndims,coordsleft ,ierr)   !Coordinates left neighbor
    call MPI_CART_COORDS(comm_cart,front ,ndims,coordsfront ,ierr) !Coordinates front neighbor
    coordsneighbor(1) = coordsleft(1)  !Store the left coordinates
    coordsneighbor(2) = coordsfront(2) !Store the front coordinates
    call MPI_CART_RANK(comm_cart,coordsneighbor,leftfront ,ierr) !Determine the rank of the leftfront
                                                                 !neighbor

    !Determine the rank of the leftback neighbor
    call MPI_CART_COORDS(comm_cart,back,ndims,coordsback ,ierr) !Coordinates back neighbor
    coordsneighbor(1) = coordsleft(1) !Store the left coordinates
    coordsneighbor(2) = coordsback(2) !Store the back coordinates
    call MPI_CART_RANK(comm_cart,coordsneighbor,leftback ,ierr) !Determine the rank of the leftfront
                                                                !neighbor

    !Store the ranks of the neighbors
    neighbor(0) = my_rank
    neighbor(1) = right
    neighbor(2) = rightfront
    neighbor(3) = front
    neighbor(4) = leftfront
    neighbor(5) = left
    neighbor(6) = leftback
    neighbor(7) = back
    neighbor(8) = rightback

    !neighbor(6)/leftback        neighbor(7)/back     neighbor(8)/rightback

    !neighbor(5)/left            neighbor(0)/myid     neighbor(1)/right

    !neighbor(4)/leftfront       neighbor(3)/front    neighbor(2)/rightfront

    !x
    !^
    !|
    !|
    !--> y

    !Store the coordinates of the neighbors
    coordsnb(0,1) = coords(1)
    coordsnb(0,2) = coords(2)
    coordsnb(1,1) = coordsright(1)
    coordsnb(1,2) = coordsright(2)
    coordsnb(2,1) = coordsright(1)
    coordsnb(2,2) = coordsfront(2)
    coordsnb(3,1) = coordsfront(1)
    coordsnb(3,2) = coordsfront(2)
    coordsnb(4,1) = coordsleft(1)
    coordsnb(4,2) = coordsfront(2)
    coordsnb(5,1) = coordsleft(1)
    coordsnb(5,2) = coordsleft(2)
    coordsnb(6,1) = coordsleft(1)
    coordsnb(6,2) = coordsback(2)
    coordsnb(7,1) = coordsback(1)
    coordsnb(7,2) = coordsback(2)
    coordsnb(8,1) = coordsright(1)
    coordsnb(8,2) = coordsback(2)

    end subroutine initmpi
end module
```

initpart.f90

```
module mod_initpart
    !This file is used to generate and initialize the particles

    use mod_param        !Load the parameters
    use mod_common       !Load the common variables
    use mod_common_mpi   !Load the common mpi variables

    implicit none

    public initpart !Make sure the routine can be found

contains
```

```
13      subroutine initpart !Subroutine that generates and initializes particles
14          integer :: values(1:8),nseed            !Random number generation variables
15          integer :: p,pp,i,j,rk,nb,nbsend,nbrec  !Integers for loops
16          integer :: pmax_all                     !Used to check if all particles are generated
17          integer :: proccoords(1:ndims), & !Used to determine the process the particle is in
18                     procrank             !Rank of this process
19          real(8)    :: leftbound,rightbound,frontbound,backbound !Boundaries of the process
20          integer, dimension(:), allocatable :: seed !The seed
21          real(8)   ,dimension(n) :: xglob,yglob,zglob, & !The global position of the particle
22                                     magmxglob, magmyglob, magmzglob !The magnetic moment of the particle
23          real(8)   :: rn, & !Normal distance between particles
24                       rx, & !x-distance between particles
25                       ry, & !y-distance between particles
26                       rz  !z-distance between particles
27          real(8) :: ax,ay !Used to determine the master process of a particle
28
29          !Particles are only generated by one process (0)
30          if (my_rank .eq. 0) then
31
32              !Generate random seed
33              call date_and_time(VALUES = values)
34              call random_seed(size=nseed)
35              allocate(seed(1:nseed))
36              seed(:) = values(8)
37              call random_seed(put=seed)
38
39              !Generation of particles
40              do p = 1,n !Loop over the particles
41                  !Generate a random x,y and z position
42  111             call random_number(rx)
43                  call random_number(ry)
44                  call random_number(rz)
45                  xglob(p) = Srx*rx
46                  yglob(p) = Sry*ry
47                  zglob(p) = Srz/2. + 0.99*(2.*(rz-0.5))*((Srz/2.) - a)
48
49                  !Determine the magnetization at this location
50                  magmxglob(p) = 4.*pi*a**3*beta*H0xU
51                  magmyglob(p) = 4.*pi*a**3*beta*H0yU
52                  magmzglob(p) = 4.*pi*a**3*beta*H0zU
53
54                  !Check wheter the particle is in bounds
55                  if ((xglob(p) < 0.) .or. (xglob(p) > Srx  ) .or. &
56                          (yglob(p) < 0.) .or. (yglob(p) > Sry  ) .or. &
57                          (zglob(p) < a ) .or. (zglob(p) > Srz-a)) then
58                      goto 111
59                  end if
60
61                  !Check if the particle overlaps with other particles,
62                  !otherwise the particle is regenerated
63                  do pp=1,p-1
64                      do i=-1,1
65                          do j=-1,1
66                              rn = sqrt((xglob(p)-xglob(pp)+i*Srx)**2 + &
67                                  (yglob(p)-yglob(pp)+j*Sry)**2 + &
68                                  (zglob(p)-zglob(pp))**2)
69                              if (rn< 1.025*2.*a) then
70                                  goto 111
71                              endif
72                          enddo
73                      enddo
74                  enddo
75              enddo
76
77              !Send the particles position and magnetic moment to each process
78              do rk=1,numprocs-1
79                  call MPI_SSEND(xglob(1:n),n,MPI_REAL8,rk,rk+0*(numprocs-1),COMM_CART,ierr)
80                  call MPI_SSEND(yglob(1:n),n,MPI_REAL8,rk,rk+1*(numprocs-1),COMM_CART,ierr)
81                  call MPI_SSEND(zglob(1:n),n,MPI_REAL8,rk,rk+2*(numprocs-1),COMM_CART,ierr)
82                  call MPI_SSEND(magmxglob(1:n),n,MPI_REAL8,rk,rk+3*(numprocs-1),COMM_CART,ierr)
83                  call MPI_SSEND(magmyglob(1:n),n,MPI_REAL8,rk,rk+4*(numprocs-1),COMM_CART,ierr)
84                  call MPI_SSEND(magmzglob(1:n),n,MPI_REAL8,rk,rk+5*(numprocs-1),COMM_CART,ierr)
85              enddo
86          else
87              !Recieve the data on each process except 0
88              call MPI_RECV(xglob(1:n),n,MPI_REAL8,0,my_rank+0*(numprocs-1),COMM_CART,status,ierr)
89              call MPI_RECV(yglob(1:n),n,MPI_REAL8,0,my_rank+1*(numprocs-1),COMM_CART,status,ierr)
90              call MPI_RECV(zglob(1:n),n,MPI_REAL8,0,my_rank+2*(numprocs-1),COMM_CART,status,ierr)
91              call MPI_RECV(magmxglob(1:n),n,MPI_REAL8,0,my_rank+3*(numprocs-1),COMM_CART,status,ierr)
92              call MPI_RECV(magmyglob(1:n),n,MPI_REAL8,0,my_rank+4*(numprocs-1),COMM_CART,status,ierr)
93              call MPI_RECV(magmzglob(1:n),n,MPI_REAL8,0,my_rank+5*(numprocs-1),COMM_CART,status,ierr)
94          endif
95
96          !====== Determine master process for each particle ========
```

```
97
98          !Set the initial vectors for the position to 0
99          ap(1:nmax)%x = 0.
100         ap(1:nmax)%y = 0.
101         ap(1:nmax)%z = 0.
102         ap(1:nmax)%magmx = 0.
103         ap(1:nmax)%magmy = 0.
104         ap(1:nmax)%magmz = 0.
105         ap(1:nmax)%mslv  = 0
106
107         ! Determine master process for each particle
108         pmax = 0 !Reset the number of particles in the process
109         i = 0 !Reset the number of particles already assigned
110
111         do p=1,n !Loop over the particles
112
113             !Determine the master process of this particle
114             ax = 0.5
115             ay = 0.5
116             if (xglob(p) .eq. 0.) ax = 0.49 !proccoords(1) = 0        (otherwise -1     )
117             if (xglob(p) .eq. Srx) ax = 0.51 !proccoords(1) = dims(1)-1 (otherwise dims(1))
118             if (yglob(p) .eq. 0.) ay = 0.49 !proccoords(2) = 0        (otherwise -1     )
119             if (yglob(p) .eq. Sry) ay = 0.51 !proccoords(2) = dims(2)-1 (otherwise dims(2))
120             proccoords(1) = nint(xglob(p)*dims(1)/Srx - ax)   !always between 0 and dims(1)-1
121             proccoords(2) = nint(yglob(p)*dims(2)/Sry - ay)   !always between 0 and dims(2)-1
122             leftbound     = (proccoords(1)  )*Srx/(1.*dims(1)) !left  boundary of particle's master
123             rightbound    = (proccoords(1)+1)*Srx/(1.*dims(1)) !right boundary of particle's master
124             frontbound    = (proccoords(2)  )*Sry/(1.*dims(2)) !front boundary of particle's master
125             backbound     = (proccoords(2)+1)*Sry/(1.*dims(2)) !back  boundary of particle's master
126             call MPI_CART_RANK(comm_cart,proccoords,procrank,ierr) !procrank = rank of process
127                                                                    !where particle's centroid is located
128
129             if (my_rank .eq. procrank) then !If the master is found, save the data on this process
130                 i = i + 1 !Update the number of particles found
131                 pmax = pmax + 1 !Update the number of particles in this process
132
133                 !Save the position of this particle
134                 ap(i)%x = xglob(p)
135                 ap(i)%y = yglob(p)
136                 ap(i)%z = zglob(p)
137
138                 !Save the magnetization of this particle
139                 ap(i)%magmx = magmxglob(p)
140                 ap(i)%magmy = magmyglob(p)
141                 ap(i)%magmz = magmzglob(p)
142
143                 !Save the particle number
144                 ap(i)%mslv = p
145             endif
146         enddo
147
148         call MPI_ALLREDUCE(pmax,pmax_all,1,MPI_INTEGER,MPI_SUM,COMM_CART,ierr) !Summation of the number of
149                                                                                !particles in each process
150
151         !This summation should be the same as the total number of particles, otherwise something went wrong
152         if (my_rank .eq. 0) then
153             if (pmax_all .ne. n) then
154                 write(n+3,*) 'Particles missing'
155                 stop
156             endif
157         endif
158
159         !Set the initial conditions for the particles
160         ap(1:nmax)%vx = 0.
161         ap(1:nmax)%vy = 0.
162         ap(1:nmax)%vz = 0.
163         ap(1:nmax)%vpx = 0.
164         ap(1:nmax)%vpy = 0.
165         ap(1:nmax)%vpz = 0.
166         ap(1:nmax)%ax = 0.
167         ap(1:nmax)%ay = 0.
168         ap(1:nmax)%az = 0.
169         ap(1:nmax)%magfx = 0.
170         ap(1:nmax)%magfy = 0.
171         ap(1:nmax)%magfz = 0.
172         ap(1:nmax)%colfx = 0.
173         ap(1:nmax)%colfy = 0.
174         ap(1:nmax)%colfz = 0.
175         ap(1:nmax)%hydfx = 0.
176         ap(1:nmax)%hydfy = 0.
177         ap(1:nmax)%hydfz = 0.
178     end subroutine
179 end module
```

forceint.f90

```
module mod_forceint
    !This file is used to determine the initial acceleration of the particles

    use mod_param       !Load the parameter
    use mod_common      !Load the common variables
    use mod_common_mpi  !Load the common mpi variables

    implicit none

    public forceint !Make sure the subroutine can be found

contains
    subroutine forceint !subroutine to determine the initial accelerations
        integer :: i,p,q,nb,nbrec,nbsend !Integers for loops
        integer, dimension(0:8)   ::        pmax_nb !Number of particles in neighboring and current process
        integer, dimension(1:nmax,0:8) ::   mslv_nb !Particle number of the particles in neighboring
                                                    !and current process

        !Struct for the neighboring process data
        type neighbourproc
            real(8) :: x,y,z, &          !positions of the particles in neighbors
                       magmx,magmy,magmz !Magnetic moment of the particles in neighbors
        end type neighbourproc
        type(neighbourproc), dimension(1:nmax,0:8) :: anb !Call struct snb

        real(8) :: dist, &      !Distance between particles
                   deltax, &    !x distance
                   deltay, &    !y distance
                   deltaz, &    !z distance
                   nx, &        !Unit vector x component
                   ny, &        !Unit vector y component
                   nz           !Unit vector z component
        real(8),dimension(3) :: magfd !dot products used in the magnetic interaction force
        real(8) :: colcoeff, &  !Coefficient in the collision force
                   eps          !Overlap between particles

        ! Master informs neighbors about the particles of which it is master and
        ! total number of particles it is holding.
        pmax_nb(:)        = 0
        pmax_nb(0)        = pmax
        mslv_nb(:,:)      = 0
        mslv_nb(1:pmax,0) = ap(1:pmax)%mslv
        do nb=1,8
          nbsend = nb + 4
          if (nbsend .gt. 8) nbsend = nbsend - 8
          nbrec  = nb
          call MPI_SENDRECV(pmax_nb(0),1,MPI_INTEGER,neighbor(nbsend),nb, &
                            pmax_nb(nbrec),1,MPI_INTEGER,neighbor(nbrec),nb, &
                            comm_cart,status,ierr)
          call MPI_SENDRECV(mslv_nb(1:pmax_nb(0),0),pmax_nb(0),MPI_INTEGER,neighbor(nbsend),8+nb, &
                            mslv_nb(1:pmax_nb(nbrec),nbrec),pmax_nb(nbrec),MPI_INTEGER,neighbor(nbrec),8+nb, &
                            comm_cart,status,ierr)
        enddo

        ! Initialisation for sending data to neighbors.
        anb(:,1:8)%x        = 0.
        anb(:,1:8)%y        = 0.
        anb(:,1:8)%z        = 0.
        anb(:,1:8)%magmx    = 0.
        anb(:,1:8)%magmy    = 0.
        anb(:,1:8)%magmz    = 0.
        anb(1:pmax,0)%x     = ap(1:pmax)%x
        anb(1:pmax,0)%y     = ap(1:pmax)%y
        anb(1:pmax,0)%z     = ap(1:pmax)%z
        anb(1:pmax,0)%magmx     = ap(1:pmax)%magmx
        anb(1:pmax,0)%magmy     = ap(1:pmax)%magmy
        anb(1:pmax,0)%magmz     = ap(1:pmax)%magmz

        ! Exchange data with neighbors
        do nb=1,8
            nbsend = nb + 4
            if (nbsend .gt. 8) nbsend = nbsend - 8
            nbrec  = nb
            call MPI_SENDRECV(anb(1:pmax_nb(0),0)%x,pmax_nb(0),MPI_REAL8,neighbor(nbsend),nb, &
                              anb(1:pmax_nb(nbrec),nbrec)%x,pmax_nb(nbrec),MPI_REAL8,neighbor(nbrec),nb, &
                              comm_cart,status,ierr)
            call MPI_SENDRECV(anb(1:pmax_nb(0),0)%y,pmax_nb(0),MPI_REAL8,neighbor(nbsend),nb, &
                              anb(1:pmax_nb(nbrec),nbrec)%y,pmax_nb(nbrec),MPI_REAL8,neighbor(nbrec),nb, &
                              comm_cart,status,ierr)
            call MPI_SENDRECV(anb(1:pmax_nb(0),0)%z,pmax_nb(0),MPI_REAL8,neighbor(nbsend),nb, &
                              anb(1:pmax_nb(nbrec),nbrec)%z,pmax_nb(nbrec),MPI_REAL8,neighbor(nbrec),nb, &
                              comm_cart,status,ierr)
```

```
            call MPI_SENDRECV(anb(1:pmax_nb(0),0)%magmx,pmax_nb(0),MPI_REAL8,neighbor(nbsend),nb, &
                              anb(1:pmax_nb(nbrec),nbrec)%magmx,pmax_nb(nbrec),MPI_REAL8,neighbor(nbrec),nb, &
                              comm_cart,status,ierr)
            call MPI_SENDRECV(anb(1:pmax_nb(0),0)%magmy,pmax_nb(0),MPI_REAL8,neighbor(nbsend),nb, &
                              anb(1:pmax_nb(nbrec),nbrec)%magmy,pmax_nb(nbrec),MPI_REAL8,neighbor(nbrec),nb, &
                              comm_cart,status,ierr)
            call MPI_SENDRECV(anb(1:pmax_nb(0),0)%magmz,pmax_nb(0),MPI_REAL8,neighbor(nbsend),nb, &
                              anb(1:pmax_nb(nbrec),nbrec)%magmz,pmax_nb(nbrec),MPI_REAL8,neighbor(nbrec),nb, &
                              comm_cart,status,ierr)
        enddo

        !Reset the shear stress
        Frtauint = 0.
        Frtauext = 0.

        !Particle interaction forces
        do p=1,pmax !Loop over the particles in this process

            !reset forces
            ap(p)%magfx = 0.
            ap(p)%magfy = 0.
            ap(p)%magfz = 0.
            ap(p)%colfx = 0.
            ap(p)%colfy = 0.
            ap(p)%colfz = 0.
            ap(p)%hydfx = 0.
            ap(p)%hydfy = 0.
            ap(p)%hydfz = 0.

            !Reset the shear force
            Ftauext = 0.

