Taylor-Aris Dispersion of Elongated Rods

by
Ajay Harishankar Kumar
Sc.M., Brown University

Thesis submitted in partial fulfillment of the requirements
for the Degree of Master of Science
in the School of Engineering at Brown University

PROVIDENCE, RHODE ISLAND

May 2020
Signature Page

This thesis by Ajay Harishankar Kumar is accepted in its present form by the School of Engineering as satisfying the thesis requirements for the degree of Master of Science.

Date: ________________

Daniel M. Harris, Ph.D., Advisor ________________________
Thomas R. Powers, Ph.D., Advisor ________________________

Approved by the Graduate Council

Date: ________________  ________________________
Andrew G. Campbell, Dean of the Graduate School
Abstract

Particles transported in fluid flows, such as cells, polymers or nanorods, are rarely spherical in nature. In this study, we numerically and theoretically investigate the dispersion of an initial concentration of elongated rods in 2D pressure-driven shear flow. The rods translate due to diffusion and advection, and rotate due to rotational diffusion as well as their classical Jeffery’s orbit in shear flow. When rotational diffusion dominates, we approach the classical Taylor Dispersion result for the longitudinal spreading rate by using an orientationally averaged translational diffusivity for the rods. However, in the high shear limit, the rods tend to align with the flow and ultimately disperse more as a direct consequence of their anisotropic diffusivities. The relative importance of the shear-induced orbit and rotational diffusivity can be represented by a rotational Peclet number, and allows us to bridge these two regimes.
Acknowledgments

First and foremost, I would like to thank my thesis advisors, Prof. Daniel Harris and Prof. Thomas Powers. Their support, knowledge and mentorship have been the primary reason for my success at Brown University. They have always been supportive of all my decisions, guided me throughout this project and patiently answered my questions. I could not have done this project without them. I would especially like to thank Prof. Daniel Harris for his wisdom, advice and help in times when I have needed it the most.

I am deeply grateful to all the faculty that have taught me courses at Brown. Without their help, I would have not been able to develop the necessary skills required to tackle all future problems in this field. I would like to acknowledge the funding received from the NSF that made this research possible.

This research was conducted using the computational and visualization resources and services at the Center for Computation and Visualization (CCV), Brown University. I would like to thank the CCV for all their help and assistance. They truly are amazing and have infinite patience in helping everyone with their code.

A lot of the work has always been discussed with the current and former members of the Harris Lab. I appreciate all the conversations and discussions and advise I have received from our lab members. A special thanks to Luke Alventosa for his insight, advice and general help, Nikolay Ionkin for his support and Ian Ho for great conversations that have expanded my abilities. I would also like to thank all associated with Fluids group at Brown University.

Finally, I would like to thank my family and friends who have always been supportive and have made this journey even more special. Without them, I would not be here.
## List of Figures

1. Taylor Dispersion in 2D Pressure driven shear flow\(^1\) ........................................... 1
2. Plot comparing the Monte Carlo solution to the analytical expression\(^2\) for \(Pe = 1000\). The initial condition is a uniform Gaussian distribution with a \(\sigma = 1\). .......................................... 8
3. Anisotropic Diffusivity of a prolate spheroid. ................................................................. 10
4. Anisotropic Diffusivity of rods compared to spheres and slender body theory.\(^3\) ............ 11
5. Figure depicting the coordinate system. .......................................................................... 12
6. Defining the coordinate axis with \(x\) being the direction of flow ..................................... 14
7. Time series of a single rod. The figure shows how Monte Carlo methods apply incremental changes on each time step making rods change their position and orientation. .............. 16
8. Plots comparing the plot of the variance vs time for spherical and ellipsoidal particles \(p = 1000\) and \(Pe_r = 100\). \(Pe = 1000\) for both cases with an initial condition of a Gaussian distribution of variance 1. ......................................................................................... 17
9. Effective Diffusivity of Ellipsoid \(p = 1000\) normalised by equation 20 as a function of \(Pe_r\) for different values of \(Pe\). ........................................................................................................ 18
10. Effective Diffusivity for different aspect ratios. ............................................................... 19
11. Distribution of particles along half the channel for \(Pe = 1000\), \(Pe_r = 10\) and \(p = 1000\) ...... 22
12. Distribution of diffusivity along the channel and orientational distributions for correspond- ing shear layers. Here, \(p = 1000\) and \(Pe_r = 100\) ......................................................... 25
13. Theoretical maximum possible effective diffusivity coefficient .................................. 26
14. Overlaying the theoretical prediction on to figure 9 ......................................................... 27
1 Motivation and Introduction to Taylor Dispersion

A uniform patch of solute in laminar flow usually typically spreads as a result of fluid advection and molecular diffusion. At short times, the solute mimics the profile of the flow and can induce concentration gradients that are later minimised by molecular diffusion, which enhances the spread of the solute. The concentration profile can be characterised by a flux equation with an effective diffusion constant. For a solute in a fluid flowing with a particular velocity, the governing equation is the advection-diffusion equation. This phenomena was first studied by G.I. Taylor.\(^4\) R. Aris has expanded on this work using his method of moments and that is why this phenomena is commonly referred to as Taylor-Aris Dispersion.\(^5\) Most of the work in this area has focused on isotropic particles whose molecular diffusion is independent of the orientation.

H. Brenner\(^6\) further expanded the theory to a Generalised Taylor Dispersion theory. This method is a framework laid out to solve dispersion problems for any shear velocity profile. The Generalised Taylor Dispersion method has been used to understand the dispersion of active matter in a shear velocity profile.\(^7\)–\(^9\) Another common method to understand dispersion problems is to use a multiscale perturbation method or homogenisation theory.\(^10\)–\(^11\) A key aspect for drug delivery applications is the channel geometry which affects the velocity profile and the subsequent dispersion.\(^12\)–\(^13\) Understanding these problems has led to the development of transport of particles, especially for drug delivery systems using microfluidics.\(^14\) Major advancements have been made in the study of active swimmers, bacteria and self-propelling systems. Understanding the dispersion and the spread of these active systems will help with a lot of applications particularly with active Brownian particles, micro-swimmers and movement of bacteria in bioreactors.\(^15\)–\(^19\)
Since most drugs, cells and species being transported are not spherical in nature, it is important to understand the effect of the shape of the particle on dispersion. The initial inspiration for this project comes from a drug that was discovered at the School of Medicine at Brown University to fix cartilage damage in the knees. The drug is delivered in the form of a nanorod which is transported in a fluid to the damage site. In this study, our particles are non-interacting, passive Brownian tracers, with any steric and wall-based effects ignored.

The goal of this thesis is to numerically and theoretically explore the dispersion of ellipsoidal particles. This study reveals that there is an enhanced spreading of rods as compared to spherical particles which is rationalized physically. Furthermore, this thesis provides a framework to solve different problems in the transport of rod-shaped particles, with all relevant codes available in the appendix section.

1.1 Advection Diffusion Equation

The advection-diffusion equation for a concentration profile describes its spacial distribution and temporal evolution. For solute particles in solution, the equation is

$$\frac{\partial C}{\partial t} = \nabla \cdot J_r$$  \hspace{1cm} (1)

where $C$ is the concentration field, $J_r$ is the flux in $r$ the positional space. Equation 1 can be written as

$$\frac{\partial C}{\partial t} = \nabla \cdot (D \nabla C) - \nabla \cdot (uC) + r'_s$$ \hspace{1cm} (2)

where, $D$ is the isotropic diffusivity or molecular diffusion constant of an isotropic spherical solute particle, $u$ is the velocity of the solvent fluid which can be obtained using the Navier-Stokes equations and $r'_s$ is a source or a sink term. Since the particles are passive Brownian tracers, the source or the sink term is zero. This equation describes the time evolution of the distribution of solute particles in space. Furthermore, the velocity profile is strictly parallel and along the length of the channel. For isotropic diffusivity and a steady simple pressure-driven shear flow in 2-D, the equation reduces to the following:

$$\frac{\partial C}{\partial t} = D\frac{\partial^2 C}{\partial x^2} - u(y)\frac{\partial C}{\partial x}.$$ \hspace{1cm} (3)

Another interpretation of this transport equation is the transport of solute particles due to molecular diffusion and advection or bulk motion. The first term on the right-hand side is the diffusive flux and the second term is the advective flux. The boundary conditions for the problem are the no flux boundary conditions on the
wall with an initial patch of solute uniformly distributed along the width of the channel and either a Gaussian distribution or a Dirac Delta function along the length of the channel. These boundary conditions can be represented as
\[ \frac{\partial C}{\partial y} \bigg|_{y=\pm a} = 0 \] (4)
and,
\[ C(x, y, t = 0) = C_0(x) = \frac{1}{2a} \delta(x). \] (5)

### 1.2 Non-Dimensionalisation and Transitional Peclet Number

The solvent is an incompressible fluid and the flow is a steady pressure-driven shear flow. The flow is a Poiseuille flow and has a parabolic profile with a characteristic maximum velocity \( U \). The characteristic length scale is the half-width of the channel \( a \) and the characteristic time scale is the diffusive time scale \( t_d \).