            !Loop over the particles in this process and it's neighbors
            do nb=0,8
                do q=1,pmax_nb(nb)

                    Ftauint = 0. !Reset the internal shear force

                    if (mslv_nb(q,nb) .eq. ap(p)%mslv) then !Same particle
                        dist =0.
                    else
                        !distance between the particles
                        deltax =  ap(p)%x - anb(q,nb)%x
                        deltay =  ap(p)%y - anb(q,nb)%y
                        deltaz =  ap(p)%z - anb(q,nb)%z

                        !Correct the distance according to boundary conditions
                        if (deltax > Srx/2.) then
                            deltax = deltax - Srx
                        elseif (deltax < -Srx/2.) then
                            deltax = deltax + Srx
                        endif

                        if (deltay > Sry/2.) then
                            deltay = deltay - Sry
                        elseif (deltay < -Sry/2.) then
                            deltay = deltay + Sry
                        endif

                        !Distance between particles
                        dist = sqrt(deltax**2 + deltay**2 + deltaz**2)

                        !Unit vector
                        nx = deltax/dist
                        ny = deltay/dist
                        nz = deltaz/dist

                        !Magnetic force (With cut-off radius)
                        if (dist .lt. 6.*a) then
                            !dot products used in the magnetic interaction force
                            magfd(1) = (ap(p)%magmx*anb(q,nb)%magmx + &
                                        ap(p)%magmy*anb(q,nb)%magmy + &
                                        ap(p)%magmz*anb(q,nb)%magmz)
                            magfd(2) = (anb(q,nb)%magmx*nx + &
                                        anb(q,nb)%magmy*ny + &
                                        anb(q,nb)%magmz*nz)
                            magfd(3) = (ap(p)%magmx*nx + &
                                        ap(p)%magmy*ny + &
                                        ap(p)%magmz*nz)

                            !Magnetic interaction force
                            ap(p)%magfx = ap(p)%magfx + 3.*mu0/(4.*pi*dist**4) * &
                                        (magfd(1)*             nx + &
                                         magfd(2)*     ap(p)%magmx + &
```

```
                                    magfd(3)*anb(q,nb)%magmx - &
                                    5.*magfd(2)*magfd(3)* nx)

                        ap(p)%magfy = ap(p)%magfy + 3.*mu0/(4.*pi*dist**4) * &
                                    (magfd(1)*              ny + &
                                     magfd(2)*    ap(p)%magmy + &
                                     magfd(3)*anb(q,nb)%magmy - &
                                     5.*magfd(2)*magfd(3)* ny)

                        ap(p)%magfz = ap(p)%magfz + 3.*mu0/(4.*pi*dist**4) * &
                                    (magfd(1)*              nz + &
                                     magfd(2)*    ap(p)%magmz + &
                                     magfd(3)*anb(q,nb)%magmz - &
                                     5.*magfd(2)*magfd(3)* nz)

                        !Addition to the internal shear stress force (y-direction)
                        Ftauint =   3.*mu0/(4.*pi*dist**4) * &
                                    (magfd(1)*              ny + &
                                     magfd(2)*    ap(p)%magmy + &
                                     magfd(3)*anb(q,nb)%magmy - &
                                     5.*magfd(2)*magfd(3)* ny)
                    endif

                    !Collision force
                    if (dist .lt. 2.*a) then
                        eps = 2.*a-dist !Overlap
                        colcoeff = (2./3.) * E/(1.-nu**2) * sqrt(a/2.) * eps**(3./2.) !Collision coefficient
                        !Collision force
                        ap(p)%colfx = ap(p)%colfx + colcoeff*nx
                        ap(p)%colfy = ap(p)%colfy + colcoeff*ny
                        ap(p)%colfz = ap(p)%colfz + colcoeff*nz

                        !Addition to the internal shear stress force (y-direction)
                        Ftauint = Ftauint + colcoeff*ny
                    endif

                    !Determine the internal shear stress force time the z-distance
                    Frtauint = Frtauint + deltaz*Ftauint
                endif
            enddo !q
        enddo !nb

        !Wall force
        do i=1,2 !Loop over the walls

            !Distance between particle and wall
            deltaz = abs(ap(p)%z - Pw(i))

            if (deltaz .lt. a) then !If contact then

                eps = a-deltaz !Overlap
                colcoeff = (2./3.) * E/(1.-nu**2) * sqrt(a) * eps**(3./2.) !Collision coefficient

                !Collision force (only in z-direction, no slip)
                ap(p)%colfz = ap(p)%colfz + colcoeff*Nw(i)

                !The wall doesn't contribute to the shear stress in y direction (Slip is not used)
                Ftauext = 0.
            endif
        enddo

        !Fluid stokes drag force
        ap(p)%hydfx = -6.*pi*a*nu_f*            ap(p)%vx
        ap(p)%hydfy =  6.*pi*a*nu_f*(sr*ap(p)%z-ap(p)%vy)
        ap(p)%hydfz = -6.*pi*a*nu_f*            ap(p)%vz

        !Addition to the external shear stress force (y-direction)
        Ftauext = Ftauext + 6.*pi*a*nu_f*(sr*ap(p)%z-ap(p)%vpy)

        !External shear stress force times particle z-location
        Frtauext = Frtauext + ap(p)%z*Ftauext

        !Acceleration
        ap(p)%ax = (ap(p)%magfx + ap(p)%colfx + ap(p)%hydfx)/m_p
        ap(p)%ay = (ap(p)%magfy + ap(p)%colfy + ap(p)%hydfy)/m_p
        ap(p)%az = (ap(p)%magfz + ap(p)%colfz + ap(p)%hydfz)/m_p
    enddo
  end subroutine
end module
```

kinetic.f90

```
module mod_kinetic
  !This file is used to determine the dynamic evolution
```

```
3
4       use mod_param        !Load the parameters
5       use mod_common       !Load the common variables
6       use mod_common_mpi   !Load the common mpi variables
7
8       implicit none
9
10      public kinetic !Make sure the routine can be found
11  contains
12      subroutine kinetic(t) !Subroutine that saves the dynamic evolution and
13                            !the steady state slab velocities + particle densities
14          implicit none
15          real(8),intent(in) :: t !The current time (input)
16          real(8) :: Ek,Ekall, &          !Kinetic energy per process and total kinetic energy
17                     Ekmax, Ekmaxall, &   !Maximum kinetic energy per process and maximum kinetic energy
18                     ztot,ztotall         !Mean z-height per process and total mean z-height
19          real(8) :: Frtauintall, &       !Total internal shear stress force times distance
20                     Frtauextall          !Total external shear stress force times z-location
21          integer :: p,zh !Used for loops
22          integer,parameter :: LVrel = nint(Srz/(2.*a)) !Number of slabs
23          real(8) :: Vrels, Vrelsall, &   !Relative slab velocity per process and total relavtive slab velocity
24                     Vs, Vsall, &         !Slab velocity per process and total slab velocity
25                     phis, phisall        !Particle density in a slab per process and total partilce denisty per slab
26          integer :: nrel, nrelall        !Number of particles in a slab per process and total particles in a slab
27
28          !Initialize the kinetic energy
29          Ek = 0.
30          Ekmax = 0.
31          ztot = 0.
32          !Determine the sum of kinetic energy, the maximum kinetic energy and the total z height per process
33          do p=1,pmax
34              Ek = Ek +   0.5*m_p*(ap(p)%vx**2+ap(p)%vy**2+ap(p)%vz**2)
35              Ekmax = max(0.5*m_p*(ap(p)%vx**2+ap(p)%vy**2+ap(p)%vz**2),Ekmax)
36              ztot = ztot + ap(p)%z
37          enddo
38
39          !Wait on all processes to finish
40          call MPI_BARRIER(comm_cart,ierr)
41
42          !Determine the sum of the kinetic energies
43          call MPI_ALLREDUCE(Ek,Ekall,1,MPI_REAL8,MPI_SUM,COMM_CART,ierr)
44
45          !Determine the maximum kinetic energy of all processes
46          call MPI_ALLREDUCE(Ekmax,Ekmaxall,1,MPI_REAL8,MPI_MAX,COMM_CART,ierr)
47
48          !Determine the total mean z-height
49          call MPI_ALLREDUCE(ztot,ztotall,1,MPI_REAL8,MPI_SUM,COMM_CART,ierr)
50
51          !Determine the total internal shear stress force times distance
52          call MPI_ALLREDUCE(Frtauint,Frtauintall,1,MPI_REAL8,MPI_SUM,COMM_CART,ierr)
53
54          !Determine the total external shear stress force times z-location
55          call MPI_ALLREDUCE(Frtauext,Frtauextall,1,MPI_REAL8,MPI_SUM,COMM_CART,ierr)
56
57          !Wait on all processes to finish
58          call MPI_BARRIER(comm_cart,ierr)
59
60          !Write the dynamic evolution to a file (only one process does this)
61          if (my_rank .eq. 0) then
62              write(n+1,*) t, Ekall, Ekmaxall, (ztotall/n)/Srz
63              write(n+2,*) t, -1./(Srx*Sry*Srz)*Frtauintall/2., 1./(Srx*Sry*Srz)*Frtauextall
64          endif
65
66          !At the last step of the simulation
67          if (step .eq. steps) then
68              do zh = 0,LVrel-1 !Loop over the slabs
69                  Vs = 0.     !Reset the slab veolcity
70                  Vrels = 0.  !Reset the relative slab velocity
71                  nrel = 0    !Reset the number of particles in slab
72
73                  do p =1,pmax !Loop over the particles in this process
74                      !If a particle is in the current slab, then
75                      if ((ap(p)%z .gt. real(zh,8)/real(LVrel,8)*Srz) .and. (ap(p)%z .lt. real(zh+1,8)/real(LVrel,8)*Srz)) ther
76                          !Add the y-velocity to the slab velocity
77                          Vs = Vs + ap(p)%vy
78
79                          !Add the relative y-velocity to the relative slab velocity
80                          Vrels = Vrels + (sr*ap(p)%z-ap(p)%vy)
81
82                          !Add the particle to the particle in the slab
83                          nrel = nrel + 1
84                      endif
85                  enddo
86
```

```
                !Wait on all processes to finish
                call MPI_BARRIER(comm_cart,ierr)

                !Combine the slab velocity of all processes
                call MPI_ALLREDUCE(Vs,Vsall,1,MPI_REAL8,MPI_SUM,COMM_CART,ierr)

                !Combine the relative slab velocity of all processes
                call MPI_ALLREDUCE(Vrels,Vrelsall,1,MPI_REAL8,MPI_SUM,COMM_CART,ierr)

                !Combine the number of particles per slab of all processes
                call MPI_ALLREDUCE(nrel,nrelall,1,MPI_INT,MPI_SUM,COMM_CART,ierr)

                !Wait on all processes to finish
                call MPI_BARRIER(comm_cart,ierr)

                !Write the data to a file (only one process does this)
                if (my_rank .eq. 0) then
                    write(n+4,*) Vsall, Vrelsall, nrelall, Vsall/nrelall, Vrelsall/nrelall,  &
                                 nrelall*Vp/(Srx*Sry*1./real(LVrel)*Srz), &
                                 (zh + (zh+1))/2. * Srz/LVrel , (zh + (zh+1))/2. * Srz/(2.*a)/LVrel
                endif
            enddo
        endif
    end subroutine
end module
```

framewrite.f90

```
module mod_framewrite
    !This file is used to save the positions and velocities of the particles at

    use mod_param        !Load the parameters
    use mod_common       !Load the common variables
    use mod_common_mpi   !Load the common mpi variables

    implicit none

    public framewrite !Make sure the routine can be found

contains
    subroutine framewrite !Subroutine to save a frame
        character (len=255) :: file_frame, &    !File name for the frame for each process
                               file_tframe      !File name for the frome of all processes
        real(8) :: tpost, &                 !The time to be read of the individual frame files
                   xpost, ypost, zpost, &   !The position to be read of the individual frame files
                   vxpost, vypost, vzpost   !The velocity to be read of the individual frame files
        integer :: steppost, &      !The step to be read of the individual frame files
                   ppost, &         !The particle number to be read of the individual frame files
                   my_rankpost,&    !The rank to be read of the individual frame files
                   Reason           !Used to determine the end of a individual frame file
        integer :: p,proc   !Integers for loops

        !Make the dat file name for this process
        write(file_tframe,"('./out/frames/my_rank=',i0,'.dat')") my_rank

        !Open the dat file for this process
        open(my_rank,file=trim(file_tframe),status='replace')

        do p =1,pmax !Loop over the particles
            !Write the data to the individual dat file
            write(my_rank,*) t, step, ap(p)%mslv, my_rank, ap(p)%x, ap(p)%y, ap(p)%z, ap(p)%vx, ap(p)%vy, ap(p)%vz
        enddo

        !Close the dat file
        close(my_rank)

        !Wait for each process to finish
        call MPI_BARRIER(comm_cart,ierr)

        !Process 0 writes all the data to one file (To not exceed diks quota)
        if (my_rank .eq. 0) then
            !Make a dat file name for this frame
            write(file_frame,"('./out/frames/frame',i0,'.dat')") frame

            !Open the dat file for this frame
            open(frame+nprocs,file=trim(file_frame),status='replace')

            do proc=0,nprocs-1 !Loop over the processes
                !Make the dat file name for this process
                write(file_tframe,"('./out/frames/my_rank=',i0,'.dat')") proc

                !Open the dat file for this process
                open(proc,file=trim(file_tframe))

```

```
57
58                do  !Read the dat file data untill the end of the file
59                    read(proc,*,IOSTAT=Reason) tpost, steppost, ppost, my_rankpost, &
60                                xpost, ypost, zpost, &
61                                vxpost, vypost, vzpost
62
63                    if (Reason < 0) then !Exit loop if the end of the file is reached
64                        exit
65                    endif
66
67                    !Write indivual data of each process to the frame dat file
68                    write(frame+nprocs,*) tpost, steppost, ppost, my_rankpost, &
69                                  xpost, ypost, zpost, &
70                                  vxpost, vypost, vzpost
71                enddo
72
73                !Close the dat file for this process and delete the process file
74                close(proc,status='delete')
75            enddo
76
77            !Close the frame dat file
78            close(frame)
79
80            !Update the frame counter
81            frame = frame + 1
82        endif
83    end subroutine
84 end module
```

forcepred.f90

```
1  module mod_forcepred
2      !This file is used to predict the acceleration of the particles
3
4      use mod_param        !Load the parameter
5      use mod_common       !Load the common variables
6      use mod_common_mpi   !Load the common mpi variables
7
8      implicit none
9
10     public forcepred !Make sure the subroutine can be found
11
12 contains
13     subroutine forcepred !subroutine to predict the accelerations
14         integer :: i,p,q,nb,nbrec,nbsend !Integers for loops
15         integer, dimension(0:8)   ::        pmax_nb !Number of particles in neighboring and current process
16         integer, dimension(1:nmax,0:8) ::  mslv_nb !Particle number of the particles in neighboring
17                                                    !and current process
18
19         !Struct for the neighboring process data
20         type neighbourproc
21             real(8) :: x,y,z, &          !positions of the particles in neighbors
22                        magmx,magmy,magmz !Magnetic moment of the particles in neighbors
23         end type neighbourproc
24         type(neighbourproc), dimension(1:nmax,0:8) :: anb !Call struct snb
25
26         real(8) :: dist, &      !Distance between particles
27                    deltax, &    !x distance
28                    deltay, &    !y distance
29                    deltaz, &    !z distance
30                    nx, &        !Unit vector x component
31                    ny, &        !Unit vector y component
32                    nz           !Unit vector z component
33         real(8),dimension(3) :: magfd !dot products used in the magnetic interaction force
34         real(8) :: colcoeff, &  !Coefficient in the collision force
35                    eps          !Overlap between particles
36
37         ! Master informs neighbors about the particles of which it is master and
38         ! total number of particles it is holding.
39         pmax_nb(:)        = 0
40         pmax_nb(0)        = pmax
41         mslv_nb(:,:)      = 0
42         mslv_nb(1:pmax,0) = ap(1:pmax)%mslv
43         do nb=1,8
44           nbsend = nb + 4
45           if (nbsend .gt. 8) nbsend = nbsend - 8
46           nbrec  = nb
47           call MPI_SENDRECV(pmax_nb(0),1,MPI_INTEGER,neighbor(nbsend),nb, &
48                             pmax_nb(nbrec),1,MPI_INTEGER,neighbor(nbrec),nb, &
49                             comm_cart,status,ierr)
50           call MPI_SENDRECV(mslv_nb(1:pmax_nb(0),0),pmax_nb(0),MPI_INTEGER,neighbor(nbsend),8+nb, &
51                             mslv_nb(1:pmax_nb(nbrec),nbrec),pmax_nb(nbrec),MPI_INTEGER,neighbor(nbrec),8+nb, &
52                             comm_cart,status,ierr)
53         enddo
```