The diffusive time can be defined as
\[ t_d = \frac{a^2}{D}. \] (6)

This timescale represents the characteristic time for a solute particle to be transported a particular distance \( a \) purely by molecular diffusion. In our case it can be understood as the characteristic time for a particle to travel from the centre of the channel to the sides of the channel purely because of diffusion. Using the scales
\[ \tilde{x} = \frac{x}{a}, \quad \tilde{y} = \frac{y}{a}, \quad \tilde{u} = \frac{u}{U}, \quad \tilde{t} = \frac{t}{t_d}, \quad \tilde{t} = \frac{t}{D/a^2} \]
we can non dimensionalise equation (3) to obtain the following equation (drop ~)
\[ \frac{\partial C}{\partial t} = -Pe \ u(y) \ \frac{\partial C}{\partial x} + \frac{\partial^2 C}{\partial x^2} + \frac{\partial^2 C}{\partial y^2}. \] (7)

This problem only has one dimensionless number called the Peclet number, defined as the ratio of advective transport to diffusive transport. Another interpretation of the Peclet number is the ratio of the diffusive time scale to the advective time scale and is represented as
\[ Pe = \frac{U a}{D}. \] (8)

The boundary conditions and the initial conditions are,
\[ \frac{\partial C}{\partial y} \bigg|_{y=\pm 1} = 0 \] (9)
and,

\[ C(x, y, t = 0) = C_0(x) = \frac{1}{2} \delta(x). \]  

(10)

1.3 Taylor’s scaling arguments

We will review Taylor’s results before moving to our results. Taylor studied the spreading of a pulse of a solute in a parabolic shear flow in a circular pipe. At \( t = 0 \), a pulse of solute is injected in the system. The evolution of concentration of this profile is given by equation 2. For the case of a simple 2-D parallel plate geometry, the equation reduces to equation 3 and in non-dimensional form equation 7 where,

\[ u(y) = 1 - y^2. \]  

(11)

The first assumption Taylor made was to neglect axial diffusion. This is because the advection in the radial direction is much larger than the diffusion when the \( Pe \gg 1 \). The correction was later made by Aris in his method of moments where the radial diffusion terms are also considered. Taylor sought a solution for the effective diffusivity for long times only, wherein a pulse has travelled a certain length \( L \) such that its initial profile does not matter. In such a case, the advective time scale along this length is much larger than the diffusive time scale along the width of the channel and diffusion has balanced all gradients in the radial direction. The solution Taylor wished to find was for the condition,

\[ \frac{L}{a} \gg Pe. \]  

(12)

When the value \( a/L = \varepsilon \) is taken to be a small parameter, separated time scales can be used to solve the Taylor Dispersion problem using homogenisation theory. A major assumption made by Taylor is that the average speed of the flow is the same as the average speed of the particles. For isotropic particles, whose diffusivities do not depend on the orientation or their position in the channel, this can be true. If there is an advective flux which causes the particles to migrate, it is important to consider the distribution along the channel, as the mean speed of the particles will be different from the mean speed of the flow. To understand the development of the concentration profile downstream, the coordinate system is shifted to the mean speed of the flow, resulting in

\[ x_1 = x - \frac{\bar{U}}{U} Pe \, t. \]  

(13)
In the case of a 2-D parallel plate geometry,

\[ \bar{U} = \frac{2}{3} U. \]  

(14)

In the mean frame of reference and by neglecting radial diffusion or diffusion in \( x \) direction, equation 7 reduces to,

\[ \frac{\partial C}{\partial t} = -Pe \left( \frac{1}{3} - y^2 \right) \frac{\partial C}{\partial x_1} + \frac{\partial^2 C}{\partial y^2}. \]  

(15)

For long times, the initial condition is forgotten and the time derivative is zero along \( x_1 \). This condition is only true for long times when equation 12 is true. The above equation then simplifies to,

\[ Pe \left( \frac{1}{3} - y^2 \right) \frac{\partial C}{\partial x_1} = \frac{\partial^2 C}{\partial y^2}. \]  

(16)

The boundary condition for this problem is equation 9. The linear PDE has a particular solution and a homogeneous solution. The homogeneous solution is independent of \( \frac{\partial C}{\partial x_1} \) and the particular solution is directly proportional to \( \frac{\partial C}{\partial x_1} \). The solution for the concentration profile is,

\[ C = Pe \frac{1}{3} \frac{\partial C}{\partial x_1} \left( \frac{y^2}{2} - \frac{y^4}{4} \right) + C_h. \]  

(17)

The first term is the particular solution and the second term is the homogeneous solution. Hence the rate of mass transfer or the flux across the section \( x_1 \) is,

\[ Q = - \int_{-1}^{1} Pe \left( \frac{1}{3} - y^2 \right) C dy. \]  

(18)

Integrating this equation results in

\[ Q = \frac{16}{945} Pe^2 \frac{\partial C}{\partial x_1} = \kappa \frac{\partial C}{\partial x_1}. \]  

(19)

where, \( \kappa \) is the effective diffusivity or dispersion constant. Therefore for a spherical particle in 2-D pressure driven flow, the effective diffusivity is,

\[ \kappa = \frac{16}{945} Pe^2. \]  

(20)

The above result indicates that, while moving at the mean speed of the particles, the problem reduces to a 1-D diffusion problem in \( x \) with an effective diffusivity \( \kappa \). In the frame of the mean speed of the particles, the concentration profile in the long-time limit converges to a Gaussian, widening with a diffusion constant \( \kappa \). For statistical purposes, the effective diffusivity is the slope of the linear segment of the second moment
or the mean square displacement of particles or the variance with respect to time. Equation 20 shows an enhanced dispersion along the axis due to the presence of variations in the velocity. It is interesting to note that $\kappa$ is inversely proportionate to the molecular diffusivity along the width of the channel. If the molecular diffusion constant along the width of the channel changes, different results would be obtained.

### 1.4 Monte Carlo Method

Problems relating to diffusion can be interpreted as a consequence of Brownian Motion. Establishing the relation between diffusion and Brownian motion was first done by Einstein. Since Brownian Motion is a stochastic process, a Monte Carlo method has quite often been used to solve diffusion problems. This method is commonly used for complex geometries or complex particle shapes. All these are based on the basic principles of a random walker. The advection-diffusion equations can also be interpreted as a stochastic PDE. The stochastic nature is a consequence of Brownian Motion. The stochastic PDE in non dimensional form is

$$dx = Pe \mu dt + \sqrt{2}dW_x.$$  \hspace{1cm} (21)

Here $dt$ is the time stepping (non dimensional), $dW_i$ is the white noise in the 'ith' direction, $\mu$ is the velocity vector and $Pe$ is the Peclet number as per equation 8. The above equation is the governing equation for each particle. Applying this equation is applied to each particle at each time step defines the Monte Carlo Brownian Dynamics simulation. In other words, for each particle the following is done calculated:

$$dx = Pe \mu(y) dt + \sqrt{2}dW_x.$$  \hspace{1cm} (22)

The change in the position of the particle is due to the first term which corresponds to the particle being advected downstream and the second term which is the diffusion as a consequence of Brownian Motion. In the $y$ direction,

$$dy = \sqrt{2}dW_y.$$  \hspace{1cm} (23)

There is no advection and the problem is a simple diffusion problem or a Random Walk. The boundary conditions are implemented through a simple billiard like reflection rule similar to work done by M. Aminian. The code has been created on MATLAB and Python. The particles ($N = 10^6$) are uniformly distributed along the width of the channel and are a Gaussian in $x$ with a given mean and variance. The white noise increments are independently sampled from a Gaussian distribution of a given mean and variance. In the case of the non dimensional problem, the mean is 0, and the variance is normalised. Most high level languages
like MATLAB and Python already have inbuilt random number generators which sample from a normal distribution, namely \texttt{normrand(\mu,\sigma)} on MATLAB or numpy.random.normal(\mu,\sigma) on Python. Therefore,

\[
dW_i = \sqrt{2 \Delta t} \text{norm}(0,1)
\]  

(24)

where, \texttt{norm}(0,1) is the random number generated at each time step for each particle in each direction. A uniform homogeneous Euler time stepping has been used. This ensures that the magnitude of the white noise is much less than the width of the channel so there is only one wall collision at most. Despite being a slow method, (convergence of the order of \(1/\sqrt{N}\) where \(N\) is the number of particles being simulated), the gridless and the Stochastic Differential Equation (SDE) approach makes it easy to combine and capture all statistics. Computing mean and variance are easy on high level languages. For example, in MATLAB, \texttt{mean(x)} gives the mean position of all particles and \texttt{var(x)} the variance. These values are stored in a vector for each time step. By combining statistics, computing the mean position and variance of all particles at each time step, the plot of the variance with time can give us the effective diffusivity. The effective diffusivity is the slope of the variance vs time, at long times when the plot is linear. The code developed has been attached in the appendix section A.1.
Figure 2: Plot comparing the Monte Carlo solution to the analytical expression\textsuperscript{2} for $Pe = 1000$. The initial condition is a uniform Gaussian distribution with a $\sigma = 1$.

2 Anisotropic Diffusivity

Diffusion for most cases is assumed to be a scalar quantity. In reality, diffusion depends on the geometry and orientation of the object and thus is a tensor. This can be understood using the Stokes-Einstein equation,\textsuperscript{28,33}

$$D_{ij} = \frac{k_b T}{f_{ij}}$$ (25)

where, $k_b$ is the Boltzmann constant, $T$ is the temperature of the fluid and $f$ is the friction that the body experience inside a fluid. For the case of a spherical particle in Stokes flow, it is well established that the friction factor is\textsuperscript{23}

$$f = 6\pi \mu r$$ (26)
here, \( r \) is the radius of the particle and \( \mu \) is the viscosity of the fluid the solute particle is in. This expression is only valid when \( Re \ll 1 \). This leads to the expression to calculate the diffusivity of any spherical particle,

\[
D = \frac{k_b T}{6\pi \mu r}.
\]  

(27)

Calculating the diffusion constant only requires the radius of the particle or the radius of gyration of a molecule, as the rest of the quantities are easy to measure. Unfortunately, measuring the radius is actually much harder and requires expensive tools like a scanning electron microscope (SEM). Another way is to calculate the diffusion constant from the dispersion coefficient. The diffusion constant can then be used to estimate the radius of the particle.\(^{34,35} \) Another important characteristic of irregular shaped objects is the rotational motion of the particles. Due to to irregular shapes, particles are constantly spinning due to rotational Brownian motion. Analogous to transitional Brownian motion, rotational Brownian motion is also a random walk and describes how irregular objects rotate due to rotational diffusion.\(^{28,33} \) Usually,

\[
D_r \sim \frac{k_b T}{\mu r^3}.
\]  

(28)

For a spherical particle, the rotational diffusivity,\(^{33} \) is

\[
D_r = \frac{k_b T}{8\pi \mu r^3}.
\]  

(29)

Even for irregular shaped objects, the rotational diffusivity can be represented as a tensor.\(^{36-40} \) It also coupled with the transitional diffusion tensor.

### 2.1 Diffusion Tensor for Ellipsoidal Particle

For the case of an ellipsoidal (prolate) particle, there is some symmetry which allows the decoupling of the rotation and transitional diffusion tensor.\(^{38} \) This reduces the complexity of the problem. The transitional diffusion tensor is symmetric has two unique components. The rotational diffusion tensor is isotropic and has only one component. In the case of a quasi 2-D problem (restricted to one degree of rotational freedom), the two components are shown in the figure 3. These expressions are well established\(^{26} \) and can be obtained from the textbook "Low Reynolds Number Hydrodynamics" by Happel and Brenner.\(^{33} \)

For a prolate spheroid particle with a semi major axis \( a_r \) and semi minor axis \( b_r \), with an aspect ratio defined as,

\[
p = \frac{a_r}{b_r}.
\]  

(30)
where $a_r > b_r$, the diffusion coefficients are,

$$D_r = \frac{3k_b T p \left( \frac{(2p^2-1) \log \left( \frac{p+\sqrt{p^2-1}}{\sqrt{p^2-1}} \right)}{p^2-1} - p \right)}{16\pi \mu a_r b_r^2 (p^4 - 1)}$$  \quad (31)$$

$$D_\parallel = \frac{k_b T \left( -\frac{2p}{p^2-1} + \frac{2p^2-1}{(p^2-1)\sqrt{p^2-1}} \log \left( \frac{p+\sqrt{p^2-1}}{p-\sqrt{p^2-1}} \right) \right)}{16\pi \mu b_r}$$  \quad (32)$$

$$D_\perp = \frac{k_b T \left( \frac{p}{p^2-1} + \frac{2p^2-3}{(p^2-1)\sqrt{p^2-1}} \log \left( \frac{p+\sqrt{p^2-1}}{p-\sqrt{p^2-1}} \right) \right)}{16\pi \mu b_r}$$  \quad (33)$$

From these expressions, an important quantity is defined:

$$\alpha(p) = \frac{D_\perp}{D_\parallel}. \quad (34)$$

For the case of a spherical particle, this value is 1 and asymptotically tends to $1/2$ as $p$ tends to infinity. This expression agrees with slender body theory.\(^3\)

For the quasi-2D problem, the diffusion tensor for an ellipsoid is,

$$D = \begin{bmatrix} D_\parallel & 0 \\ 0 & D_\perp \end{bmatrix} \quad (35)$$

where, the orientationally averaged diffusivity can be obtained from the trace of the diffusion tensor which we define as the characteristic diffusion constant,

$$\frac{tr(D)}{2} = \frac{(D_\parallel + D_\perp)}{2}. \quad (36)$$
Therefore,

$$\bar{D} = \frac{D_{\parallel} + D_{\perp}}{2}. \quad (37)$$

2.2 Jeffery’s Orbit

In 1922, G.B. Jeffery calculated the rotation rate for a prolate spheroid in a shear flow:

$$\omega_{r}(\dot{\gamma}, \theta, p) = \dot{\gamma} \frac{p^2 \sin^2 \theta + \cos^2 \theta}{p^2 + 1} \quad (38)$$

Here, $p$ is the aspect ratio as defined by equation 30 and $\dot{\gamma}$ is the shear rate defined as,

$$\dot{\gamma} = \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}. \quad (39)$$
Equation 38 is called the Jeffery’s Orbit. The solution shows that the particles spend more time hydrodynamically aligned with the flow. For the case of a simple 2-D pressure driven flow with velocity profile is given by equation 11 and the shear rate is linear and is given by,

$$\dot{\gamma}(y) = -2U \frac{y}{a^2}. \quad (40)$$

### 2.3 Rotational Advection Diffusion for constant Shear Flow: Rotational Peclet Number

Analogous to the advection diffusion equation for the evolution of the concentration profile over time in translational space, there is a rotational advection diffusion equation for the time evolution of the distribution in the orientational space. Similarly writing a flux equation in orientational space $q$ like equation 1,

$$\frac{\partial C}{\partial t} = \nabla_q \cdot J_q. \quad (41)$$

The flux in orientation space is because of the rotational advection and rotational diffusion, just like the transitional space. For the case of rotation constricted about one axis only (quasi static 2-D confinement of an prolate spheroid in a shear flow) the rotational advection diffusion equation from the flux is,

$$\frac{\partial C}{\partial t} = -\frac{\partial \omega C}{\partial \theta} + D_r \frac{\partial^2 C}{\partial \theta^2}. \quad (42)$$

The characteristic time scale is the time the the rod takes to complete one rotation purely because of rotational diffusion:

$$t_r \sim \frac{1}{D_r} \quad (43)$$

Non dimensionalising using the characteristic time scale,

$$D_r \frac{\partial C}{\partial t} = -\dot{\gamma} \frac{\partial \omega C}{\partial \theta} + D_r \frac{\partial^2 C}{\partial \theta^2} \quad (44)$$
and dividing by $D_r$ results in
\[
\frac{\partial C}{\partial t} = -\frac{\dot{\gamma}}{D_r} \frac{\partial \omega C}{\partial \theta} + \frac{\partial^2 C}{\partial \theta^2}.
\] (45)

A new dimensionless quantity is defined as the Rotational Peclet number,
\[
Pe_r = \frac{\dot{\gamma}}{D_r}.
\] (46)

For the case of a Poiseuille flow, the Rotational Peclet number has been defined on the basis of the average shear rate of the channel. The Rotational Peclet number is defined as the ratio of the shear to the rotational diffusivity or the time it takes for a particle to rotate because of rotational diffusion to the time it takes for the same angle to be rotated by advection or local shear. The non dimensionalised form is,
\[
\frac{\partial C}{\partial t} = -Pe_r \frac{\partial \omega C}{\partial \theta} + \frac{\partial^2 C}{\partial \theta^2}.
\] (47)