```
54
55          ! Initialisation for sending data to neighbors.
56          anb(:,1:8)%x        = 0.
57          anb(:,1:8)%y        = 0.
58          anb(:,1:8)%z        = 0.
59          anb(:,1:8)%magmx    = 0.
60          anb(:,1:8)%magmy    = 0.
61          anb(:,1:8)%magmz    = 0.
62          anb(1:pmax,0)%x     = ap(1:pmax)%x
63          anb(1:pmax,0)%y     = ap(1:pmax)%y
64          anb(1:pmax,0)%z     = ap(1:pmax)%z
65          anb(1:pmax,0)%magmx     = ap(1:pmax)%magmx
66          anb(1:pmax,0)%magmy     = ap(1:pmax)%magmy
67          anb(1:pmax,0)%magmz     = ap(1:pmax)%magmz
68
69          ! Exchange data with neighbors
70          do nb=1,8
71              nbsend = nb + 4
72              if (nbsend .gt. 8) nbsend = nbsend - 8
73              nbrec  = nb
74              call MPI_SENDRECV(anb(1:pmax_nb(0),0)%x,pmax_nb(0),MPI_REAL8,neighbor(nbsend),nb, &
75                                anb(1:pmax_nb(nbrec),nbrec)%x,pmax_nb(nbrec),MPI_REAL8,neighbor(nbrec),nb, &
76                                comm_cart,status,ierr)
77              call MPI_SENDRECV(anb(1:pmax_nb(0),0)%y,pmax_nb(0),MPI_REAL8,neighbor(nbsend),nb, &
78                                anb(1:pmax_nb(nbrec),nbrec)%y,pmax_nb(nbrec),MPI_REAL8,neighbor(nbrec),nb, &
79                                comm_cart,status,ierr)
80              call MPI_SENDRECV(anb(1:pmax_nb(0),0)%z,pmax_nb(0),MPI_REAL8,neighbor(nbsend),nb, &
81                                anb(1:pmax_nb(nbrec),nbrec)%z,pmax_nb(nbrec),MPI_REAL8,neighbor(nbrec),nb, &
82                                comm_cart,status,ierr)
83              call MPI_SENDRECV(anb(1:pmax_nb(0),0)%magmx,pmax_nb(0),MPI_REAL8,neighbor(nbsend),nb, &
84                                anb(1:pmax_nb(nbrec),nbrec)%magmx,pmax_nb(nbrec),MPI_REAL8,neighbor(nbrec),nb, &
85                                comm_cart,status,ierr)
86              call MPI_SENDRECV(anb(1:pmax_nb(0),0)%magmy,pmax_nb(0),MPI_REAL8,neighbor(nbsend),nb, &
87                                anb(1:pmax_nb(nbrec),nbrec)%magmy,pmax_nb(nbrec),MPI_REAL8,neighbor(nbrec),nb, &
88                                comm_cart,status,ierr)
89              call MPI_SENDRECV(anb(1:pmax_nb(0),0)%magmz,pmax_nb(0),MPI_REAL8,neighbor(nbsend),nb, &
90                                anb(1:pmax_nb(nbrec),nbrec)%magmz,pmax_nb(nbrec),MPI_REAL8,neighbor(nbrec),nb, &
91                                comm_cart,status,ierr)
92          enddo
93
94          !Reset the shear stress
95          Frtauint = 0.
96          Frtauext = 0.
97
98          !Particle interaction forces
99          do p=1,pmax !Loop over the particles in this process
100
101             !reset forces
102             ap(p)%magfx = 0.
103             ap(p)%magfy = 0.
104             ap(p)%magfz = 0.
105             ap(p)%colfx = 0.
106             ap(p)%colfy = 0.
107             ap(p)%colfz = 0.
108             ap(p)%hydfx = 0.
109             ap(p)%hydfy = 0.
110             ap(p)%hydfz = 0.
111
112             !Reset the shear force
113             Ftauext = 0.
114
115             !Loop over the particles in this process and it's neighbors
116             do nb=0,8
117                 do q=1,pmax_nb(nb)
118
119                     Ftauint = 0. !Reset the internal shear force
120
121                     if (mslv_nb(q,nb) .eq. ap(p)%mslv) then !Same particle
122                         dist =0.
123                     else
124                         !distance between the particles
125                         deltax =  ap(p)%x - anb(q,nb)%x
126                         deltay =  ap(p)%y - anb(q,nb)%y
127                         deltaz =  ap(p)%z - anb(q,nb)%z
128
129                         !Correct the distance according to boundary conditions
130                         if (deltax > Srx/2.) then
131                             deltax = deltax - Srx
132                         elseif (deltax < -Srx/2.) then
133                             deltax = deltax + Srx
134                         endif
135
136                         if (deltay > Sry/2.) then
137                             deltay = deltay - Sry
```

```
                    elseif (deltay < -Sry/2.) then
                        deltay = deltay + Sry
                    endif

                    !Distance between particles
                    dist = sqrt(deltax**2 + deltay**2 + deltaz**2)

                    !Unit vector
                    nx = deltax/dist
                    ny = deltay/dist
                    nz = deltaz/dist

                    !Magnetic force (With cut-off radius)
                    if (dist .lt. 6.*a) then
                        !dot products used in the magnetic interaction force
                        magfd(1) = (ap(p)%magmx*anb(q,nb)%magmx + &
                                    ap(p)%magmy*anb(q,nb)%magmy + &
                                    ap(p)%magmz*anb(q,nb)%magmz)
                        magfd(2) = (anb(q,nb)%magmx*nx + &
                                    anb(q,nb)%magmy*ny + &
                                    anb(q,nb)%magmz*nz)
                        magfd(3) = (ap(p)%magmx*nx + &
                                    ap(p)%magmy*ny + &
                                    ap(p)%magmz*nz)

                        !Magnetic interaction force
                        ap(p)%magfx = ap(p)%magfx + 3.*mu0/(4.*pi*dist**4) * &
                                    (magfd(1)*              nx + &
                                     magfd(2)*     ap(p)%magmx + &
                                     magfd(3)*anb(q,nb)%magmx - &
                                     5.*magfd(2)*magfd(3)* nx)

                        ap(p)%magfy = ap(p)%magfy + 3.*mu0/(4.*pi*dist**4) * &
                                    (magfd(1)*              ny + &
                                     magfd(2)*     ap(p)%magmy + &
                                     magfd(3)*anb(q,nb)%magmy - &
                                     5.*magfd(2)*magfd(3)* ny)

                        ap(p)%magfz = ap(p)%magfz + 3.*mu0/(4.*pi*dist**4) * &
                                    (magfd(1)*              nz + &
                                     magfd(2)*     ap(p)%magmz + &
                                     magfd(3)*anb(q,nb)%magmz - &
                                     5.*magfd(2)*magfd(3)* nz)

                        !Addition to the internal shear stress force (y-direction)
                        Ftauint =   3.*mu0/(4.*pi*dist**4) * &
                                    (magfd(1)*              ny + &
                                     magfd(2)*     ap(p)%magmy + &
                                     magfd(3)*anb(q,nb)%magmy - &
                                     5.*magfd(2)*magfd(3)* ny)
                    endif

                    !Collision force
                    if (dist .lt. 2.*a) then
                        eps = 2.*a-dist !Overlap
                        colcoeff = (2./3.) * E/(1.-nu**2) * sqrt(a/2.) * eps**(3./2.) !Collision coefficient
                        !Collision force
                        ap(p)%colfx = ap(p)%colfx + colcoeff*nx
                        ap(p)%colfy = ap(p)%colfy + colcoeff*ny
                        ap(p)%colfz = ap(p)%colfz + colcoeff*nz

                        !Addition to the internal shear stress force (y-direction)
                        Ftauint = Ftauint + colcoeff*ny
                    endif

                    !Determine the internal shear stress force time the z-distance
                    Frtauint = Frtauint + deltaz*Ftauint
                endif
            enddo !q
        enddo !nb

        !Wall force
        do i=1,2 !Loop over the walls

            !Distance between particle and wall
            deltaz = abs(ap(p)%z - Pw(i))

            if (deltaz .lt. a) then !If contact then

                eps = a-deltaz !Overlap
                colcoeff = (2./3.) * E/(1.-nu**2) * sqrt(a) * eps**(3./2.) !Collision coefficient

                !Collision force (only in z-direction, no slip)
                ap(p)%colfz = ap(p)%colfz + colcoeff*Nw(i)
```

```
222
223                     !The wall doesn't contribute to the shear stress in y direction (Slip is not used)
224                     Ftauext = 0.
225                 endif
226             enddo
227
228             !Fluid stokes drag force
229             ap(p)%hydfx = -6.*pi*a*nu_f*            ap(p)%vpx
230             ap(p)%hydfy =  6.*pi*a*nu_f*(sr*ap(p)%z-ap(p)%vpy)
231             ap(p)%hydfz = -6.*pi*a*nu_f*            ap(p)%vpz
232
233             !Addition to the external shear stress force (y-direction)
234             Ftauext = Ftauext + 6.*pi*a*nu_f*(sr*ap(p)%z-ap(p)%vpy)
235
236             !External shear stress force times particle z-location
237             Frtauext = Frtauext + ap(p)%z*Ftauext
238
239             !Acceleration
240             ap(p)%ax = (ap(p)%magfx + ap(p)%colfx + ap(p)%hydfx)/m_p
241             ap(p)%ay = (ap(p)%magfy + ap(p)%colfy + ap(p)%hydfy)/m_p
242             ap(p)%az = (ap(p)%magfz + ap(p)%colfz + ap(p)%hydfz)/m_p
243         enddo
244     end subroutine
245 end module
```

verlet.f90

```
1  module mod_verlet
2      !This file is used to integrate the newton's equation
3      !and determine the new master processes after the positions are updated
4
5      use mod_param        !Load the parameters
6      use mod_common       !Load the common variables
7      use mod_common_mpi   !Load the common mpi variables
8      use mod_forcepred    !Load the forcepred subroutine
9
10     implicit none
11
12     public verlet,new_masters !Make sure the subroutines can be found
13
14 contains
15     subroutine verlet !This routine is used to integrate newton's equation
16         integer :: p !Used to loop over the particles
17
18
19         do p=1,pmax !Loop over the partilces in this process
20                 !Update particle position
21                 ap(p)%x = ap(p)%x + dt*ap(p)%vx + 0.5*dt**2*ap(p)%ax
22                 ap(p)%y = ap(p)%y + dt*ap(p)%vy + 0.5*dt**2*ap(p)%ay
23                 ap(p)%z = ap(p)%z + dt*ap(p)%vz + 0.5*dt**2*ap(p)%az
24
25                 !Predict the velocity
26                 ap(p)%vpx = ap(p)%vx + lambda*dt*ap(p)%ax
27                 ap(p)%vpy = ap(p)%vy + lambda*dt*ap(p)%ay
28                 ap(p)%vpz = ap(p)%vz + lambda*dt*ap(p)%az
29
30                 !Update the velocity with the old accelerations
31                 ap(p)%vx = ap(p)%vx + 0.5*dt*ap(p)%ax
32                 ap(p)%vy = ap(p)%vy + 0.5*dt*ap(p)%ay
33                 ap(p)%vz = ap(p)%vz + 0.5*dt*ap(p)%az
34         enddo
35
36         !Wait for all processes to finish
37         call MPI_BARRIER(comm_cart,ierr)
38
39         !Update the time
40         t = t + dt
41
42         !Determine the new master processes of the particles
43         call new_masters
44
45         !Wait for all processes to finish
46         call MPI_BARRIER(comm_cart,ierr)
47
48
49         do p =1,pmax !Loop over the particles in this process
50             !Determine the magnetic moment at the new position
51             ap(p)%magmx = 4.*pi*a**3*beta*H0xU
52             ap(p)%magmy = 4.*pi*a**3*beta*H0yU
53             ap(p)%magmz = 4.*pi*a**3*beta*H0zU
54         enddo
55
56         !Wait for all processes to finish
57         call MPI_BARRIER(comm_cart,ierr)
```

```
58
59          !Predict the acceleration of the particles
60          call forcepred
61
62          !Wait for all processes to finish
63          call MPI_BARRIER(comm_cart,ierr)
64
65
66          do p=1,pmax !Loop over the particles
67              !Update the velocity with the predicted accelerations
68              ap(p)%vx = ap(p)%vx + 0.5*dt*ap(p)%ax
69              ap(p)%vy = ap(p)%vy + 0.5*dt*ap(p)%ay
70              ap(p)%vz = ap(p)%vz + 0.5*dt*ap(p)%az
71          enddo
72
73          !Wait for all processes to finish
74          call MPI_BARRIER(comm_cart,ierr)
75
76      end subroutine verlet
77
78      subroutine new_masters
79          implicit none
80          integer :: p,p2,i,j,nb,nbsend,nbrec,nbrec2,proc !Integers for loops
81          integer :: tag                                  !tag for communication between processes
82          integer :: pmax_all                 !The total number of particles of all processes
83          real(8) :: npx, npy,npz, &          !The recieved new particle position
84                     npvx,npvy,npvz, &        !The recieved new particle velocity
85                     npvpx,npvpy,npvpz, &     !The recieved new particle predicted velocity
86                     npax,npay,npaz           !The recieved new particle acceleration
87          integer :: npmslv                   !The recieved new partilce number
88          integer,dimension(0:8) :: pmax_nb   !Vector to communicate the number of particles
89                                              !in the neighboring processes
90          integer,dimension(1:nmax,0:8) :: newmaster_nb, &    !Matrix that stores the particles that need to change process
91                                                              !and to which process to go
92                                           mslv_nb            !Matrix to communicate which particles are
93                                                              !in the neighboring processes
94          integer :: proccoords(1:ndims),procrank             !The coordinates and ranks of the processes
95          real(8) :: ax, ay                                   !Used to determine the master process of a particle
96
97          ! Correction for periodic b.c. 's.
98          do p=1,pmax
99              if (ap(p)%x .gt. Srx) ap(p)%x = ap(p)%x - Srx
100             if (ap(p)%x .lt. 0. ) ap(p)%x = ap(p)%x + Srx
101             if (ap(p)%y .gt. Sry) ap(p)%y = ap(p)%y - Sry
102             if (ap(p)%y .lt. 0. ) ap(p)%y = ap(p)%y + Sry
103         enddo
104
105
106         ! Master informs neighbors about the total number of particles it is holding and which particles.
107         pmax_nb(:)        = 0
108         pmax_nb(0)        = pmax
109         mslv_nb(:,:)      = 0
110         mslv_nb(1:pmax,0) = ap(1:pmax)%mslv
111         do nb=1,8
112           nbsend = nb + 4
113           if (nbsend .gt. 8) nbsend = nbsend - 8
114           nbrec  = nb
115           call MPI_SENDRECV(pmax_nb(0),1,MPI_INTEGER,neighbor(nbsend),nb, &
116                             pmax_nb(nbrec),1,MPI_INTEGER,neighbor(nbrec),nb, &
117                             comm_cart,status,ierr)
118           call MPI_SENDRECV(mslv_nb(1:pmax_nb(0),0),pmax_nb(0),MPI_INTEGER,neighbor(nbsend),8+nb, &
119                             mslv_nb(1:pmax_nb(nbrec),nbrec),pmax_nb(nbrec),MPI_INTEGER,neighbor(nbrec),8+nb, &
120                             comm_cart,status,ierr)
121         enddo
122
123         !Wait for all processes to finish
124         call MPI_BARRIER(comm_cart,ierr)
125
126         !Determine if a particle has changed process
127         do p = 1,pmax_nb(0) !loop over the particles
128             !Reset the newmaster
129             newmaster_nb(p,0) = 0
130
131             !Determine the process the particle is in
132             ax = 0.5
133             ay = 0.5
134             if (ap(p)%x .eq. 0.)  ax = 0.49
135             if (ap(p)%x .eq. Srx) ax = 0.51
136             if (ap(p)%y .eq. 0.)  ay = 0.49
137             if (ap(p)%y .eq. Sry) ay = 0.51
138             proccoords(1) = nint(ap(p)%x*dims(1)/Srx - ax)
139             proccoords(2) = nint(ap(p)%y*dims(2)/Sry - ay)
140             call MPI_CART_RANK(comm_cart,proccoords,procrank,ierr) !The rank of the process the particle is in
141             if (my_rank .ne. procrank) then !if the particle has changed process
```

```
142                 !Determine to which neighbor the particle traveled
143                 do nb = 1,8
144                     if (procrank .eq. neighbor(nb)) then
145                         newmaster_nb(p,0) = nb !Store to which neighbor the particle has traveled
146                         exit                   !Exit if the neighbor is found
147                     endif
148                 enddo
149             endif
150         enddo
151
152         !Wait for all processes to finish
153         call MPI_BARRIER(comm_cart,ierr)
154
155         !Inform neighbors about changes in masters
156         do nb=1,8
157             nbsend = nb + 4
158             if (nbsend .gt. 8) nbsend = nbsend - 8
159             nbrec  = nb
160             call MPI_SENDRECV(newmaster_nb(1:pmax_nb(0),0)         ,pmax_nb(0)     ,MPI_INTEGER,neighbor(nbsend),nb, &
161                               newmaster_nb(1:pmax_nb(nbrec),nbrec),pmax_nb(nbrec),MPI_INTEGER,neighbor(nbrec) ,nb, &
162                               comm_cart,status,ierr)
163         enddo
164
165         !Wait for all processes to finish
166         call MPI_BARRIER(comm_cart,ierr)
167
168         !Send data from old master to new master
169         do p = 1,pmax_nb(0) !loop over the particle
170             if (newmaster_nb(p,0) .gt. 0) then !If the particle has changed master then
171                 nbsend = newmaster_nb(p,0)     !Neighbor to send the data to
172                 tag = ap(p)%mslv               !tag of the communication
173
174                 !Send data to the other process
175                 call MPI_SEND(ap(p)%mslv  ,1,MPI_INTEGER,neighbor(nbsend),tag      ,comm_cart,ierr)
176                 call MPI_SEND(ap(p)%x     ,1,MPI_REAL8  ,neighbor(nbsend),tag+1*n  ,comm_cart,ierr)
177                 call MPI_SEND(ap(p)%y     ,1,MPI_REAL8  ,neighbor(nbsend),tag+2*n  ,comm_cart,ierr)
178                 call MPI_SEND(ap(p)%z     ,1,MPI_REAL8  ,neighbor(nbsend),tag+3*n  ,comm_cart,ierr)
179                 call MPI_SEND(ap(p)%vx    ,1,MPI_REAL8  ,neighbor(nbsend),tag+4*n  ,comm_cart,ierr)
180                 call MPI_SEND(ap(p)%vy    ,1,MPI_REAL8  ,neighbor(nbsend),tag+5*n  ,comm_cart,ierr)
181                 call MPI_SEND(ap(p)%vz    ,1,MPI_REAL8  ,neighbor(nbsend),tag+6*n  ,comm_cart,ierr)
182                 call MPI_SEND(ap(p)%vpx   ,1,MPI_REAL8  ,neighbor(nbsend),tag+7*n  ,comm_cart,ierr)
183                 call MPI_SEND(ap(p)%vpy   ,1,MPI_REAL8  ,neighbor(nbsend),tag+8*n  ,comm_cart,ierr)
184                 call MPI_SEND(ap(p)%vpz   ,1,MPI_REAL8  ,neighbor(nbsend),tag+9*n  ,comm_cart,ierr)
185                 call MPI_SEND(ap(p)%ax    ,1,MPI_REAL8  ,neighbor(nbsend),tag+10*n,comm_cart,ierr)
186                 call MPI_SEND(ap(p)%ay    ,1,MPI_REAL8  ,neighbor(nbsend),tag+11*n,comm_cart,ierr)
187                 call MPI_SEND(ap(p)%az    ,1,MPI_REAL8  ,neighbor(nbsend),tag+12*n,comm_cart,ierr)
188
189                 !Remove the data from the particle that changed process
190                 ap(p)%mslv = -ap(p)%mslv !The particle's number is changed negative for reording later
191                 ap(p)%x    = 0
192                 ap(p)%y    = 0
193                 ap(p)%z    = 0
194                 ap(p)%vx   = 0
195                 ap(p)%vy   = 0
196                 ap(p)%vz   = 0
197                 ap(p)%vpx  = 0
198                 ap(p)%vpy  = 0
199                 ap(p)%vpz  = 0
200                 ap(p)%ax   = 0
201                 ap(p)%ay   = 0
202                 ap(p)%az   = 0
203
204                 !Change the number of particles in this process by -1
205                 pmax = pmax-1
206             endif
207         enddo
208
209         !Wait for all processes to finish
210         call MPI_BARRIER(comm_cart,ierr)
211
212         !Reorder the particle in the process
213         do p =1,pmax_nb(0) !loop over the particles
214             !If the particle is removed from this process
215             if (ap(pmax_nb(0)-p+1)%mslv .lt. 0) then !If the particle number is negative then
216
217                 !Move all following particle one element up
218                 do p2 = pmax_nb(0)-p+1,(nmax-1)
219                     ap(p2)%mslv = ap(p2+1)%mslv
220                     ap(p2)%x = ap(p2+1)%x
221                     ap(p2)%y = ap(p2+1)%y
222                     ap(p2)%z = ap(p2+1)%z
223                     ap(p2)%vx = ap(p2+1)%vx
224                     ap(p2)%vy = ap(p2+1)%vy
225                     ap(p2)%vz = ap(p2+1)%vz
```

```
226                     ap(p2)%vpx = ap(p2+1)%vpx
227                     ap(p2)%vpy = ap(p2+1)%vpy
228                     ap(p2)%vpz = ap(p2+1)%vpz
229                     ap(p2)%ax = ap(p2+1)%ax
230                     ap(p2)%ay = ap(p2+1)%ay
231                     ap(p2)%az = ap(p2+1)%az
232                 enddo
233             endif
234         enddo
235
236         !Wait for all processes to finish
237         call MPI_BARRIER(comm_cart,ierr)
238
239         !Determine if a new particle has arrived in this process
240         do nbrec = 1,8
241             nbrec2 = nbrec + 4
242             if (nbrec2 .gt. 8) nbrec2 = nbrec2 - 8
243             do i=1,pmax_nb(nbrec)           !for neighbor nbrec, check all its particles = pmax_nb(nbrec)
244                 if (newmaster_nb(i,nbrec) .eq. nbrec2)  then !check wheter the particle went to this process
245
246                     tag = mslv_nb(i,nbrec)              !tag used to recieve the particle
247
248                     !Recieve the data for the new particle in this process
249                     call MPI_RECV(npmslv,1,MPI_INTEGER,neighbor(nbrec),tag     ,comm_cart,status,ierr)
250                     call MPI_RECV(npx   ,1,MPI_REAL8  ,neighbor(nbrec),tag+1*n ,comm_cart,status,ierr)
251                     call MPI_RECV(npy   ,1,MPI_REAL8  ,neighbor(nbrec),tag+2*n ,comm_cart,status,ierr)
252                     call MPI_RECV(npz   ,1,MPI_REAL8  ,neighbor(nbrec),tag+3*n ,comm_cart,status,ierr)
253                     call MPI_RECV(npvx  ,1,MPI_REAL8  ,neighbor(nbrec),tag+4*n ,comm_cart,status,ierr)
254                     call MPI_RECV(npvy  ,1,MPI_REAL8  ,neighbor(nbrec),tag+5*n ,comm_cart,status,ierr)
255                     call MPI_RECV(npvz  ,1,MPI_REAL8  ,neighbor(nbrec),tag+6*n ,comm_cart,status,ierr)
256                     call MPI_RECV(npvpx ,1,MPI_REAL8  ,neighbor(nbrec),tag+7*n ,comm_cart,status,ierr)
257                     call MPI_RECV(npvpy ,1,MPI_REAL8  ,neighbor(nbrec),tag+8*n ,comm_cart,status,ierr)
258                     call MPI_RECV(npvpz ,1,MPI_REAL8  ,neighbor(nbrec),tag+9*n ,comm_cart,status,ierr)
259                     call MPI_RECV(npax  ,1,MPI_REAL8  ,neighbor(nbrec),tag+10*n,comm_cart,status,ierr)
260                     call MPI_RECV(npay  ,1,MPI_REAL8  ,neighbor(nbrec),tag+11*n,comm_cart,status,ierr)
261                     call MPI_RECV(npaz  ,1,MPI_REAL8  ,neighbor(nbrec),tag+12*n,comm_cart,status,ierr)
262
263
264                     !Place the data in the correct position so that the particle numbers are ascending
265                     do j =1,pmax
266                         if (npmslv .lt. ap(pmax-j+1)%mslv) then
267                             ap(pmax-j+1+1)%mslv = ap(pmax-j+1)%mslv
268                             ap(pmax-j+1+1)%x = ap(pmax-j+1)%x
269                             ap(pmax-j+1+1)%y = ap(pmax-j+1)%y
270                             ap(pmax-j+1+1)%z = ap(pmax-j+1)%z
271                             ap(pmax-j+1+1)%vx = ap(pmax-j+1)%vx
272                             ap(pmax-j+1+1)%vy = ap(pmax-j+1)%vy
273                             ap(pmax-j+1+1)%vz = ap(pmax-j+1)%vz
274                             ap(pmax-j+1+1)%vpx = ap(pmax-j+1)%vpx
275                             ap(pmax-j+1+1)%vpy = ap(pmax-j+1)%vpy
276                             ap(pmax-j+1+1)%vpz = ap(pmax-j+1)%vpz
277                             ap(pmax-j+1+1)%ax = ap(pmax-j+1)%ax
278                             ap(pmax-j+1+1)%ay = ap(pmax-j+1)%ay
279                             ap(pmax-j+1+1)%az = ap(pmax-j+1)%az
280
281                         elseif (npmslv .gt. ap(pmax-j+1)%mslv) then
282                             ap(pmax-j+1+1)%mslv = npmslv
283                             ap(pmax-j+1+1)%x = npx
284                             ap(pmax-j+1+1)%y = npy
285                             ap(pmax-j+1+1)%z = npz
286                             ap(pmax-j+1+1)%vx = npvx
287                             ap(pmax-j+1+1)%vy = npvy
288                             ap(pmax-j+1+1)%vz = npvz
289                             ap(pmax-j+1+1)%vpx = npvpx
290                             ap(pmax-j+1+1)%vpy = npvpy
291                             ap(pmax-j+1+1)%vpz = npvpz
292                             ap(pmax-j+1+1)%ax = npax
293                             ap(pmax-j+1+1)%ay = npay
294                             ap(pmax-j+1+1)%az = npaz
295                             exit
296                         endif
297
298
299                     enddo
300
301                     if (npmslv .lt. ap(1)%mslv) then
302                         ap(1)%mslv = npmslv
303                         ap(1)%x = npx
304                         ap(1)%y = npy
305                         ap(1)%z = npz
306                         ap(1)%vx = npvx
307                         ap(1)%vy = npvy
308                         ap(1)%vz = npvz
309                         ap(1)%vpx = npvpx
```

```
                        ap(1)%vpy = npvpy
                        ap(1)%vpz = npvpz
                        ap(1)%ax = npax
                        ap(1)%ay = npay
                        ap(1)%az = npaz
                    endif

                    !Add one particle to the total number of particles in this process
                    pmax = pmax+1
                endif
            enddo
        enddo

        !Wait for all processes to finish
        call MPI_BARRIER(comm_cart,ierr)

        !Check wheter a particle was tagged wrongly
        do p = 1,pmax
            if (ap(p)%mslv .lt. 0) then
                write(n+3,*) 'Error particle', ap(p)%mslv, 'is negative', p, my_rank
                do p2 = 1,pmax
                    write(n+3,*) ap(p2)%mslv
                enddo
                stop !Stop if there is a negative particle number, something went wrong with the communication
            endif
        enddo

        !Reset conditions for particle properties
        ap(1:nmax)%magmx = 0.
        ap(1:nmax)%magmy = 0.
        ap(1:nmax)%magmz = 0.
        ap(1:nmax)%magfx = 0.
        ap(1:nmax)%magfy = 0.
        ap(1:nmax)%magfz = 0.
        ap(1:nmax)%colfx = 0.
        ap(1:nmax)%colfy = 0.
        ap(1:nmax)%colfz = 0.
        ap(1:nmax)%hydfx = 0.
        ap(1:nmax)%hydfy = 0.
        ap(1:nmax)%hydfz = 0.

        !Check if no particles are missing or added
        call MPI_ALLREDUCE(pmax,pmax_all,1,MPI_INTEGER,MPI_SUM,COMM_CART,ierr)
        if (my_rank .eq. 0) then
            if (pmax_all .ne. n) then
                print*, 'Particles missing'
                stop    !stop if there are particles missing
            endif
        endif

        !Wait for all processes to finish
        call MPI_BARRIER(comm_cart,ierr)

    end subroutine new_masters
end module
```

main.f90

```
program main
    !This is the main file to simulate the behavior of an MRF

    use mod_param        !Load the parameters
    use mod_common       !Load the common variables
    use mod_common_mpi   !Load the common mpi variables
    use mod_initmpi      !Load the initmpi file
    use mod_initpart     !Load the initpart file
    use mod_forceint     !Load the forceint file
    use mod_kinetic      !Load the kinetic file
    use mod_verlet       !Load the verlet file
    use mod_framewrite   !Load the framewrite file

    implicit none

    integer :: irc !Used to communicate the finalization of the mpi
    real(8) ::  startsim, &     !Start timer for the simulation
                finishsim, &    !Finish the times for the simulation
                start, &        !Start timer for one step
                finish1, &      !Finish timer for one step before estimatimg the time left
                finish2, &      !Finish times for one step after  estimating the time left
                test, &         !The estimated time left
                perc            !The percentage done
    real(8), dimension(steps) :: time   !Time vector

    !format for the output of the estimated time left
```

```
27      10 format (a,f8.2,a,f6.2,a)
28
29      !Initialize MPI
30      call MPI_INIT( ierr )
31
32      !start timer to determine the total simulation time
33      startsim = MPI_WTIME()
34
35      !Initialize the processes
36      call initmpi
37
38      if (my_rank .eq. 0) then !Only one processes write and opens certain files
39
40          !Open data files
41          open(unit=n+1,file='./out/Ek.dat' ,status='replace') !For the kinetic energies + mean z-height
42          open(unit=n+2,file='./out/tau.dat',status='replace') !For the shear stresses
43          open(unit=n+3,file='./out/output.txt',status='replace') !For output
44          open(unit=n+4,file='./out/Vrel.dat',status='replace')   !For the slab velocities
45
46          !Check wheter each process is big enough to support the cut-off distance = 6*a
47          if (Sry/dims(2) .lt. 6.*a) then
48              write(n+3,*) 'The procces size in y-direction is to small, decrease the number of processes or increase Sry'
49              stop
50          elseif (Srx/dims(1) .lt. 6.*a) then
51              write(n+3,*) 'The procces size in y-direction is to small, decrease the number of processes or increase Sry'
52              stop
53          endif
54
55          !Check for the boundary conditions
56          if (Sry .lt. 2.*6.*a) then
57              write(n+3,*) 'Increase Sry, PBC does not hold'
58              stop
59          elseif (Srx .lt. 2.*6.*a) then
60              write(n+3,*) 'Increase Srx, PBC does not hold'
61              stop
62          endif
63
64          !Write the number of particles to the output
65          write(n+3,*) 'Number of particles:', n
66
67          !Write the simulation domain to the output
68          write(n+3,*) 'Srx/(2.*a)', 'Sry/(2.*a)', 'Srz/(2.*a)'
69          write(n+3,*) Srx/(2.*a), Sry/(2.*a), Srz/(2.*a)
70      endif
71
72      !Generate particles
73      call initpart
74
75      !Determine the forces on the particles and the acceleration of the particles
76      call forceint
77
78      !Determine the kinetic energy in the fluid
79      call kinetic(t)
80
81      !Generate a frame
82      call framewrite
83
84      !======= MAIN LOOP ===========
85      do step=1,steps !Loop over the steps until t = tfinal
86
87          !start timer to determine the time a step took
88          start = MPI_WTIME()
89
90          !Do the verlet-integration for this step
91          call verlet
92
93          !Write the data to dat files
94          call kinetic(t)
95
96          if (mod(step,intsave) .eq. 0) then !Only Nframes times per simulation
97              !Create a frame
98              call framewrite
99          endif
100
101         !Stop time
102         finish1 = MPI_WTIME()
103
104         !Determine the estimated time for this step
105         time(step) = finish1-start
106
107         !Determine the estimated time left
108         test = sum(time)/(step-1.)*(steps-step)/60.
109
110         !Determine the percentage of the simulation that is done
```

```
111         perc = real(step)/real(steps)*100.
112
113         !Print the estimated time and the percentage
114         if (my_rank .eq. 0) then
115             write(n+3,10) 'Estimated time left of : ', test,' / min @ ',perc,'%'
116         endif
117
118         !Stop time 2
119         finish2 = MPI_WTIME()
120
121         !Determine the real time for this step
122         time(step) = finish1-start
123
124     enddo
125
126     !finish the time for the total simulation time
127     finishsim = MPI_WTIME()
128
129     !Print the total time for this simulation
130     if (my_rank .eq. 0) then
131         write(n+3,*) 'Simulation time', (finishsim-startsim)
132     endif
133
134     !Wait for all processes to finish
135     call MPI_BARRIER(comm_cart,ierr)
136
137     !Finilize MPI
138     call MPI_FINALIZE(irc)
139
140 end
```

F.1.2. Non-uniform magnetic field

The current model can only be used to determine the behavior of a MRF in a uniform magnetic field. Replacing the uniform field to a non-uniform magnetic field is thus required. Firstly, the file param.f90 is changed to include the parameters of the bar magnet. The following lines of code are used to replace lines 43 and 47.

```
    !Magnet parameters
    real(8),parameter :: w = 2.5D-3, !0.5 times the x dimension of the magnet
                         h =  2.5D-3, !0.5 times the y dimension of the magnet
                         l = 1.D-3, !0.5 times the z dimension of the magnet
                         B_r = 1.45 !remanence of the magnet

    !Displacement of the bar manget, to make sure the center of the magnet is at the
    !correct location
    real(8),dimension(3),parameter :: mdisp = (/-Srx/2., -Sry/2. - h, l + 2.D-3/)
```

The magnetic field generated by the permanent magnet is computed by field.f90 for which the source code is given in appendix E.1.3.

The magnetic moment of the particles is initially set in initpart.f90, therefore line 50 to 52 should be changed to:

```
                call magms(xglob(p)+mdisp(1),yglob(p)+mdisp(2),zglob(p)+mdisp(3),&
                           magmxglob(p),magmyglob(p),magmzglob(p))
```

Furthermore, the following line should be added between line 6 and 8:

```
    use mod_field !Load the file to determine the magnetic field
```

The magnetic gradient force should be added to forceint.f90 and forcepred.f90. The following line of code should be added between line 6 and 8 to include the field.f90 file

```
    use mod_field !Load the file to determine the magnetic field
```

Then between line 35 and 37 the variables for this force need to be declared:

```
    real(8) :: gradFx, gradFy, gradFz !Declare the variables for the magnetic gradient force
```

Furthermore, line 237 and 239 of both files should be replaced by the following to include the gradient force:

```
            !External magnetic field gradient force
            call gmBs(ap(p)%x+mdisp(1),ap(p)%y+mdisp(2),ap(p)%z+mdisp(3),gradFx,gradFy,gradFz)
            ap(p)%magfx = ap(p)%magfx + gradFx
            ap(p)%magfy = ap(p)%magfy + gradFy
            ap(p)%magfz = ap(p)%magfz + gradFz

            !Addition to the external shear stress
            Ftauext = Ftauext + gradFy
```

The magnetization in the main loop is set in verlet.f90. Therefore, the field.f90 file should be loaded by adding the following code between line 8 and 10 of verlet.f90:

```
    use mod_field !Load the file to determine the magnetic field
```

Then, in order to make sure the particles have the correct magnetic moment, line 51 to 53 should be replaced by:

```
            call magms(ap(p)%x+mdisp(1),ap(p)%y+mdisp(2),ap(p)%z+mdisp(3),ap(p)%magmx,ap(p)%magmy,ap(p)%magmz)
```

Finally, in order to save the field produced by the bar magnet, the following should be added between line 12 and 14 of main.f90:

```
    use mod_field !Load the file to determine the magnetic field
```

The field is then saved by adding the following line between line 69 and 70:

```
        !Store the magnetic field
        call field
```

Now the code can be used to simulate the MRF in a non-uniform magnetic field generated by a bar magnet that is homogeneously polarized in the z-direction.

F.1.3. Compiling and running the code

The Fortran code is compiled using a Makefile, for which GFortran should be installed on your machine. A guide for the installation of GFortran on Windows, Linux, macOS or OpenBSD can be found here [63]. The models are compiled by running the Makefile given below [2]:

Makefile

```
1  FC=mpif90 #fortran compiler
2  FFLAGS= -O3 -std=legacy
3  SRC=param.f90 common.f90 initmpi.f90 initpart.f90 forceint.f90 kinetic.f90 forcepred.f90
       verlet.f90 framewrite.f90 main.f90
4  OBJ=${SRC:.f90=.o} #substitute .f90 with .o
5  NAME=MCF
6
7  %.o: %.f90 #wildcard rule, creation of *.o depends on *.f90
8          $(FC) $(FFLAGS) -o $@ -c $<
9
10 $(NAME): $(OBJ)
11         $(FC) $(FFLAGS) -o $@ $(OBJ)
12
13 clean: #cleans all the old compilation files
14         @rm -f *.mod *.o $(NAME)
```

The Makefile is executed using the command:

```
        make
```

The compilation is cleared by using the command:

```
        make clean
```

It should be noted, that for a non-uniform magnetic field, line 3 should change to:

```
SRC=param.f90 common.f90 initmpi.f90 field.f90 initpart.f90 forceint.f90 kinetic.f90 forcepred.f90 verlet.
    f90 framewrite.f90 main.f90
```

This Makefile will generate a executable that can be used to run the simulation. In order to make sure that the simulation does not give an error, the directories /out and /out/frames should already exist in the working directory.

To run the model on DHPC (Delft High Performance Computing Centre) [58] the jobscript given below is required:

Make sure that the executable has the same name as in line 14, the account is correct and that the correct number of tasks (number of processes) are assigned. ntasks should be equal to the number of processes that are assigned in **param.f90**. More information about the jobscript required to run on DHPC can be found here here [58].

In order to make the executable on DHPC, make sure that you have loaded the following modules:

```
module load 2022r2
module load openmpi
```

A job is submitted to DHPC using:

```
sbatch jobscript
```

F.2. Results

F.2.1. Dynamic evolution

Uniform magnetic field

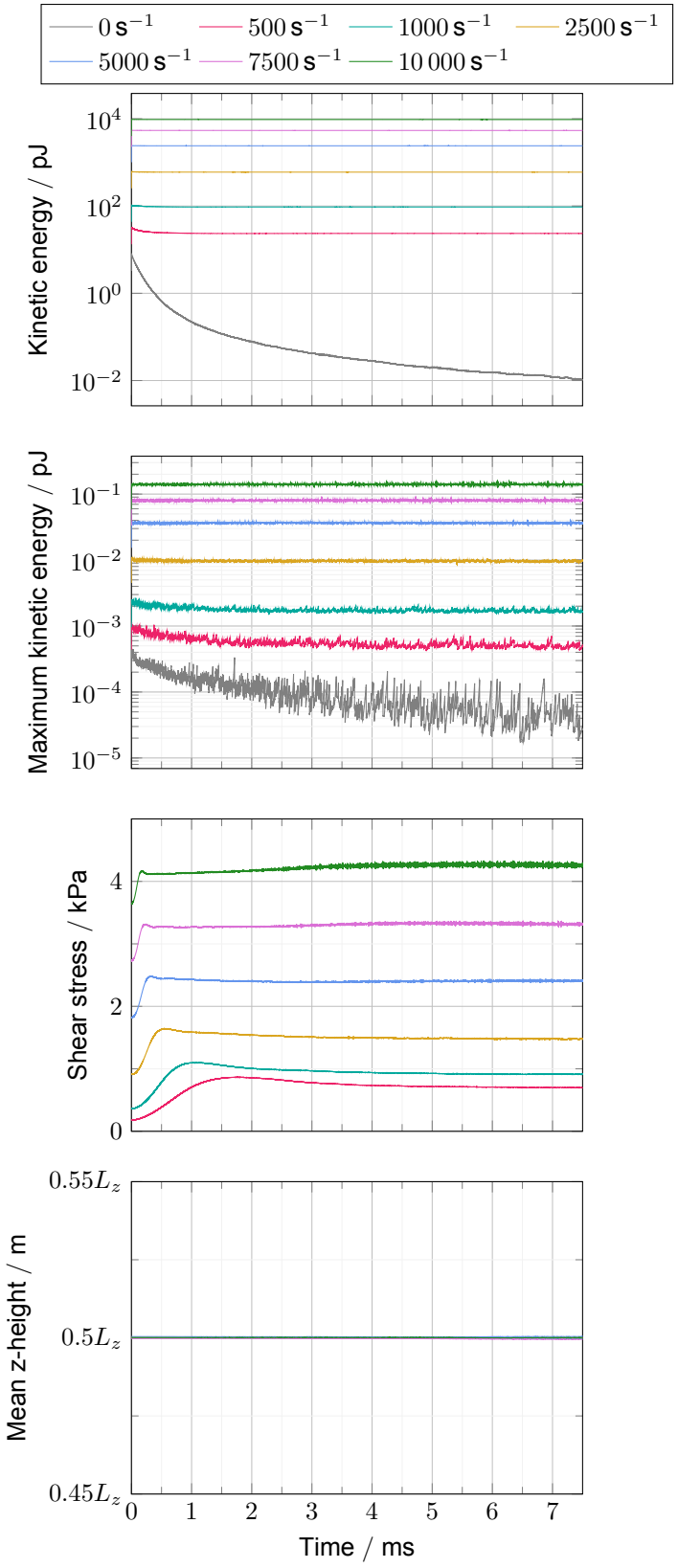


Figure F.1: Dynamic evolution in a uniform magnetic field

Non-uniform: Bar magnet 1

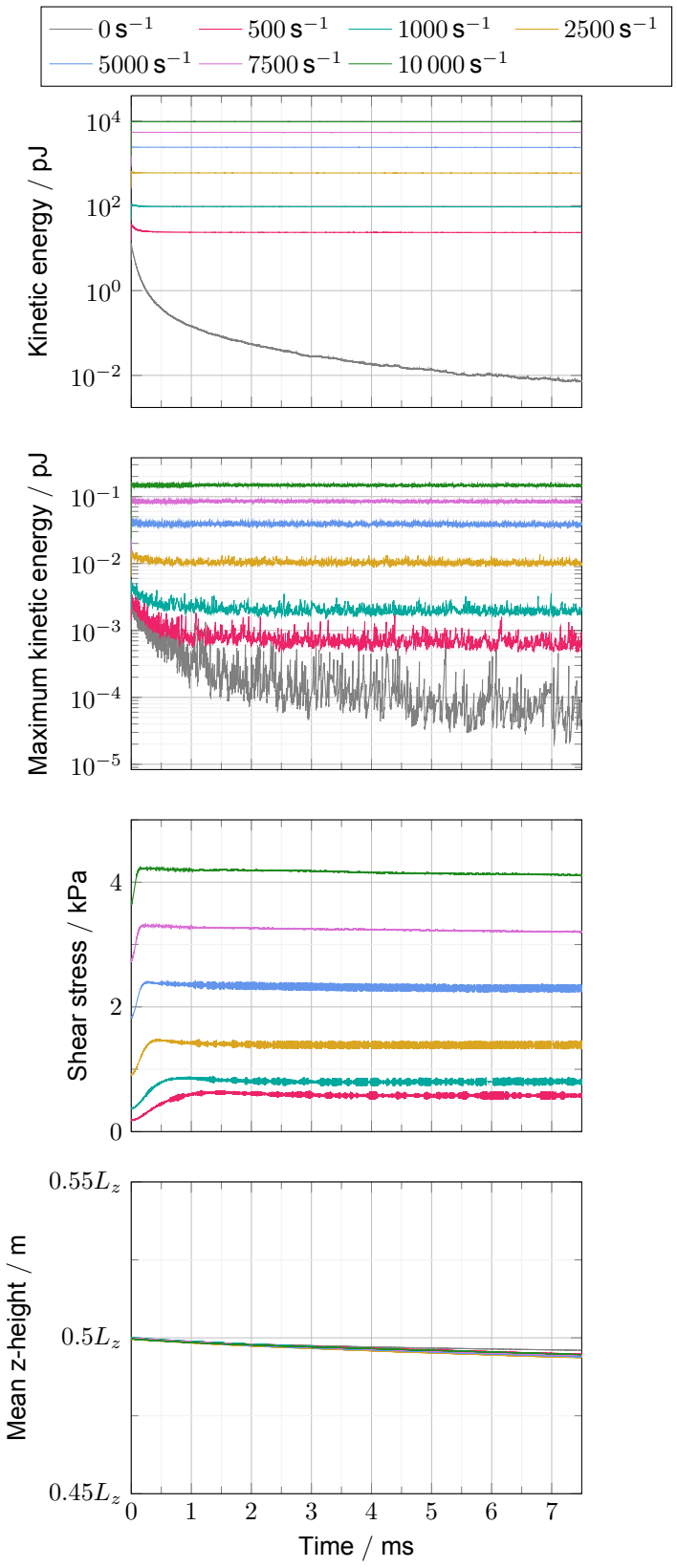


Figure F.2: Dynamic evolution using a $5\,\text{mm} \times 5\,\text{mm} \times 2\,\text{mm}$ (N52) bar magnet

Non-uniform: Bar magnet 2

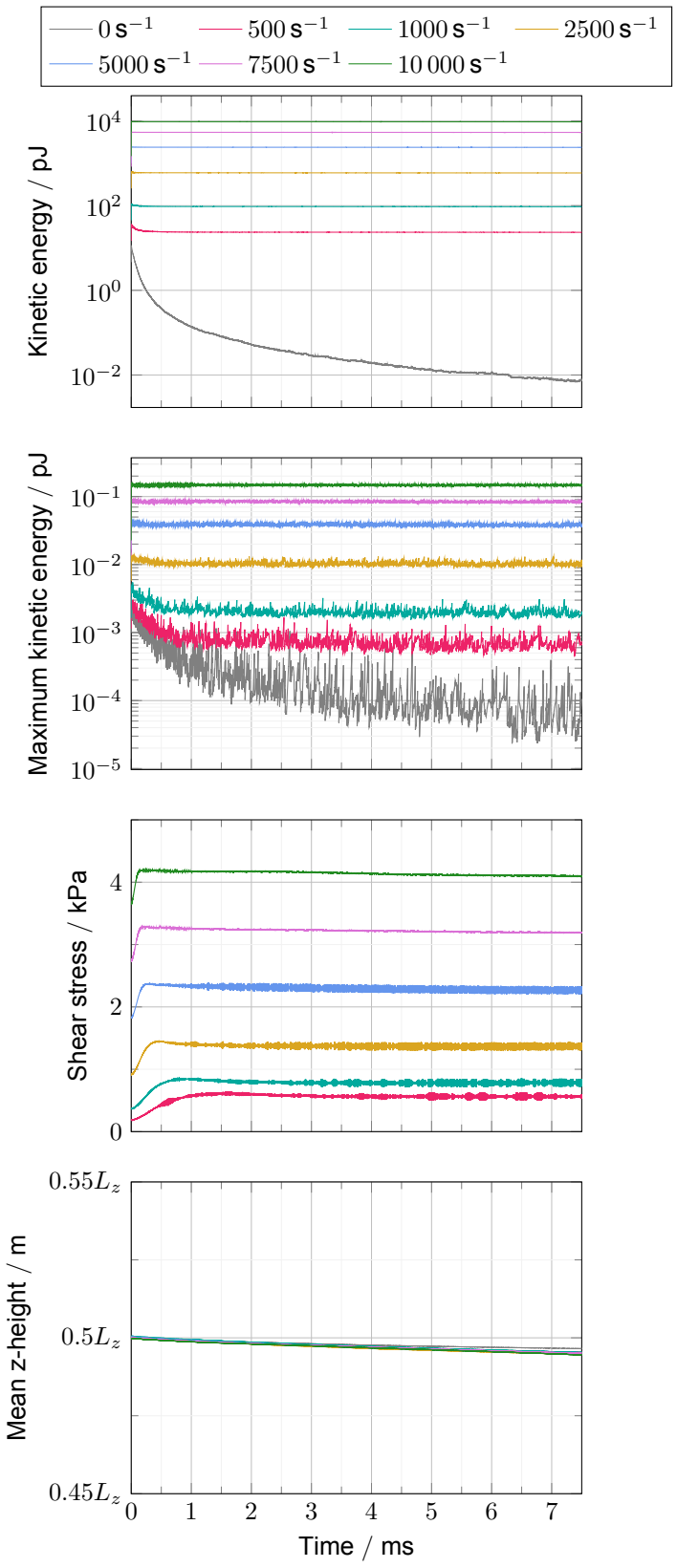


Figure F.3: Dynamic evolution using a $6\,\text{mm} \times 5\,\text{mm} \times 2\,\text{mm}$ (N45) bar magnet

Non-uniform: Bar magnet 3

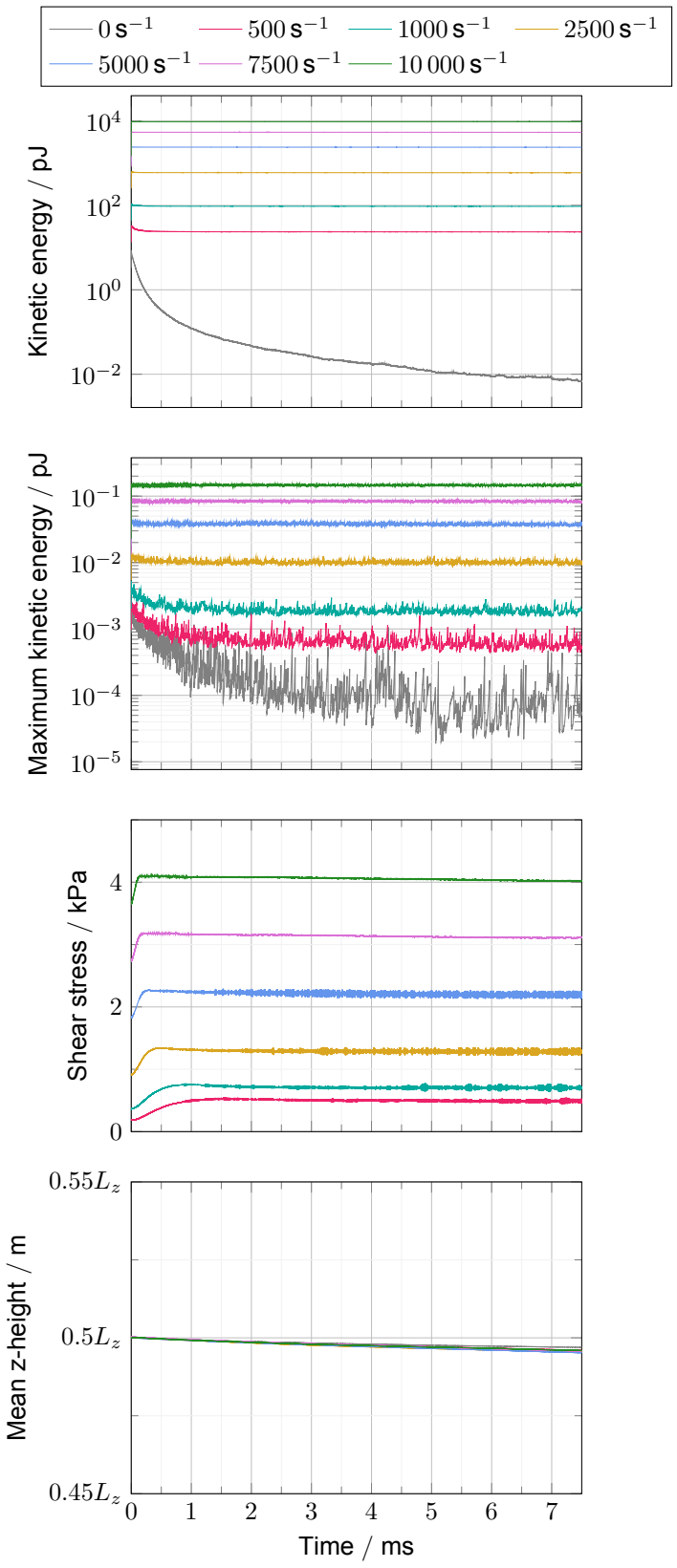


Figure F.4: Dynamic evolution using a $8\,\text{mm} \times 4\,\text{mm} \times 2\,\text{mm}$ (N45) bar magnet

F.2.2. Rheological models

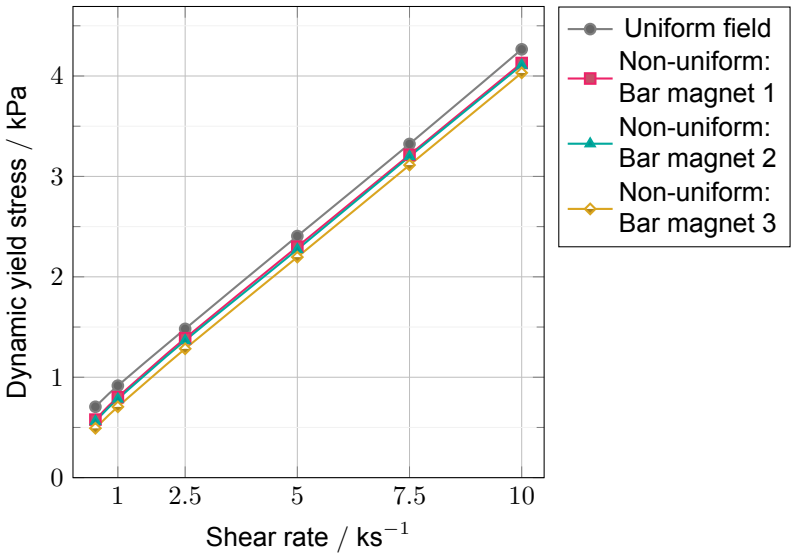


Figure F.5: Rheological models

F.2.3. Slab volume fractions and velocities

Uniform magnetic field

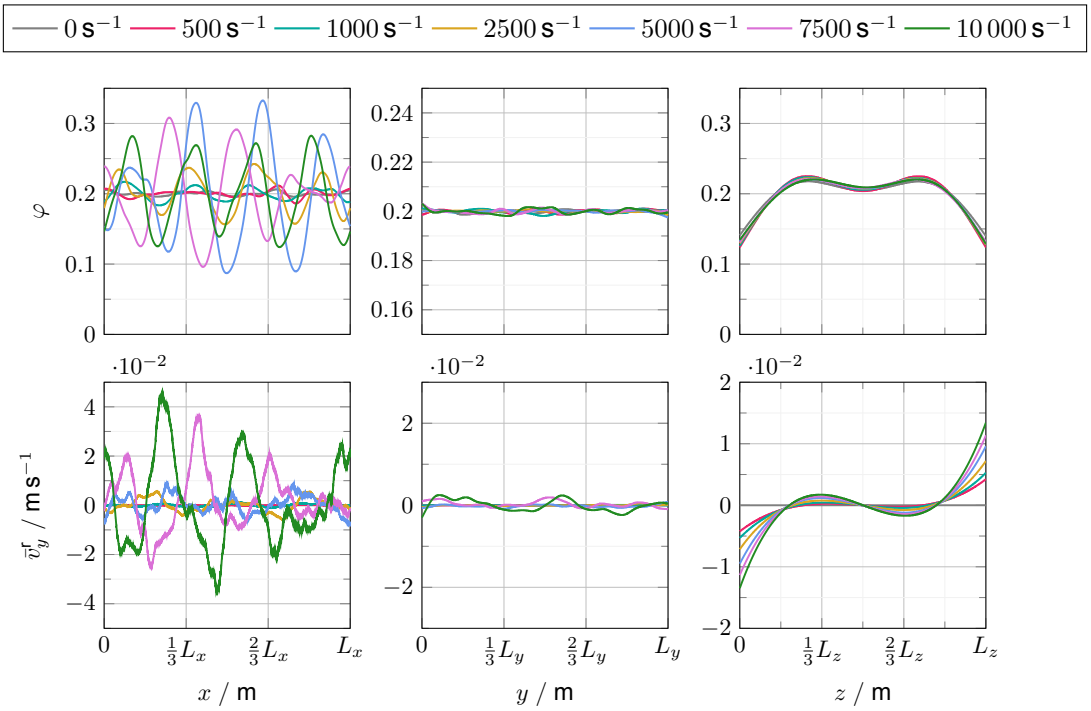


Figure F.6: Slab profiles

Non-uniform: Bar magnet 1

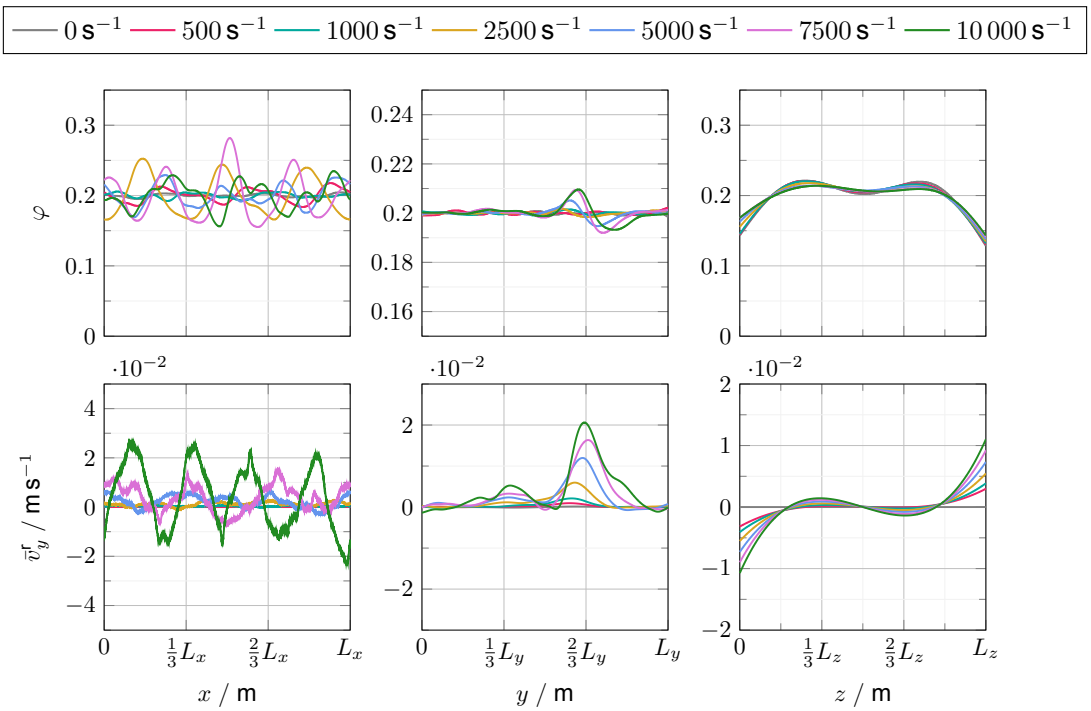


Figure F.7: Slab profiles

Non-uniform: Bar magnet 2

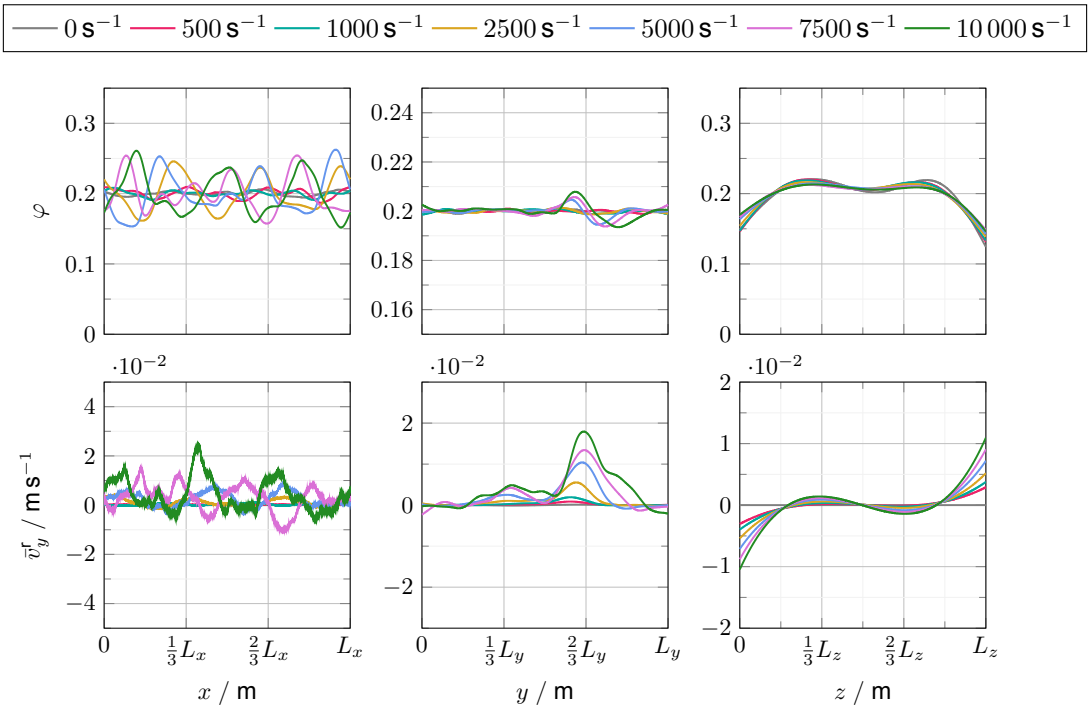


Figure F.8: Slab profiles

Non-uniform: Bar magnet 3

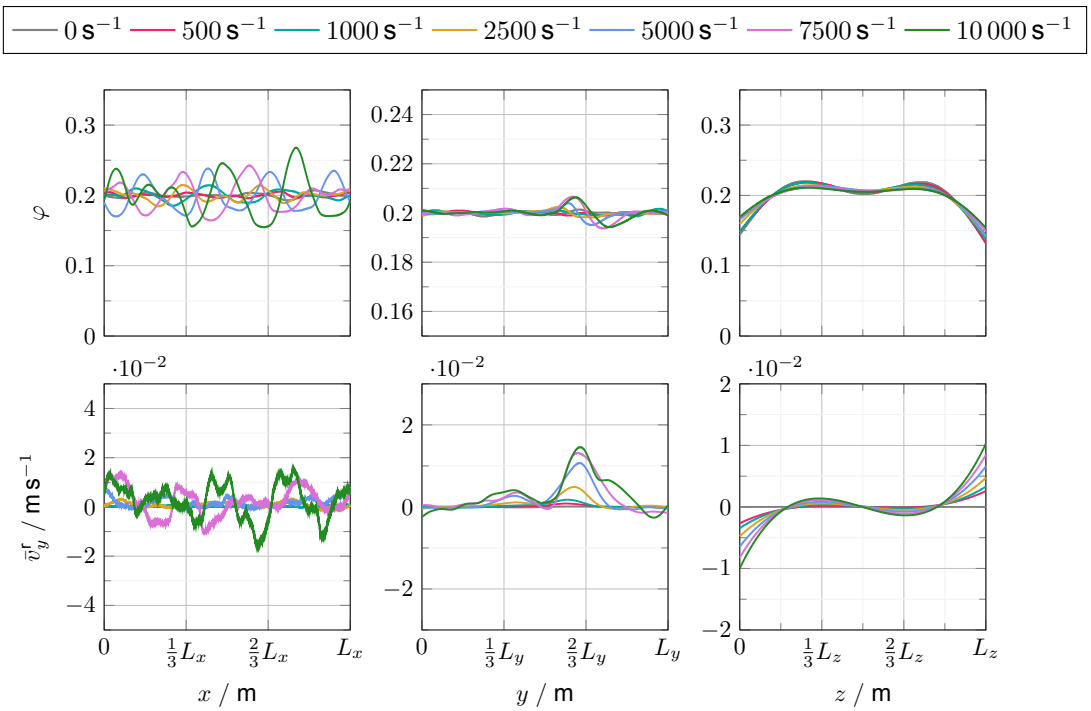
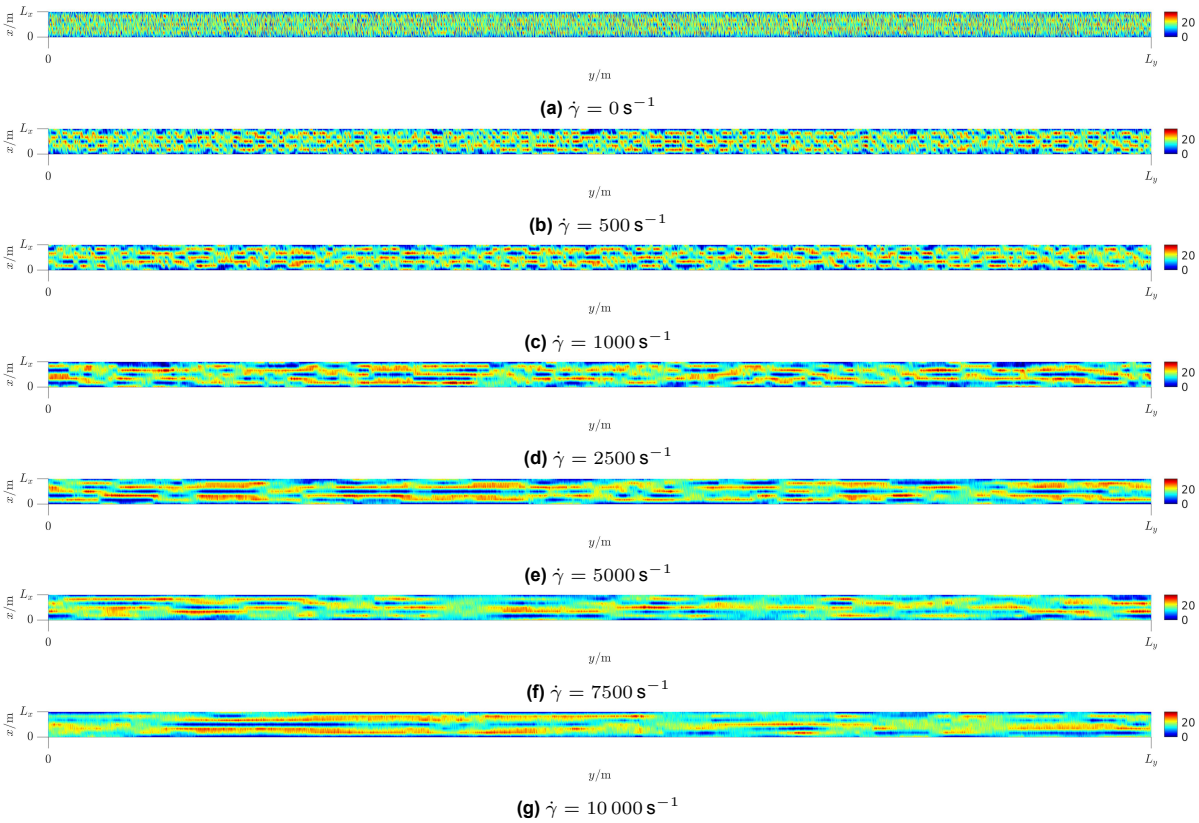


Figure F.9: Slab profiles

F.2.4. Density plots

Uniform magnetic field
yx-plane



(a) $\dot{\gamma} = 0\,\mathrm{s}^{-1}$

(b) $\dot{\gamma} = 500\,\mathrm{s}^{-1}$

(c) $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$

(d) $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$

(e) $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$

(f) $\dot{\gamma} = 7500\,\mathrm{s}^{-1}$

(g) $\dot{\gamma} = 10\,000\,\mathrm{s}^{-1}$

Figure F.10: Density plot
Color bar: Number of particles per $4R \times 3R$ m^2

yz-plane

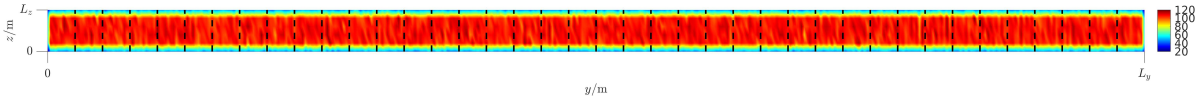
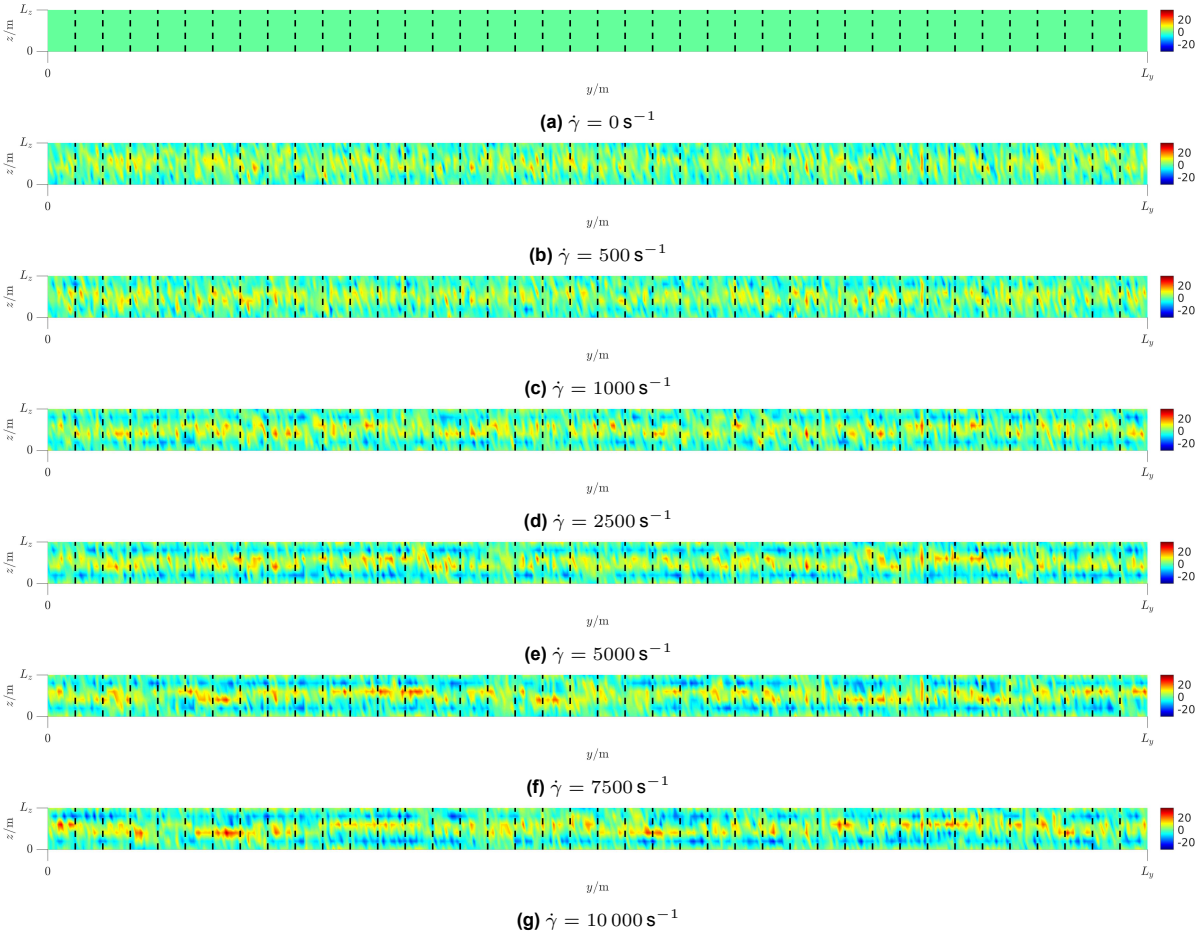


Figure F.11: Density plot when $\dot{\gamma} = 0\,\mathrm{s}^{-1}$