For the case of a simple shear flow, past work has been done to understand the effect of rotational Brownian motion on advection$^{42-44}$ The interplay between rotational advection and rotational diffusion has been studied to show that there is a migration unlike spherical particles$^{18,45-47}$
3 Monte Carlo Method for Ellipsoidal Particles

To the author’s knowledge, no Monte Carlo simulations have been done for a spheroids in a parabolic flow to understand their dispersion. The same way the problem for spherical particles was solved as a Stochastic PDE (SPDE), the SPDE for ellipsoidal particles is,

\[ dx = ud\tau + \left( \sqrt{2dtD_r} \cdot norm(0,1) \right) R(\theta). \] (48)

As the translational and orientational diffusion tensors are not coupled,\(^{38}\) the change in the orientation for the quasi 2-D case is given by,

\[ d\theta = \omega(\dot{\gamma}, p, \theta) dt + \sqrt{2dtD_r} norm(0,1) \] (49)

Here, \( u \) is the velocity profile or equation 11, \( D \) is the diffusion tensor as defined in equation 35. \( norm \) is the white noise value sampled from a normal distribution. It is a vector in the transitional space because the first value is for the parallel direction and the second is for the perpendicular direction. The obtained vector is then rotated using a regular 2-D rotation matrix \( R(\theta) \) to be in the standard rectangular coordinate system.

The change in orientation is given by the Jeffery’s orbit \( \omega \) defined in equation 38 with shear as equation 40. Finally we have the rotational Brownian Motion term.

3.1 Non Dimensional Forms of Equations

![Figure 6: Defining the coordinate axis with x being the direction of flow](image)

The SPDE’s in the previous sections are non dimensionalised. Using the diffusive time as the characteristic time scale as per equation 6, the characteristic length scale for both \( x \) and \( y \) is the half width of the channel \( a \) and the characteristic velocity \( U \) is the maximum velocity of the channel. The Jeffery’s orbit term is non dimensionalised by the average shear in the channel. The diffusion tensor is non dimensionalised by the orientationally averaged diffusivity defined as per equation 37.
Using these conditions, the equations reduce to (drop ~),

\[
dx = Peu(y)dt + \left( \sqrt{2dtD} \cdot \text{norm}(0, 1) \right) R(\theta)
\]  \hspace{1cm} (50)

and,

\[
d\theta = Pe\omega(y, p, \theta)dt + \sqrt{2dt \frac{Pe}{Pe_r}} \cdot \text{norm}(0, 1).
\]  \hspace{1cm} (51)

There are three dimensionless groups. Firstly, The Peclet number \( Pe \) which has been defined as per equation 8. Secondly, the Rotational Peclet number \( Pe_r \), defined as equation 46, and lastly the particle aspect ratio \( p \) as defined in equation 30. The Jeffery’s orbit, shear rate and the velocity for a 2D channel are well known.

The boundary conditions are implemented through a simple billiard like reflection rule similar to the work done by M. Aminian.13 Also, since the particles are ellipsoidal in nature, there is a \( 2\pi \) symmetry in the system. So if the value of the angle is more than \( 2\pi \), a mod of \( 2\pi \) of the angle is taken. The particles \( N = 10^6 \) are uniformly distributed along the width of the channel and are a Gaussian in \( x \) centered at \( x = 0 \) with a specified variance. The particles are initialized uniformly distributed in their orientational space as well.

A uniform homogeneous Euler time stepping has been used. This ensures that the magnitude of the white noise is much lesser than the width of the channel so there is only one wall collision at most. Also, the time step has to be small enough such that the Jeffery’s Orbit is well resolved. Despite being a slow method, (convergence of the order of \( 1/\sqrt{N} \) where \( N \) is the number of particles being simulated), the gridless and the SDE approach makes it easy to combine and capture all statistics.

3.2 Parallelize Code and Combining Statistics

To carry out a simulation of such a large number of particles, it is necessary to parallelise the code over many CPUs. This reduces computational times significantly. Since the particles are assumed to be non interacting and passive, the Monte Carlo method is easy to distribute the particles over many CPUs. Instead of one CPU carrying out all \( 10^6 \) particles and saving their statistics, the particles were split into many different CPU cores and the statistics were combined later. MATLAB uses a simple command called "parfor" instead of "for" which creates multiple instances of a function on different cores of the CPU. To maximize the number of CPU cores, the CCV facilities at Brown University were used. The facility enabled access to over 200 CPU cores making the computation much faster.
All the important data for each CPU instance was stored, like mean, variance, time vector and the position and orientation of the last time step. This made combining statistics of different data sets easy. For each set containing \( n \) particles, there are \( N/n_0 = r \) sets or instances with the mean of the \( i \)th instance defined as \( \mu_i \) and variance as \( \sigma_i^2 \) for each time step:

\[
\bar{\mu}(t) = \frac{1}{r} \sum_{i=1}^{r} \mu_i(t) 
\]

\[
\bar{\sigma}^2(t) = \frac{1}{r} \sum_{i=1}^{r} \left( \sigma_i^2(t) + (\mu_i(t) - \bar{\mu}(t))^2 \right) 
\]

The effective diffusivity of the particles is mathematically defined as,

\[
\frac{d\bar{\sigma}^2}{dt} \bigg|_{t \to \infty} = \kappa 
\]

For the case of case of spherical particles in a 2-D channel flow equation 20 is the effective diffusivity equation. Approximately, after \( 0.25t_d \), the slope of the plot of the variance vs time is linear. To calculate the effective diffusivity from the variance. We fit a curve to the variance vs time data. The curve fitted is of the form,

\[
\bar{\sigma}^2(t) = a_0 + a_1 t + a_2 \exp(-a_3 t) 
\]

where \( a_0 \) is the offset due to the initial conditions, \( a_2 \) and \( a_3 \) are fitting constants for the growth phase of the variance. \( a_1 \) is the slope of the linear phase. Substituting the fitting expression into equation 54,

\[
\kappa = a_1 
\]

The Monte Carlo code has been attached in appendix A.2.

Figure 7: Time series of a single rod. The figure shows how Monte Carlo methods apply incremental changes on each time step making rods change their position and orientation.
3.3 Results and Discussion

The code was implemented for a range of $p$, $Pe$, and $Pe_r$. The effective diffusivities $\tilde{\kappa}$ have been normalised by the effective diffusivity of the spherical particle $\kappa$ given by equation 20. It can be seen that for a fixed $Pe$ and $p$, as $Pe_r$ increases the particles have a greater dispersion. When the $Pe_r$ is high, the shear rate is also very high. Due to the Jeffery’s Orbit, the particles are aligned with the flow for more time. This means it is harder for the particles to diffuse through the width of the channel as the perpendicular side has a lower diffusivity. Since the effective diffusivity is inversely proportionate to the molecular diffusion along the width of the channel, the rods spread more as a consequence of the Jeffery’s orbit. Essentially, the rods spend a longer time aligned with the flow. Conversely, for low $Pe_r$, the rods are constantly spinning due to more pronounced rotational diffusion. They behave like spherical particles as each side spends equal time aligned with the flow.
As we lower the aspect ratio, the particles behave more like spherical particles and their effective diffusivity decreases.
4 Shear Induced Lateral Migration of Brownian Rods

Analysing the simulation results showed that there is a migration flux that drives particles towards the walls. This concept has been well studied.\textsuperscript{47–49} In 1996, Nitsche and Hinch\textsuperscript{46} studied the shear induced lateral migration of Brownian rods. In their system, they had elongated Brownian rods suspended homogeneously in a fluid in a parabolic velocity profile. Over time, they saw a migration of these rods towards the wall due to the difference in orientational distributions at each shear layer, giving rise to a flux. The difference in orientational distribution along the width of the channel gives rise to a migration velocity.
4.1 Equations

The equations can be explained using a flux law similar to equation 1 and equation 41. There is a conservation of flux in the transitional and orientational space:

\[
\frac{\partial P}{\partial t} = \nabla_r \cdot J_r + \nabla_q \cdot J_q.
\] (57)

The fluxes have a rotated diffusion tensor. The diffusion tensor defined in equation 35 is rotated such that it is a function of the orientational space:

\[
D = D(q).
\] (58)

Here, \( C(r,q,t) \). This equation describes the evolution of the concentration profile in the orientational space and translational space. The equation is a Fokker-Planck equation.\(^{50}\) For the case of a 2-D pressure driven, shear flow, Nitsche and Hinch had a homogeneous distribution in \( x \). So,

\[
\int_{-\infty}^{\infty} P(x,y,\theta,t)dx = C(y,\theta,t).
\] (59)