Color bar: Number of particles per $20R \times 6R\,\mathrm{m}^2$
Dashed lines: Magnetic field



(a) $\dot{\gamma} = 0\,\mathrm{s}^{-1}$

(b) $\dot{\gamma} = 500\,\mathrm{s}^{-1}$

(c) $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$

(d) $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$

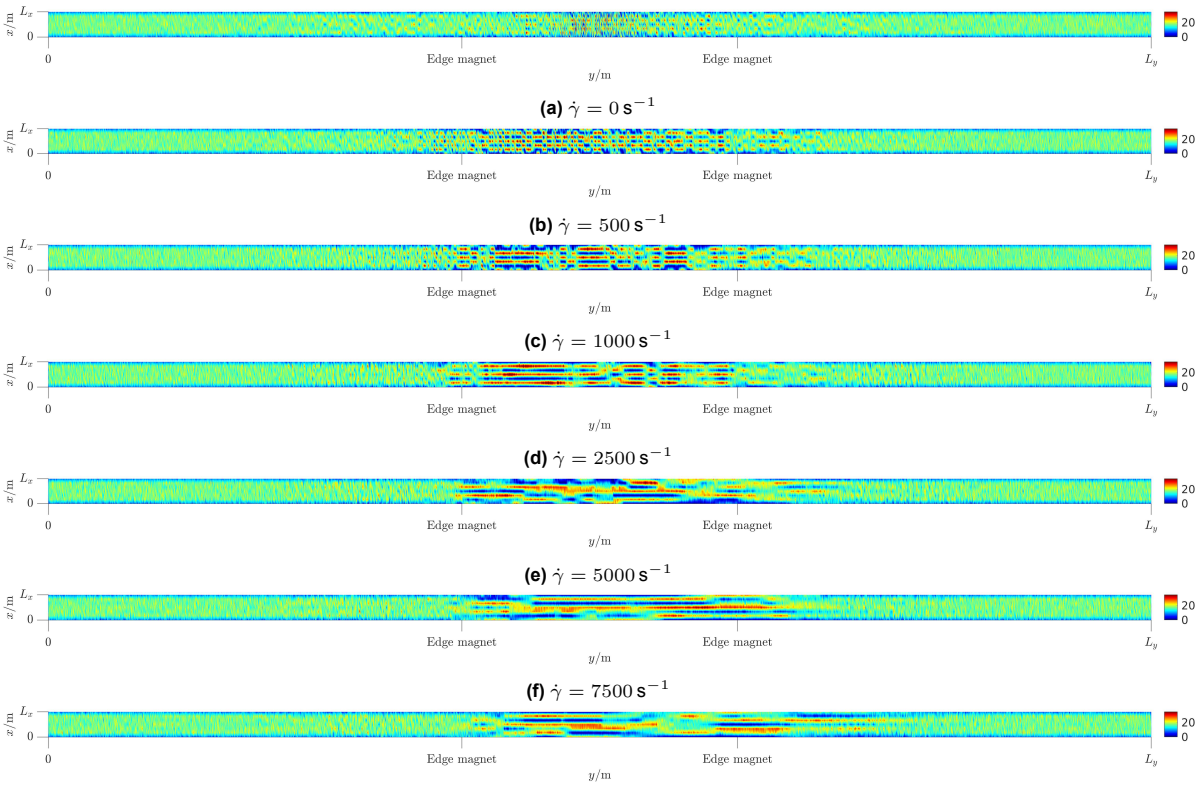
(e) $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$

(f) $\dot{\gamma} = 7500\,\mathrm{s}^{-1}$

(g) $\dot{\gamma} = 10\,000\,\mathrm{s}^{-1}$

Figure F.12: Density differences with respect to figure F.11
Color bar: Difference in number of particles per $20R \times 6R\,\mathrm{m}^2$
Dashed lines: Magnetic field

Non-uniform: Bar magnet 1
yx-plane



(a) $\dot{\gamma} = 0\,\mathrm{s}^{-1}$

(b) $\dot{\gamma} = 500\,\mathrm{s}^{-1}$

(c) $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$

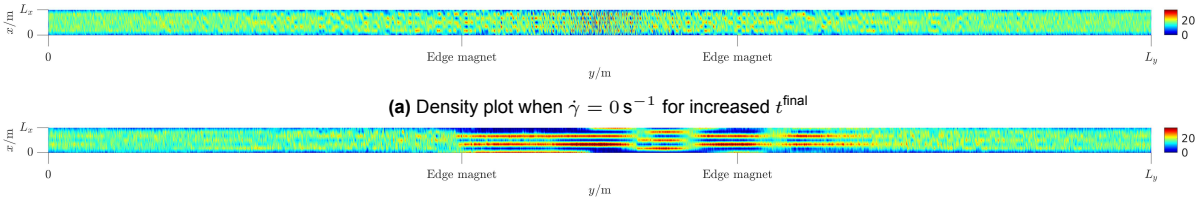
(d) $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$

(e) $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$

(f) $\dot{\gamma} = 7500\,\mathrm{s}^{-1}$

(g) $\dot{\gamma} = 10\,000\,\mathrm{s}^{-1}$

Figure F.13: Density plot
Color bar: Number of particles per $4R \times 3R$ m^2



(a) Density plot when $\dot{\gamma} = 0\,\mathrm{s}^{-1}$ for increased t^{final}

(b) Density plot when $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$ for increased t^{final}

Figure F.14: Density plots for increased t^{final}
Color bar: Number of particles per $4R \times 3R$ m^2

yz-plane

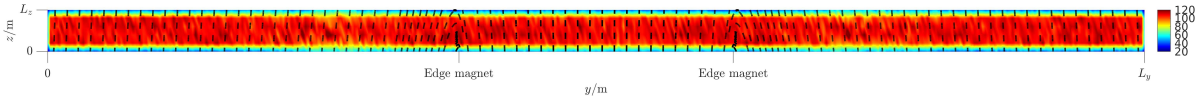
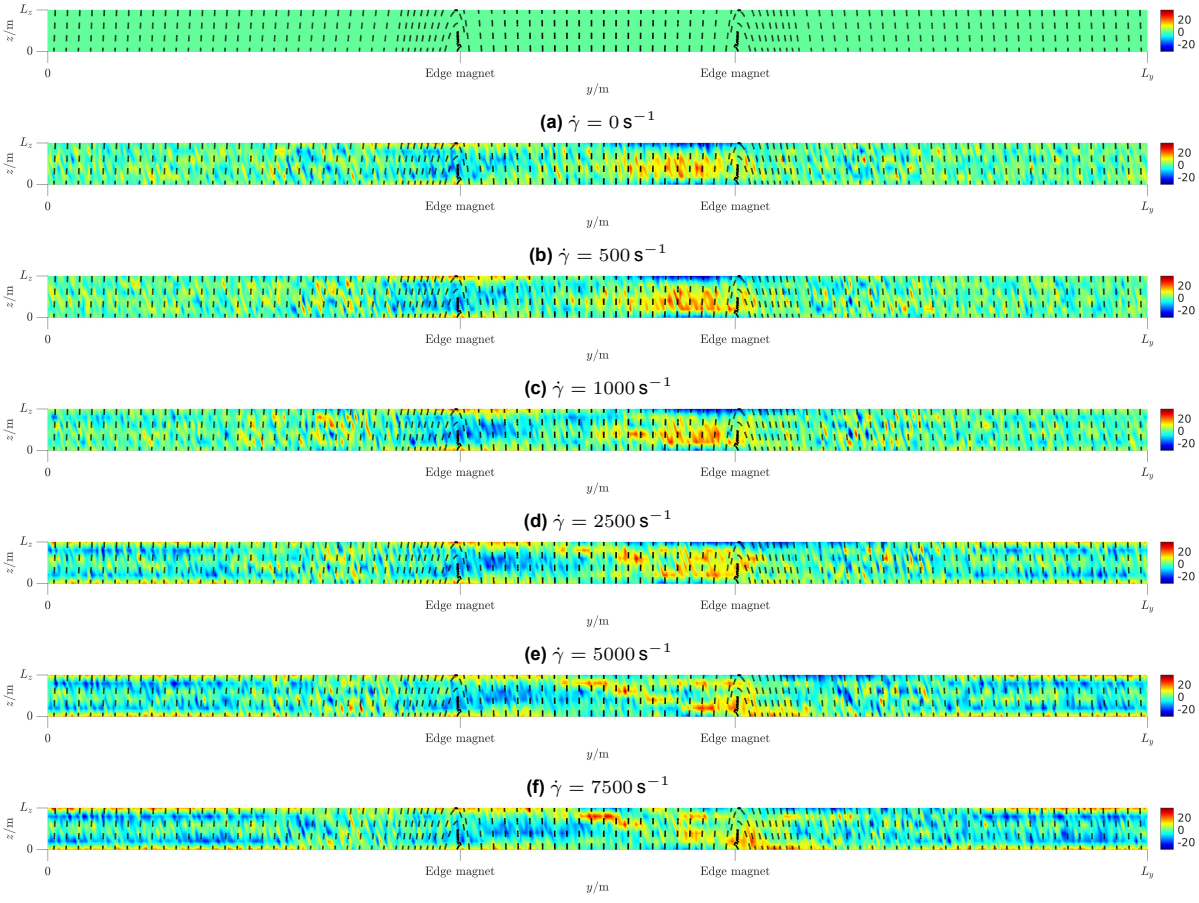


Figure F.15: Density plot when $\dot{\gamma} = 0\,\mathrm{s}^{-1}$
Color bar: Number of particles per $20R \times 6R\,\mathrm{m}^2$
Dashed lines: Magnetic field



(a) $\dot{\gamma} = 0\,\mathrm{s}^{-1}$

(b) $\dot{\gamma} = 500\,\mathrm{s}^{-1}$

(c) $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$

(d) $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$

(e) $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$

(f) $\dot{\gamma} = 7500\,\mathrm{s}^{-1}$

(g) $\dot{\gamma} = 10\,000\,\mathrm{s}^{-1}$

Figure F.16: Density differences with respect to figure F.15
Color bar: Difference in number of particles per $20R \times 6R\,\mathrm{m}^2$
Dashed lines: Magnetic field

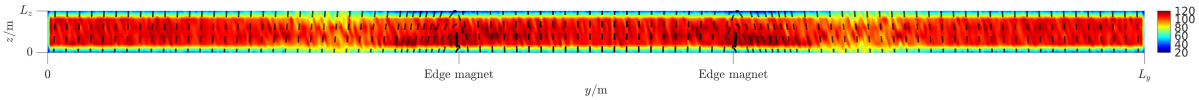


Figure F.17: Density plot when $\dot{\gamma} = 0\,\mathrm{s}^{-1}$ for increased t^{final}
Color bar: Number of particles per $20R \times 6R\,\mathrm{m}^2$
Dashed lines: Magnetic field

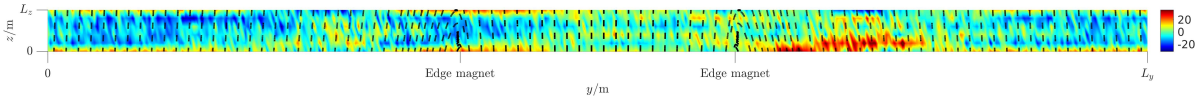
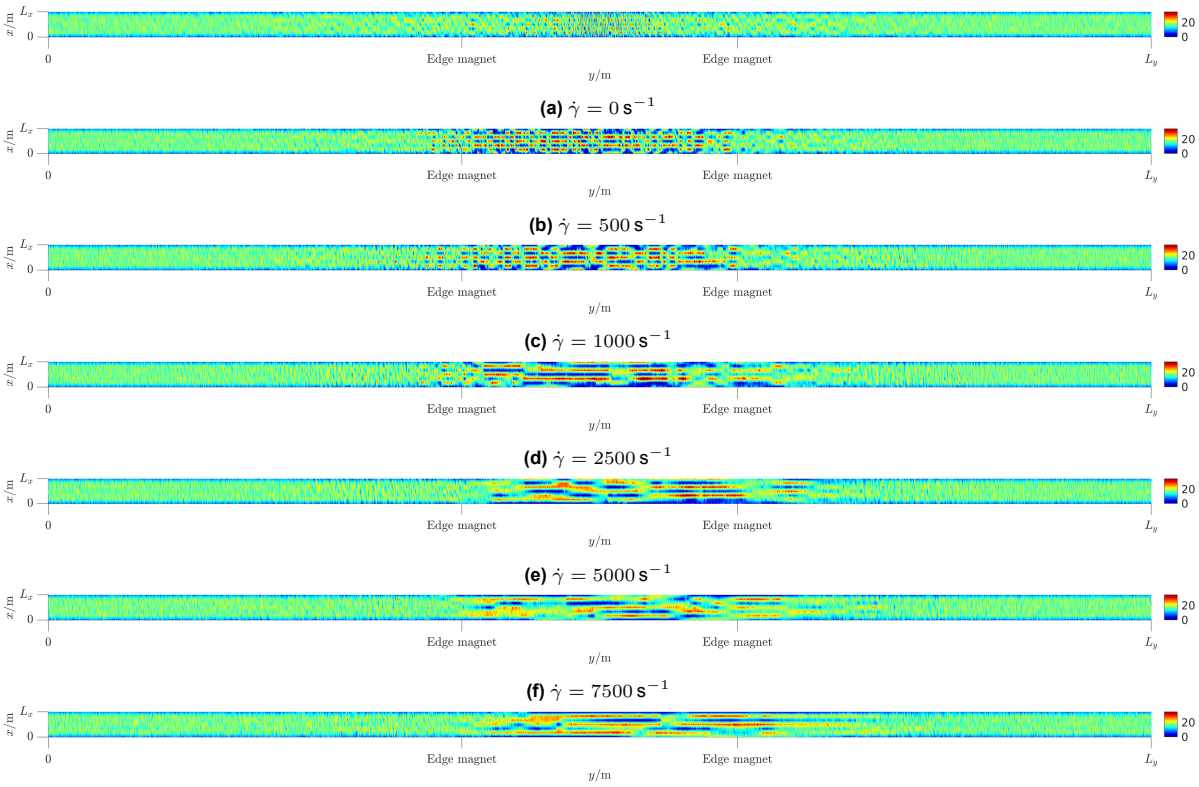


Figure F.18: Density differences with respect to figure F.17 where $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$
Color bar: Difference in number of particles per $20R \times 6R$ m^2
Dashed lines: Magnetic field

Non-uniform: Bar magnet 2
yx-plane



(a) $\dot{\gamma} = 0\,\mathrm{s}^{-1}$

(b) $\dot{\gamma} = 500\,\mathrm{s}^{-1}$

(c) $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$

(d) $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$

(e) $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$

(f) $\dot{\gamma} = 7500\,\mathrm{s}^{-1}$

(g) $\dot{\gamma} = 10\,000\,\mathrm{s}^{-1}$

Figure F.19: Density plot of the xy-plane
Color bar: Number of particles per $4R \times 3R$ m^2

yz-plane

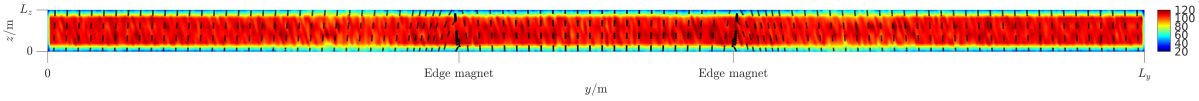
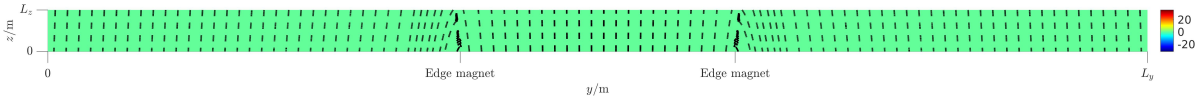
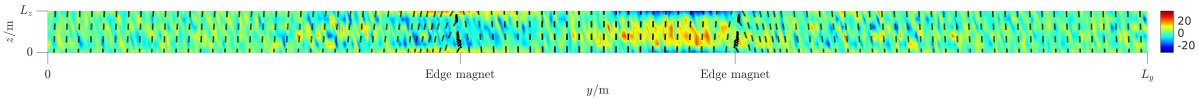


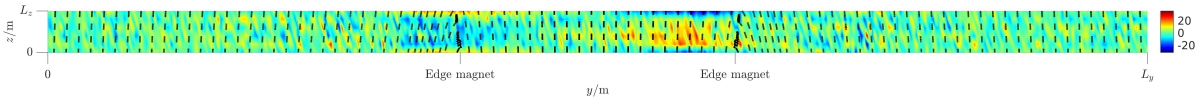
Figure F.20: Density plot when $\dot{\gamma} = 0\,\mathrm{s}^{-1}$
Color bar: Number of particles per $20R \times 6R\,\mathrm{m}^2$
Dashed lines: Magnetic field



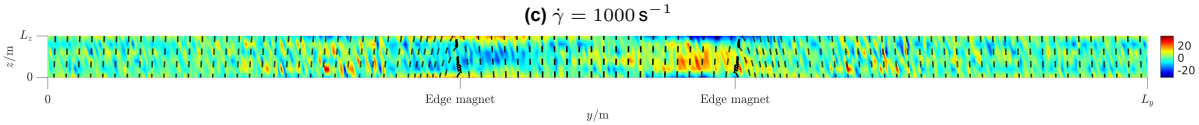
(a) $\dot{\gamma} = 0\,\mathrm{s}^{-1}$



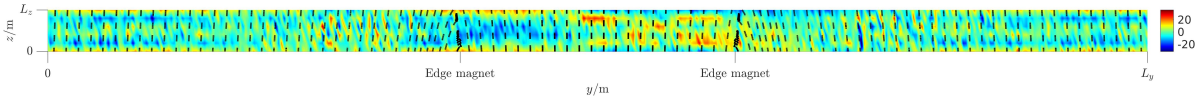
(b) $\dot{\gamma} = 500\,\mathrm{s}^{-1}$



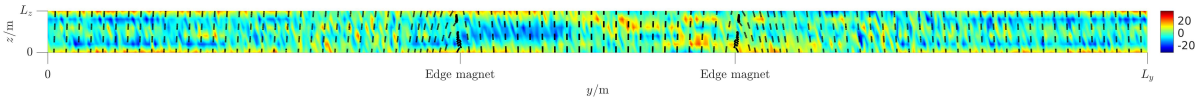
(c) $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$



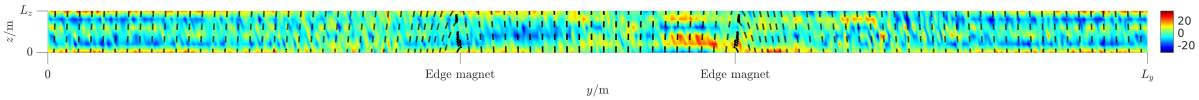
(d) $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$



(e) $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$



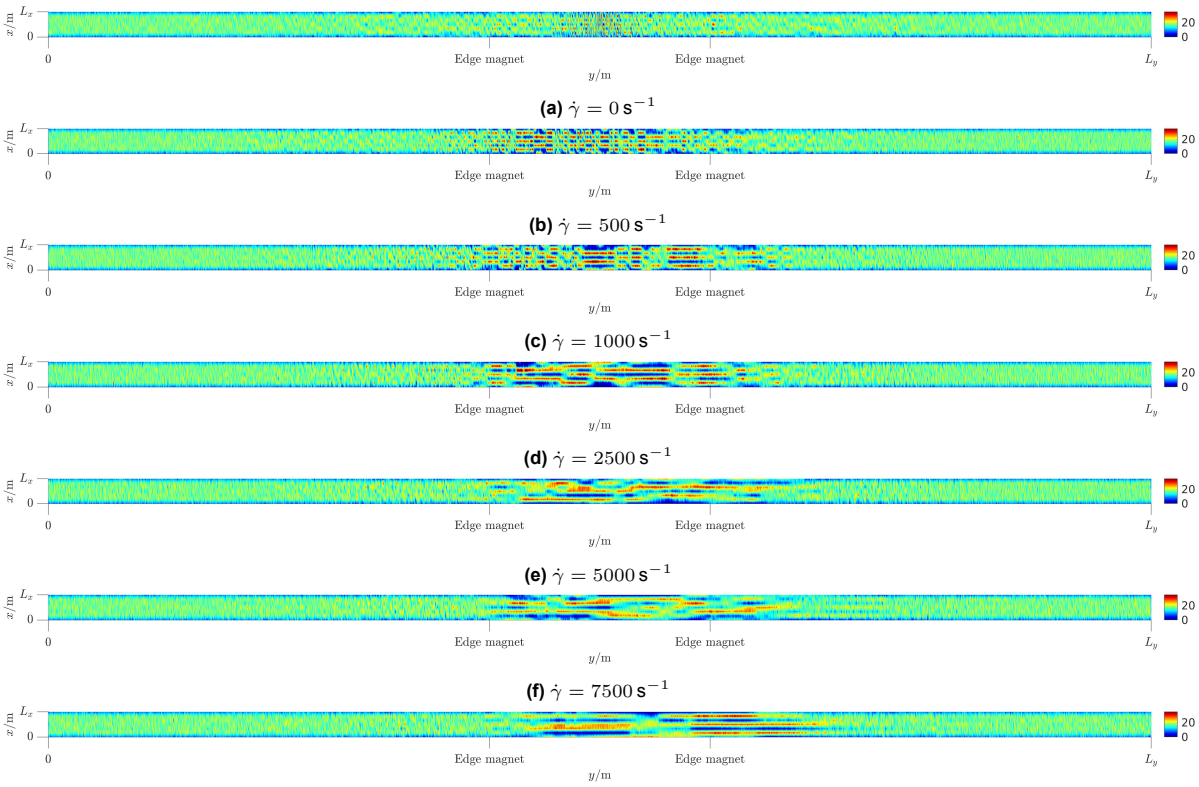
(f) $\dot{\gamma} = 7500\,\mathrm{s}^{-1}$



(g) $\dot{\gamma} = 10\,000\,\mathrm{s}^{-1}$

Figure F.21: Density differences with respect to figure F.20
Color bar: Difference in number of particles per $20R \times 6R\,\mathrm{m}^2$
Dashed lines: Magnetic field

Non-uniform: Bar magnet 3
yx-plane



(a) $\dot{\gamma} = 0\,\mathrm{s}^{-1}$

(b) $\dot{\gamma} = 500\,\mathrm{s}^{-1}$

(c) $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$

(d) $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$

(e) $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$

(f) $\dot{\gamma} = 7500\,\mathrm{s}^{-1}$

(g) $\dot{\gamma} = 10\,000\,\mathrm{s}^{-1}$

Figure F.22: Density plot of the xy-plane
Color bar: Number of particles per $4R \times 3R$ m^2

yz-plane

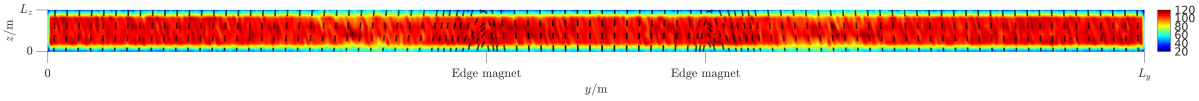
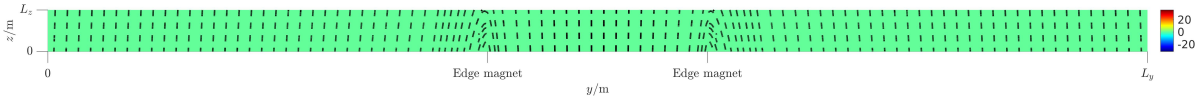
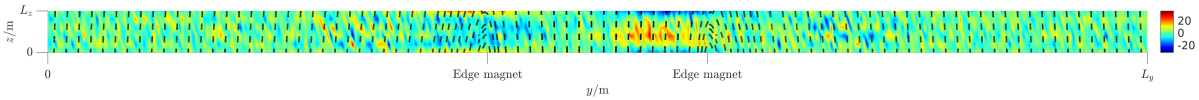


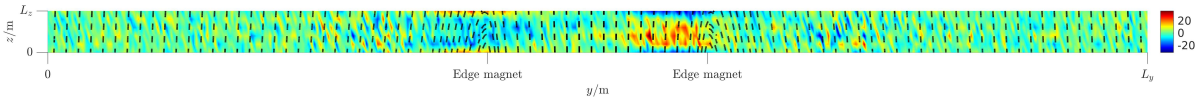
Figure F.23: Density plot when $\dot{\gamma} = 0\,\mathrm{s}^{-1}$
Color bar: Number of particles per $20R \times 6R\,\mathrm{m}^2$
Dashed lines: Magnetic field



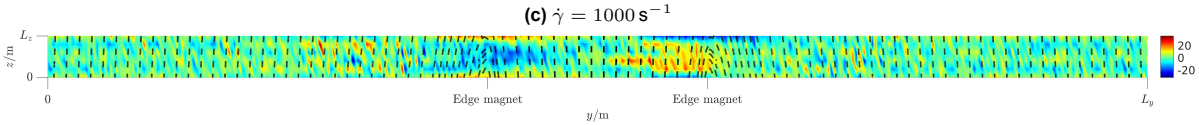
(a) $\dot{\gamma} = 0\,\mathrm{s}^{-1}$



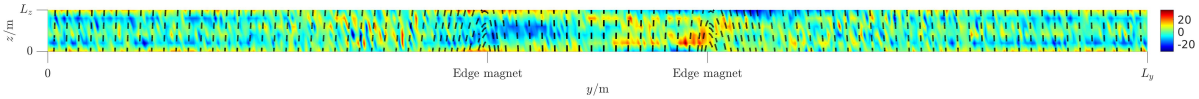
(b) $\dot{\gamma} = 500\,\mathrm{s}^{-1}$



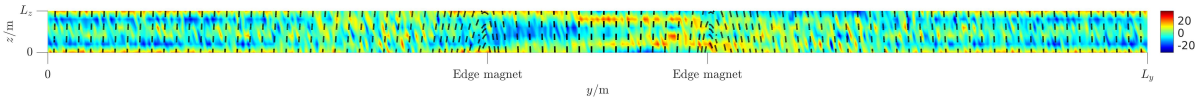
(c) $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$



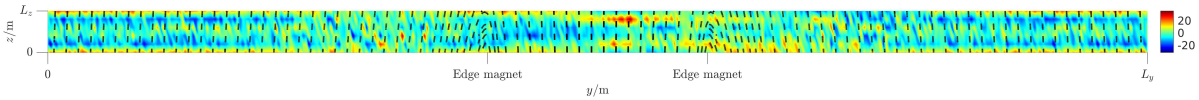
(d) $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$



(e) $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$



(f) $\dot{\gamma} = 7500\,\mathrm{s}^{-1}$



(g) $\dot{\gamma} = 10\,000\,\mathrm{s}^{-1}$

Figure F.24: Density differences with respect to figure F.23
Color bar: Difference in number of particles per $20R \times 6R\,\mathrm{m}^2$
Dashed lines: Magnetic field

‘

G

Breugem's DNS code with magnetic particles