The simplified equation is for the density function \( C \)

\[
\frac{\partial C}{\partial t} = D_r \frac{\partial^2 C}{\partial \theta^2} - \frac{\partial}{\partial \theta} (\omega_r(\gamma(y),\theta,p)C) + \frac{\partial}{\partial y} \left( D_{yy}(\theta) \frac{\partial C}{\partial y} \right)
\] (60)

The first term on the right hand side is the rotational diffusion. The second is the rotational advection with the Jeffery’s orbit and the last is the translational diffusion along the channel. Non dimensionalising the equation 60 similar to the Monte Carlo equations,

\[
\tilde{y} = \frac{y}{a}, \quad \tilde{t} = \frac{t}{t_d}, \quad \tilde{\omega} = \frac{\omega}{U_0}, \quad \tilde{D}_{i,j} = \frac{D_{i,j}}{\bar{D}}
\]

\[
\frac{Pe_r}{Pe} \frac{\partial C}{\partial \tilde{t}} = \frac{\partial^2 C}{\partial \tilde{\theta}^2} - Pe_r \frac{\partial \omega C}{\partial \tilde{\theta}} + \frac{Pe_r}{Pe} \frac{\partial}{\partial \tilde{y}} \left( D_{yy}(\theta) \frac{\partial C}{\partial \tilde{y}} \right)
\] (61)

Here there are three non dimensional groups like the Monte Carlo problem. They are the Peclet number defined as per equation 8 with the diffusion constant defined as the orientationally averaged diffusion constant like equation 37, the Rotational Peclet number defined as per equation 46 and the particle aspect ratio defined as per 30. For consistency, the ratio of the translational Peclet number and Rotational Peclet number can be combined as,

\[
\varepsilon_r = \frac{Pe_r}{Pe} = \frac{\bar{D}}{a^2 D_r} = \frac{t_r}{t_d}
\] (62)
where \( \varepsilon_r \) can also be understood as the ratio of rotational time scale to the diffusive time scale. This quantity can be interpreted as the ratio of the particle size to the channel width and that is why it is always a small quantity. The Fokker-Planck equation for this system can also be written as,

\[
\varepsilon_r \frac{\partial C}{\partial t} = \frac{\partial^2 C}{\partial \theta^2} - Pe_r \frac{\partial \omega C}{\partial \theta} + \varepsilon_r \frac{\partial}{\partial y} \left( D_{yy}(\theta) \frac{\partial C}{\partial y} \right).
\]

The boundary conditions for this problem are the no flux boundary condition along the wall, periodicity in orientational space \( C(y, 0, t) = C(y, 2\pi, t) \) and lastly the integral condition,

\[
\int_{-1}^{1} \int_{0}^{2\pi} C(y, \theta) d\theta dy = 1.
\]

### 4.2 Direct Numerical Solution and Results

To obtain an equilibrium solution, equation 63 was solved using a finite difference solver with the appropriate boundary conditions. A central difference in \( y \) and \( \theta \) with a Forward Euler time stepping was used. The problem was solved until the change in the L-2 norm error from the previous time step to the next time step was much lesser than a user specific value. The value we used was \( 10^{-8} \). Exploiting the symmetry of an anisotropic particle, the domain in orientational space was reduced from 0 to \( 2\pi \) to 0 to \( \pi \). An initial condition satisfying the integral condition was chosen: \( C(y, \theta, 0) = 1/(2\pi) \). At each time step, a Local Truncation Error (LTE) exists and the normalisation boundary condition was checked and rectified to minimize the LTE. Integration over the orientational space was done to obtain the distribution along the channel.

\[
C_y(y) = 2 \int_{0}^{\pi} C(y, d\theta)
\]

For different value of \( Pe_r \) and a fixed value of \( Pe = 1000 \), the solutions were compared with the Monte Carlo method as shown in figure 11.

### 4.3 Comparing Monte Carlo Results to the PDE solution

The code has been posted in the appendix A.3. Comparing the Monte Carlo and PDE solver shows that the migrations for both the problems are similar. This clearly shows that the particles migrate towards the wall. The Monte Carlo for infinitely many particles should have the same smooth shape as the numerical PDE solver.
5 Modified Taylor Dispersion

5.1 Flux Equation from Diffusion Tensor

The diffusion tensor for an ellipsoidal particle in a shear flow was first calculated by Brenner, 38

\[ D = \varepsilon \varepsilon D + (I - \varepsilon \varepsilon) D_\perp \]  \hspace{1cm} (66)

\( \varepsilon \) is the unit vector along the axis of symmetry and \( \varepsilon \varepsilon \) is the dyadic product and \( I \) is the identity matrix. For this case,

\[ \varepsilon = [\cos \theta \sin \theta]. \]  \hspace{1cm} (67)

Therefore,

\[ \varepsilon \varepsilon = \begin{bmatrix} \cos^2 \theta & \sin \theta \cos \theta \\ \sin \theta \cos \theta & \sin^2 \theta \end{bmatrix}. \]  \hspace{1cm} (68)
Thus the diffusion tensor is,
\[
\mathbf{D} = \begin{bmatrix}
D_\parallel \cos^2 \theta + D_\perp \sin^2 \theta & (D_\parallel - D_\perp) \sin \theta \cos \theta \\
(D_\parallel - D_\perp) \sin \theta \cos \theta & D_\parallel \sin^2 \theta + D_\perp \cos^2 \theta
\end{bmatrix}
\] (69)

and the trace of the diffusion tensor which helps define an orientationally average diffusivity is,
\[
\frac{tr(D)}{2} = \bar{D}
\] (70)

Same as equation 36. From the flux equation,
\[
\frac{\partial C}{\partial t} = \nabla \mathbf{r} \cdot \mathbf{J}_r + \nabla \mathbf{q} \cdot \mathbf{J}_q
\] (71)

Where,
\[
\mathbf{J}_r = \mathbf{u} C - \mathbf{D} \nabla \mathbf{r} C
\] (72)
\[
\mathbf{J}_q = \omega C - \mathbf{D}_r \nabla \mathbf{q} C
\] (73)

Combining the above equations, the flux equation is a Fokker-Planck equation. For a distribution \(C(x, y, \theta, t)\)
\[
\frac{\partial C}{\partial t} = -u(y) \frac{\partial C}{\partial x} + 2D_{xx}(\theta) \frac{\partial^2 C}{\partial x^2} + 2D_{xy}(\theta) \frac{\partial^2 C}{\partial x \partial y} + 2D_{yy}(\theta) \frac{\partial^2 C}{\partial y^2} + D_r \frac{\partial^2 C}{\partial \theta^2} - \frac{\partial}{\partial \theta} (\omega(y, \theta, p)C).
\] (74)

### 5.2 Non Dimensionalisation of Fokker-Planck equation

Similar to the Monte Carlo method, non dimensionalising the problem gives the same non dimensional groups, \(Pe_r, Pe\) and \(p\). The non dimensionalising is done as follows:
\[
\bar{x} = \frac{x}{L}, \quad \bar{y} = \frac{y}{a}, \quad \bar{u} = \frac{u}{U}
\]
\[
\bar{t} = \frac{L}{t_d} = \frac{D}{\bar{u}}, \quad \bar{\omega} = \frac{\omega a}{u_0}, \quad \bar{D}_{i,j} = \frac{D_{i,j}}{\bar{D}}
\]

and equation 74 becomes (Drop ~),
\[
\frac{Pe_r}{Pe} \frac{\partial C}{\partial t} = -\frac{a}{L}Pe_r \frac{\partial u C}{\partial x} + 2D_{xx} Pe_r \frac{\partial^2 C}{\partial x^2} + 2D_{xy} Pe_r \frac{\partial^2 C}{\partial x \partial y} + 2D_{yy} Pe_r \frac{\partial^2 C}{\partial y^2} + Pe_r \frac{\partial^2 C}{\partial \theta^2} - Pe_r \frac{\partial \omega C}{\partial \theta}
\] (75)
The boundary conditions are the no flux boundary conditions on the wall, periodic boundary condition on the orientation, the integral condition and the initial condition similar to Taylor’s case,

$$\frac{\partial C}{\partial y} \bigg|_{\pm 1} = 0 \quad (76)$$

$$C(x, y, 0, t) = C(x, y, 2\pi, t) \quad (77)$$

$$\int_{-\infty}^{\infty} \int_{-1}^{1} \int_{0}^{2\pi} C(x, y, \theta, t) d\theta dy dx = 1 \quad (78)$$

$$C(x, y, \theta, t) = C(x, y, \theta, t = 0) = \frac{1}{4\pi} \delta(x) \quad (79)$$