G.1. Results

G.1.1. Steady state structures

G.1.2. $\dot{\gamma} = 500\,\mathrm{s}^{-1}$

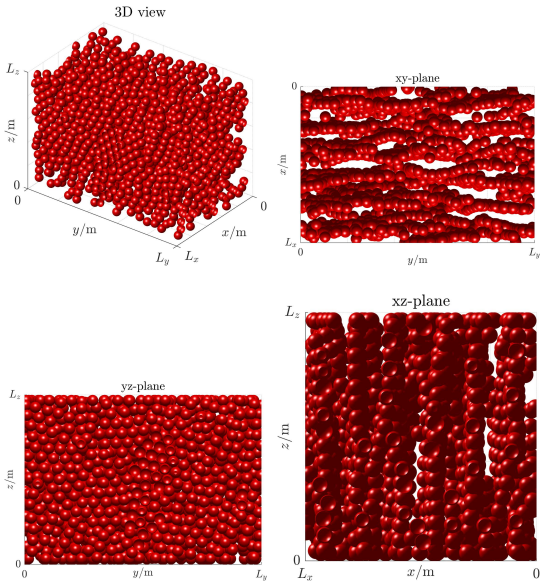


Figure G.1: Steady state structures where $\dot{\gamma} = 500\,\mathrm{s}^{-1}$

G.1.3. $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$

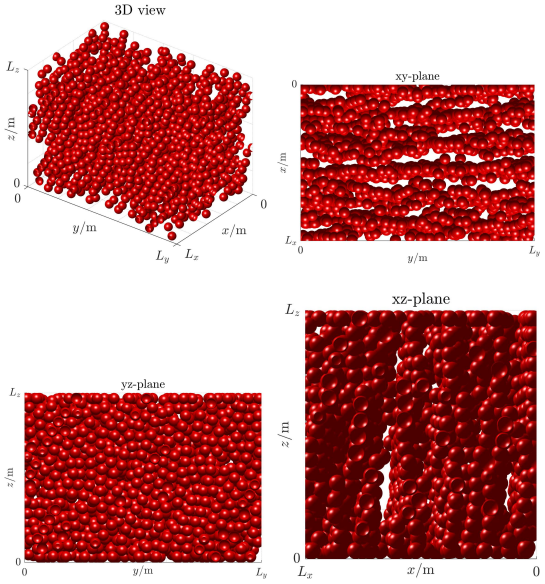


Figure G.2: Steady state structures where $\dot{\gamma} = 1000\,\mathrm{s}^{-1}$

G.1.4. $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$

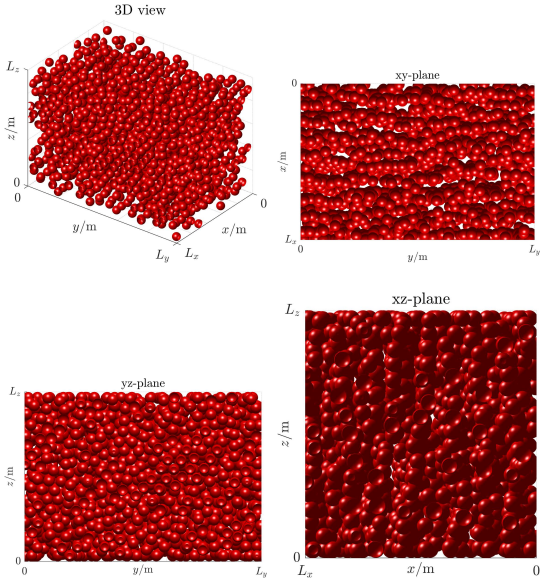


Figure G.3: Steady state structures where $\dot{\gamma} = 2500\,\mathrm{s}^{-1}$

G.1.5. $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$

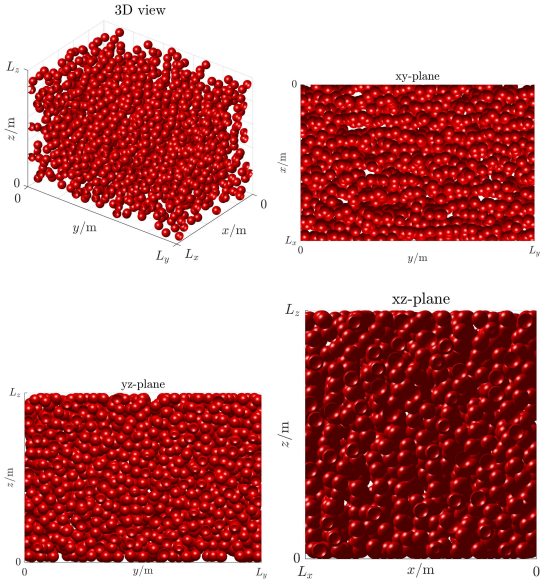


Figure G.4: Steady state structures where $\dot{\gamma} = 5000\,\mathrm{s}^{-1}$

G.1.6. $\dot{\gamma} = 7500\,\mathrm{s}^{-1}$

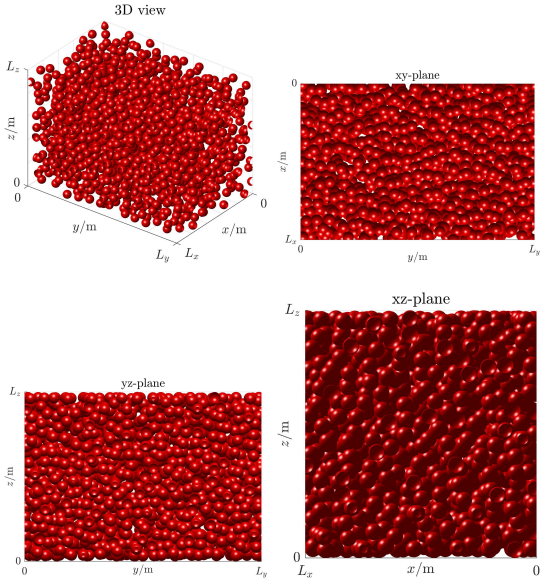


Figure G.5: Steady state structures where $\dot{\gamma} = 7500\,\mathrm{s}^{-1}$

G.1.7. $\dot{\gamma} = 10\,000\,\mathrm{s}^{-1}$

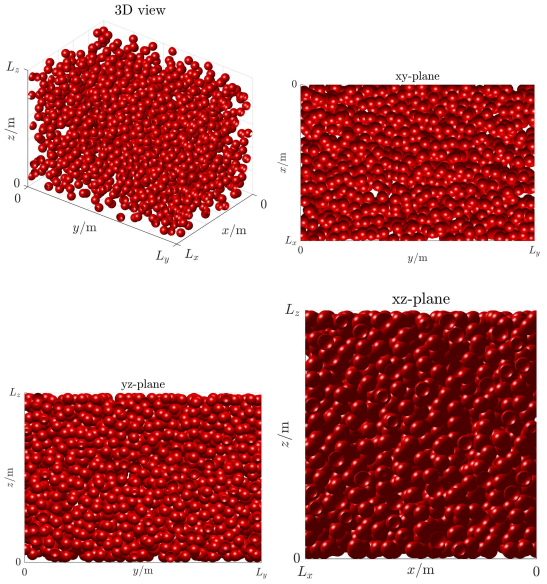


Figure G.6: Steady state structures where $\dot{\gamma} = 10\,000\,\mathrm{s}^{-1}$

G.1.8. Shear stress response

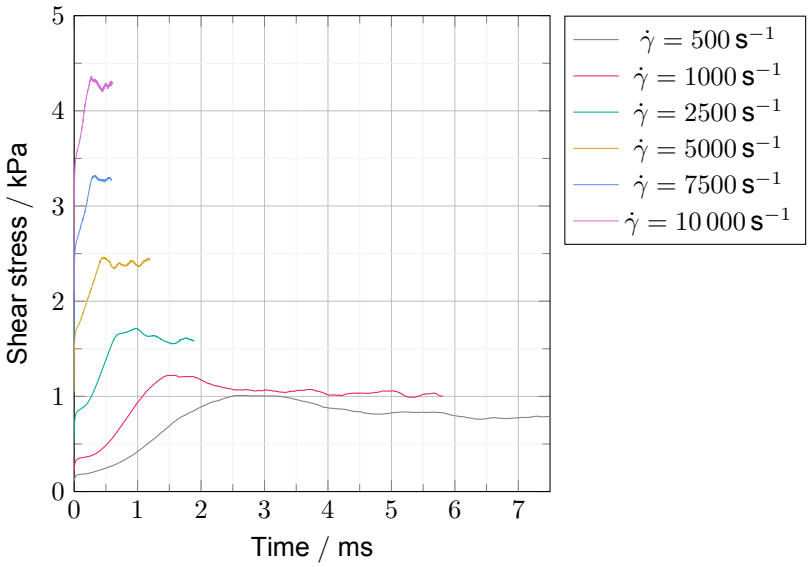


Figure G.7: Shear stress response of the MRF

G.1.9. Rheological model

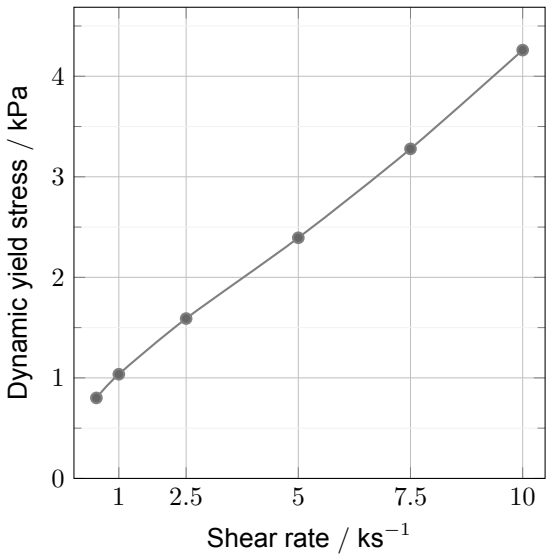


Figure G.8: Rheological model

H

Recommendations

H.1. Integrating the mutual dipole method

The mutual dipole method could be integrated into the parallel Fortran model to get a more accurate estimation of the behavior of the MRF. At the initialization and at each time step the following algorithm could be of use.

Algorithm 3 Integrating the mutual dipole method

for Number of iterations required **do**
Communicate the magnetic moments of the particles between the neighbors
Determine the new magnetic moments of all particles in the current process using the mutual dipole method. The magnetic moments thus depends on the particles in the current process and the particles in all the neighboring processes. It should be noted that only the old magnetic moments should be used.
Wait for all processes to finish
end for

It should be noted that this algorithm is different then the algorithm that is normally used to determine the magnetic moment of the particles, given in equation (2.15). Therefore, the required number of iterations needs to be determined. Integrating this algorithm will increase the run times of the model, because there is more communication between the processes and extra for loops are introduced.

H.2. Determine the texture height

Instead of dividing the domain in slabs, the domain could be divided into cubes. In each cube, the volume fraction of particles could be determined. If the cubes are small enough, it should be possible to estimate the dimensions of the aggregates that are formed within the MRF. Just as with the slabs, any volume of a particle that is present in a cube should be taken into account.