### 5.3 Simplification of master equation

The above equation is separated based on the time scales. Since the rotational time scale is well separated from the diffusive time scale, two independent coupled problems can be solved to obtain the effective diffusivity. Separating the orientaional and transitional flux for long times, the orientational distribution reduces to the rotational advection diffusion equation or equation 47.

$$\frac{d^2C}{d\theta^2} - \text{Pe}_r \frac{d\omega C}{d\theta} = 0 \quad (80)$$

with the boundary conditions for the problem as the periodic boundary condition, $C(0) = C(2\pi)$ and the integral boundary condition,

$$\int_{0}^{2\pi} C(\theta) d\theta = 1. \quad (81)$$

To implement the integral boundary condition, it is converted to a third order Boundary Value Problem (BVP) by defining,

$$C(\theta) = f'(\theta) \quad (82)$$

and the ODE is,

$$f'''(\theta) - \omega(\theta)f''(\theta) - \omega'(\theta)f'(\theta) = 0. \quad (83)$$

The boundary conditions on $f$ are, $f(0) = 0$, $f(2\pi) = 1$ and $f'(0) = f''(2\pi)$. This BVP is much easier to integrate.
Based on the solution to this equation, a new spatially-dependent average lateral diffusion constant \( \bar{D}_y \) can be defined. The following is an ODE, with a local shear rate. Splitting the domain into different shear layers, we have a system of ODE’s for different shear layers, or different \( C_\theta \) for each shear layer. Integrating and solving each of the ODE’s on Mathematica for different shear layers we obtain a diffusion coefficient which is a function along the width of the channel.

\[
\bar{D}_y(y) = \frac{\int_0^{2\pi} C(\theta)D_{yy}(\theta)d\theta}{\int_0^{2\pi} C(\theta)d\theta}
\]  

(84)

where \( C_\theta \) is the orientational distribution at each shear layer. For different shear layers we have different orientational distributions. \( D_{yy} \) has been defined in the rotated diffusion tensor or equation 69.

Figure 12: Distribution of diffusivity along the channel and orientational distributions for corresponding shear layers. Here, \( p = 1000 \) and \( Pe_r = 100 \)

This local diffusivity is substituted in the translational non dimensional advection diffusion equation and simplified using Taylor’s arguments. The resulting equation is thus,

\[
Pe \left( \frac{1}{3} - y^2 \right) \frac{\partial C_t}{\partial x_1} = \frac{\partial}{\partial y} \left( \bar{D}_y(y) \frac{\partial C_t}{\partial y} \right).
\]  

(85)

The homogeneous solution is independent of \( \frac{\partial C_t}{\partial x_1} \) and the particular solution is directly proportional to \( \frac{\partial C_t}{\partial x_1} \).

\[
C_t = C_{tp} + C_{th}
\]

(86)

Hence the rate of mass transfer or the flux across the section \( x_1 \) is,

\[
Q = -\int_{-1}^{1} Pe \left( \frac{1}{3} - y^2 \right) C_t dy = \tilde{k} \frac{\partial C_t}{\partial x_1}.
\]

(87)
For different Rotational Peclet, $\tilde{\kappa}$ can be calculated using this approximation.

### 5.4 Theoretically Maximum Dispersion for Elongated Rods

The theoretical limit of dispersion is the condition when all elongated particles are aligned in the direction of the flow. Using Taylor’s arguments as discussed previously, the diffusion coefficient along the width of the channel becomes $D_\perp$. The resultant the effective diffusivity is,

$$\tilde{\kappa}_{\text{max}} = \frac{16}{945} \frac{Pe^2}{D_\perp}$$

where, $Pe$ is defined on the basis of the orientationally averaged diffusivity $\bar{D}$. $D_\perp$ Normalising this result with equation 20 gives us the maximum enhancement possible as compared to spherical particles.

![Figure 13: Theoretical maximum possible effective diffusivity coefficient](image)

Figure 13: Theoretical maximum possible effective diffusivity coefficient
5.5 Results

Figure 14: Overlaying the theoretical prediction onto figure 9

The theoretical prediction is a very good estimate for small $Pe_r$. It also does a good job in capturing the overall trend. As the Rotational Peclet number increases, the rotational time scale approaches the diffusive time scale and the approximation that the time scales are well separated fails. Another approximation that fails is that the mean speed of the particles is the same as the mean speed of the flow. This simple prediction driven by the physical motivation that a steady orientational state is rapidly achieved and gives rise to a spatially dependent diffusivity along the width of the channel appears to be a good approximation to capture the dominant physics of the problem.
6 Conclusion and Future work

Here, we present a numerical and theoretical study of Taylor Dispersion of Elongated Rods. Numerical simulations show that in the region of low $Pe_r$, the ellipsoidal particles behave like spherical particles and are constantly spinning. As the $Pe_r$ or shear rate increases, the particles tend to align themselves due to their Jeffery’s Orbit and have a lowered diffusion constant along the width of the channel. This causes them to have a greater effective diffusivity and therefore spread more. The situation where the $Pe_r$ is high, has a distribution that aligns with the flow. The use of elongated particles can also be used to enhance dispersion. Many applications like chemical reactions and mixing require enhanced dispersion. This mechanism is a way to enhance or even control dispersion.

In the future, we plan to further extend the theoretical predictions. Different methods like Generalised Taylor Dispersion\(^6\) or the homogenisation theory\(^10\) could be used to get further corrections to our simplified theory. Homogenisation theory allows us to rigorously exploit the naturally separated time scales that arise in our problem.
References


A Appendix

The appendix includes the code of each of the simulations and the finite difference solvers for each section. The MATLAB scripts have been attached. You can copy the following and paste the functions in MATLAB.

A.1 MATLAB: Monte Carlo Taylor Dispersion Spherical Particles

The following is the function that executes the Monte Carlo method for Taylor Dispersion. Followed by the code to run the code over many CPU’s and lastly to combine statistics of the many runs.

```matlab
function Monte_Carlo_Taylor_Dispersion_2D(steps,N,Pe,tfinal,r)
%steps is the number of time steps you want to run. tfinal is the final time
%till which you want your simulation to run. dt=tfinal/steps and has to be
%greater than 0.001 to ensure the particles don’t bounce multiple times
%from the wall. Pe is the Peclet number. The entire simulation is non
%dimensional and the time is normalised by diffusive time. x and y with the
%half width of the channel. r is the core number or the simulation number,
%in the parallelisation. N is the number of particles per core. Total
%number of particles simulated is N*r.
format long
count = tfinal/ steps;
%initial conditions for time and space.
s=1;
%initial variance in x if s=0, it is a initial condition that is a
%dirac delta funtion in x
x=((normrnd(0,s,[1,N])));
%normal distribution of N particle at t=0,
%with mean as 0 and variance as s. (Gaussian solution like pure diffusion)

y=((2*rand(1,N)) - 1); %Randomly generate an approximate plug of N
% particles along y axisat t=0. The plug is randomly distributed between
```
%+1 and -1. Similar to initial condition for Taylor Dispersion

\[ t = \text{zeros}(1, \text{steps}+1); \quad \text{time vector}. \]

\[ \text{mux} = \text{zeros}(1, \text{steps}+1); \quad \text{% mean position of particles in x for all times.} \]
\[ \text{mux}(1) = \text{mean}(x); \quad \text{% mean of initial distribution.} \]

\[ \text{variancex} = \text{zeros}(1, \text{steps}+1); \quad \text{% variance vector in x.} \]
\[ \text{variancex}(1) = \text{var}(x); \quad \text{% initial variance vector in x. should be equal to s} \]

%Actual Monte Carlo simulation

\[ \text{for } i = 1: \text{steps } \text{% time stepping} \]
\[ \quad t(i+1) = \text{count} \times i; \quad \text{% update time vector} \]
\[ \quad \text{for } j = 1: N \quad \text{% applying the physics for each particle.} \]
\[ \quad \quad \text{% Looping over each particle} \]
\[ \quad \quad y(j) = y(j) + (\sqrt{2 \times ((t(i+1) - t(i)))}) \times \text{normrnd}(0, 1); \]
\[ \quad \quad \text{% Random walk in y dictated physics will change this} \]
\[ \quad \quad \text{while } (y(j)) > 1 \quad \text{% bouncing particles back into the channel} \]
\[ \quad \quad \quad \text{% as y cannot be more than 1 or less than -1.} \]
\[ \quad \quad \quad \text{bounce} = (y(j)) - 1; \]
\[ \quad \quad \quad y(1,j) = (y(j)) - 2 \times \text{bounce}; \]
\[ \quad \quad \text{end} \]
\[ \quad \text{while } (y(j)) < -1 \]
\[ \quad \text{bounce} = (y(j)) + 1; \]
\[ \quad y(j) = (y(j)) - 2 \times \text{bounce}; \]
\[ \text{end} \]
\[ x(j) = x(j) + \text{Pe} \times (1 - ((y(j))^2)) \times ((t(i+1) - t(i))) + (\sqrt{2 \times ((t(i+1)) - ... - t(i)))}) \times \text{normrnd}(0, 1)); \quad \text{% Pe(1-y^2) is the advection and second is diffusion. Dictated physics will change this.} \]
end
mu(x(i+1)) = mean(x); %Mean at each time step.
variancex(i+1)=var(x); %Variance at each time step
end
sprintf("Run number %d has been completed",r)
csvwrite(sprintf('MC meanx = %d.csv',r),mu(x))
%Save the mean vector for 'rth' run to combine statistics later
csvwrite(sprintf('MC variance = %d.csv',r),variancex)
%Save variance vector
csvwrite(sprintf('MC y = %d.csv',r),y)
%Save final time step y position
csvwrite(sprintf('MC x = %d.csv',r),x)
%Save final time step x position
if r==1
csvwrite(sprintf('MC t = %d.csv',r),t)
end
%Save only one time vector.
end

function parallel_running(steps,N,Pe,tfinal,r)
format long
%running simulation.
parfor m=1:length(r) %parallel execution of code
    Monte_Carlo_Taylor_Dispersion_2D(steps,N,Pe,tfinal,r(m))
end
%The function is executed on many cores.
end
sprintf("All runs have been completed combining statistics next")
end

function stats_combine_sphere(steps,runs,N,Pe)
format long
steps=steps+1; %including t=0;
Np=N*runs;
sprintf("Total number of particles whose stats have been combines is \%d"...,
   Np)
variancex=zeros(runs,steps);
%Variable to load variance if each run
mu=zeros(runs,steps);
%variable to load mean of each run
muaverage = zeros(1,steps);
%the average mean of means of all runs
musq = zeros(runs,steps);
%square of mean - mean of mean, needed to calculate combined variance
variance = zeros(1,steps);
%The actual variance combined.
musqsum = zeros(1,steps);
%Sum of squares - mean of mean, needed to calculate variance
t=csvread(sprintf('MC t = 1.csv'));
%Read time vector

%actually loading the files and analysing the data
for i=1:runs
   mu(i,:)=csvread(sprintf('MC meanx = \%d.csv',i));
   variancex(i,:)=csvread(sprintf('MC variancex = \%d.csv',i));
end
%Read and load every runs mean and variance
for j=1:steps

36
muaverage(j) = mean(mu(:,j));

% Take the mean of all means
for i=1:runs
    musq(i,:) = (mu(i,:)- muaverage).^2;
end

% Mean of individual runs subtracted from the average of means at each time
% step.
for j=1:steps
    musqsum(j) = sum(musq(:,j));
    variance(j) = ((sum(variancex(:,j)) + musqsum(j))/(runs);
end

% Calculate variance using formula for combining variance.
csvwrite(sprintf("Variance after statscombine for Pe=%d.csv",Pe),variance)
% Write combined variance vector
csvwrite(sprintf("Mean after statscombine for Pe=%d.csv",Pe),muaverage)
% Write combined mean vector
csvwrite(sprintf("Time vector for Pe=%d.csv",Pe),t)
% Write time vector.

% Lastly plot variance vs time and obtain effective diffusivity from the
% linear section of the variance vs time.
end

function run_file_Monte_Carlo(steps,N,Pe,tfinal,runs)
% steps is the number of time steps. N is the number of particles simulated
% on each core. Pe is Peclet number. tfinal is the final time step till
% which the simulation has to be done (1 is enough). runs is the number of
% runs you want to execute.
format long
r=linspace(1,runs,runs); % 'rth' run
Please follow the following instructions to execute the code.

- Copy the four MATLAB function and save them to your system for execution.
- Monte Carlo Taylor Dispersion 2D is the actual simulation executed on each core. Edit the physics according to your convenience. i.e. adding some self propulsion, removing Brownian motion etc.
- Parallel running is to actually run it on many CPU’s and stats combine spheres is to combine the statistics and save the time vector, combined mean and variance vector.
- To execute the three functions, just use the last function run file Monte Carlo with the inputs of your choice. The output will be saved.

A.2 MATLAB: Monte Carlo Taylor Dispersion Ellipsoidal Particles

```matlab
%% Non Dimensional Diffusion Advection
function Monte_Carlo_Ellipsoids_Taylor_Dispersion(steps,N,Pe,Pe_rot,...
    ,p,tfinal,r)
%steps is the number of time steps you want to run. tfinal is the final
time till which you want your simulation to run. dt=tfinal/steps and has
to be greater than 0.0001 to ensure the particles don't bounce multiple
times from the wall and ensures the Jeffery’s orbit is well resolved.
%Pe is the Peclet number, p the aspec ratio, Pe_r is the rotational Peclet
% number. The entire simulation is non dimensional and
% the time is normalised by diffusive time. x and y with the
%half width of the channel. r is the core number or the simulation number,
%in the parallelisation. N is the number of particles per core. Total
%number of particles simulated is N*r.
format long
count = tfinal/ steps; %dt
```
\%alpha is D_{parallel}/D_{perp}

if \( p = 1 \)

\[ \alpha = 1; \]  \%the \( p = 1 \) returns the case for spheres but checks the
\% orientation so that the particles are spinning in cases where the
\% angle matters like the case of active matter.

else

\[ \alpha = \left( p \cdot \left( (1-p^2)^{-1} + (1-p^2)^{-3/2} \cdot \left( 2 \cdot (1-p^2) \cdot \log((1-p^2)^{(1/2)}) \right) \right) \right)^{-1}; \]

end \%Expression obtained from Happel and Brenner.

\texttt{s=1;}
\%initial variance in x if \( s = 0 \), it is a initial condition that is a
\%dirac delta funtion in x

\texttt{x=((normrnd(0,s,[1,N])));
\%normal distribution of N particle at t=0,
\%with mean as 0 and variance as s. (Gaussian solution like pure diffusion)

\texttt{y=((2*rand(1,N)) - 1);
\%Randomly generate an approximate plug of N
\% particles along y axis at t=0. The plug is randomly distributed between
\%+1 and -1. Similar to intial condition for Taylor Dispersion

\texttt{t=zeros(1,steps+1); \%time vector.
theta = 2*pi*(zeros([1,N])); %Random orientational distribution of particles.

mux=zeros(1,steps+1); % mean position of particles in x for all times.
mux(1) = mean(x); % mean of initial distribution.

variancex=zeros(1,steps+1); % variance vector in x.
variancex(1) = var(x); % initial variance vector in x. should be equal to s

% Actual Mote Carlo simulation
for i=1:steps % time stepping
    t(i+1)=count*i; % update time vector
    for j=1:N % applying the physics for each particle.
        % Looping over each particle
        ncap = sqrt(2*((2*alpha)/(1+alpha))*(t(i+1)-t(i)))*normrnd(0,1);
        % Brownian motion in the parallel direction of the ellipsoidal particle
        kcap = sqrt(2*(2/(1+alpha))*(t(i+1)-t(i))) * normrnd(0,1);
        % Brownian motion in the perpendicular direction of the ellipsoidal particle.
        A = [x(j);y(j)] + [cos(theta(j)), -sin(theta(j)); ...
            sin(theta(j)), cos(theta(j))]*[kcap ; ncap] + ...
        [Pe*(1-((y(j)^2))*(t(i+1)-t(i)))*0];
        % We calculate the change in the x and y position of each particle. The first is the diffusion in the parallel and perpendicular direction % rotated to be in the direction of the flow i.e. appropriate
% coordinate system. The second term is the advection.
y(j) = A(2,1);
while (y(j))>1 %bouncing particles back into the channel
    %as y cannot be more than 1 or less than -1.
    bounce=(y(j))-1;
    y(j)=(y(j))-2*bounce;
end
while (y(j))<-1
    bounce=(y(j))+1;
    y(j)=(y(j))-2*bounce;
end
x(j) = A(1,1);
theta(j) = mod(theta(j) - (t(i+1)-t(i))...  
*2*Pe*y(j)*(((cos(theta(j)))^2) - 2) + ...  
((p*sin(theta(j)))^2)/(1 + (p^2)))+...
    sqrt(2*(Pe/Pe_rot)*(t(i+1)-t(i)))*normrnd(0,1,2*pi);
end
% Updating the angular orientation of each particle. The first is the
% Jeffery Orbits terms or the rotational advection. The second term is
% the rotational brownian motion. The mod 2pi ensures the the
% periodicity of the particle.
% All these can be changed depending on the physics for non interacting
% particles.
mux(i+1) = mean(x); %Mean at each time step.
variancex(i+1)=var(x); %Variance at each time step
%More statistics like the mean of y position etc can be calculated
end
sprintf("Run number %d has been completed",r)
csvwrite(sprintf('MC Ellipsoid mux = %d.csv',r),mux)
%Save the mean vector for 'rth' run to combine statistics later
csvwrite(sprintf('MC Ellipsoid variancex = %d.csv',r),variancex)
%Save variance vector
csvwrite(sprintf('MC Ellipsoid ynd = %d.csv',r),y)
%Save final time step y position
csvwrite(sprintf('MC Ellipsoid xnd = %d.csv',r),x)
%Save final time step x position
csvwrite(sprintf('MC Ellipsoid theta = %d.csv',r),theta)
%Save final time step theta position
if r==1
    csvwrite(sprintf('MC t = %d.csv',r),t)
end
%Save only one time vector.
end

function parallel_running_ellispoid(steps,N,Pe,Pe_rot,p,tfinal,r)
    format long
    %running simulation.
    parfor m=1:length(r) %parallel execution of code
        Monte_Carlo_Ellipsoids_Taylor_Dispersion(steps,N,Pe,Pe_rot,p,...
            tfinal,r(m))
        %The function is executed on many cores.
    end
    sprintf("All runs have been completed combining statistics next")
end

function stats_combine_ellipsoid(steps,runs,N,Pe,Pe_rot,p)
    format long
    steps=steps+1; %including t=0;
    Np=N*runs;
    sprintf("Total number of particles whose stats have been combines is %d"...
\texttt{variancex=zeros(runs,steps);}

\%Variable to load variance if each run
\texttt{mu=zeros(runs,steps);}

\%variable to load mean of each run
\texttt{muaverage = zeros(1,steps);}

\%the average mean of means of all runs
\texttt{musq = zeros(runs,steps);}

\%square of mean - mean of mean, needed to calculate combined variance
\texttt{variance = zeros(1,steps);}

\%The actual variance combined.
\texttt{musqsum = zeros(1,steps);}

\%Sum of squares - mean of mean, needed to calculate variance
\texttt{t=csvread(sprintf('MC t = 1.csv'));}  
\%Read time vector

\%actually loading the files and analysing the data
\texttt{for i=1:runs}
\hspace{1em} \texttt{mu(i,:) = csvread(sprintf('MC Ellipsoid mux = %d.csv',i));}
\hspace{1em} \texttt{variancex(i,:) = csvread(sprintf('MC Ellipsoid variance = %d.csv',i));}
\texttt{end}

\%Read and load every runs mean and variance
\texttt{for j=1:steps}
\hspace{1em} \texttt{muaverage(j) = mean(mu(:,j));}
\texttt{end}

\%Take the mean of all means
\texttt{for i=1:runs}
\hspace{1em} \texttt{musq(i,:) = (mu(i,:)- muaverage).^2;}
\texttt{end}

\%mean of individual runs subtracted from the average of means at each time
\%step.
for j=1:steps
    musqsum(j) = sum(musq(:,j));
    variance(j) = ((sum(variancex(:,j))) + musqsum(j))/(runs);
end

% calculate variance using formula for combining variance.
csvwrite(sprintf("Pe=%d Variance after statscombine for aspect ratio=%d Pe_r=%d.csv",...
    Pe,p,Pe_rot),variance)
% write combined variance vector
csvwrite(sprintf("Pe=%d Mean after statscombine for aspect ratio=%d Pe_r=%d.csv",...
    Pe,p,Pe_rot),muaverage)
% write combines mean vector
csvwrite(sprintf("Pe=%d Time vector for aspect ratio=%d Pe_r=%d.csv",...
    Pe,p,Pe_rot),t)
% write time vector.
% Lastly plot varianace vs time and obtain effective diffusivity from the
% linear section of the variance vs time.
end

function run_file_Monte_Carlo_ellipsoid(steps,N,Pe,Pe_rot,p,tfinal,runs)
% steps is the number of time steps. N is the number of particles simulated
% on each core. Pe is Peclet number. tfinal is the final time step till
% which the simulation has to be done (1 is enough). runs is the number of
% runs you want to execute.
format long
r=linspace(1,runs,runs); % 'rth' run
parallel_running_ellipsoid(steps,N,Pe,Pe_rot,p,tfinal,r) % parallel execution of code
stats_combine_ellipsoid(steps,runs,N,Pe,Pe_rot,p) % combine stats.
sprintf("Done!");
end
A.3 Hinch’s Shear Induced Migration DNS code

clear
format long
tic

% Defining all constants

p = 1000; % Aspect Ratio
Pe_r = 10; % Rotational Peclet Number
Pe = 1000; % Transitional Peclet Number

e = Pe_r / Pe; % Ratio of rotational time scale to diffusive time scale.
if p == 1
    alpha = 1; % Dperp/Dpara
else
    alpha = (p.*((-1)+p.^2).^(-1)...
             +((-1)+p.^2).^(-3/2).*((-3)+2.*p.^2).*log(p+((-1)+p.^2).^...
             (1/2))).*(2.*p.*(1+(-1).*p.^2).^(-1)+((-1)+p.^2).^(-3/2).*...
             ((-1)+2.*p.^2).*log((p+(-1).*((-1)+p.^2).^(-1/2)).^...
             (-1).*((p+(-1)+p.^2).^(-1/2))).^(-1);
end

% Initialize Grid for solver
yspace = linspace(-1,1,200);
thetaspace = linspace(0,pi,100);
dy = yspace(2)-yspace(1);
dtheta = thetaspace(2)-thetaspace(1);
ny = length(yspace);
 ntheta = length(thetaspace);

% Declare time stepping
dt = (dy^2 + dtheta^2) / 3;
tcount = 0;

% Create Grid for solution
C = (ones([ny+1, ntheta+1])) * (1/(2*pi));
c2 = zeros(1, ny);  % C(\theta, y) integrated from 0 to 2pi
cnorm1 = zeros(1, ny);  % Normalisation at each time step to ensure the
% integral condition is satisfied.
Cnew = C;  % next time step solution
Cold = zeros([ny+1, ntheta+1]);  % previous time step distribution.
err = ((sqrt(sum((sum((abs((Cold-Cnew)).^2)))/(ny+ntheta+2))));
% L2 Norm at each step.

while ((err>10^-8) && (err<10))  % loop till steady solution is obtained
    tcount = tcount + dt;
    % time stepping to obtain Steady State Solution
    for j = 2: ntheta  % Iterate over \theta
        for i = 2: ny  % Iterate over y
            % Finite Differencing.
            residue1 = dt * ((e*(((2*(((sin(j-1)*dtheta))^2) + (2*alpha*(((cos((j-1)*dtheta))^2)/(1 + alpha))))*(((C(i+1,j) - 2*C(i,j) + C(i-1,j))/(dy^2))));
            residue2 = dt * (((C(i,j+1)-2*C(i,j)+C(i,j-1))/(dtheta^2));
            residue3 = dt * (((p^2-1)/(p^2+1))*(sin(2*dtheta*(j-1)))*2*Pe_r*(((i-1)*dy -1)))*((C(i,j)));
            residue4 = dt * (((p*sin((j-1)*dtheta))^2) + (((cos((j-1)*dtheta))^2)/(p^2 + 1)))*2*Pe_r*(((C(i,j+1)-C(i,j-1))/(2*dtheta)));
            Cnew(i,j) = C(i,j) + residue1 + residue2 - residue3 - residue4;
        end
    end
Cnew(:,end)=Cnew(:,2);  %Periodic BC (Ghost Point)
Cnew(:,1)=Cnew(:,end-1);  %Periodic BC
Cnew(end,:)=Cnew(end-1,:);  % No flux Boundary condition
Cnew(1,:)=Cnew(2,:);  % No flux Boundary condition

for o=1:ny
    cnorm1(o)=trapz(thetaspace,C(o,(1:end-1)));
end  %normalise to make sure integral boundary condition is satisfied.

% Integrate over theta

for m=1:ny
    c2(m)=trapz(thetaspace,C(m,(1:end-1)));
end

c3=trapz(yspace,c2);
c2=(c2/c3);
csvwrite(sprintf(’c2_Hinch_distribution Pe_r=%d.csv’,Pe_r),c2);
toc