Imperial College London
Department of Aeronautical Engineering

# Immersed Boundary Methods with explicitly enforced viscous forces and pressure gradient equilibrium at the solid boundary

Charles Luzzato

June 2011

Supervised by J.C. Vassilicos

Submitted in part fulfilment of the requirements for the degree of
Master of Engineering in Aeronautical Engineering of Imperial College London

# Abstract

Immersed boundary methods are used to account for the boundary conditions of solids in CFD simulations by adding forcing functions to the Navier Stokes equations. The current implementation of IBMs with Dirichlet no-slip boundary conditions limit the formal accuracy of the scheme to 2nd order. These implementations do not explicitly balance viscous forces and pressure gradient at the boundary. The presence of the solid is account for by variations of velocity only. By imposing a new surrogate source field within the simulation, and explicitly imposing a balance between the pressure gradient and viscous forces at the boundaries of the solid, it is expected that the order of accuracy will increase. This paper deals with the formulation of the IBM problem with imposed surrogate sources, and resulting variations of pressure. It also covers different methods that can be used to determine the surrogate source field. The numerical implementation of these methods, and their limitations are also explained.

The implementation of the methods described in this report in a CFD code produces encouraging results, with small variations of pressure and velocity in the areas around the solid. The order of accuracy of the IBM methods with equilibrium of pressure gradient and viscous forces at the boundary still needs to be determined.

# Contents

# List of Figures

# Introduction

One of the objectives of fundamental turbulence research is to be able to capture all scales of turbulence in Direct Numerical Simulations (DNS), whilst conserving acceptable calculation times and accuracy. For this task, the Finite Difference Method (FDM) is commonly used for its low dissipative and dispersive properties. In the case of simulations around complex structures, body conformal grids must be used. This leads to cumbersome grid construction, deterioration in grid quality, and can negatively impact convergence and calculation time [4]. The concept of Immersed Boundary Methods (IBM) is to create a nonbody conformal cartesian grid. In other words, the grid surface used to discretise the boundary of the solid is completely independent from the cartesian grid used to discretise the simulation domain. To account for the effects of the solid boundary, a forcing function is introduced in the Navier Stokes equations by imposing either direct or indirect boundary condition. These methods currently limit themselves to verifying a null velocity field in the solid and on the boundary. As such, they only offer a 2nd order formal accuracy [5]. It is suspected that this limitation comes from the fact that the pressure gradient is not explicitly set to balance viscous forces on the Boundary. Some argue that this balance comes naturally from the imposition of the velocity field within the solid.

This report shows how the balance of the pressure gradient and viscous forces at the boundary can be obtained explicitly. This is done through the introduction of a new surrogate source field. The paper also explains the techniques used to determine the source field through methods based on finite elements and Dirac distributions. It is hoped that the imposition of the pressure gradient and viscous field balance will lead to an increase in the order of accuracy of the calculation.

The structure of the paper is as follows. The first chapter deals with the description of the IBM problem, and the formulation of the expression that needs to be satisfied by the surrogate source filed. The second chapter deals with techniques developed to find adequate surrogate source fields which satisfy the set conditions for IBM. The third chapter considers the structure of the numerical algorithm used to determine the surrogate source field, and underlined important accuracy issues. Finally, the final chapter gives applied test cases of the algorithm.

# 1. Brief overview of Immersed Boundary Methods

## 1.1. Governing equations

The governing equations of fluid flow are the Navier Stokes Equations (NSE), two of those also known as the equation for the conservation of mass (1.1), and the conservation of momentum (1.2). Here, $\rho$ is the volumic mass, $\mathbf{u}$ is the speed vector, $\otimes$ is the tensor product, $p$ is the pressure, $(\overline{\overline{\tau}})$ is the viscous stress tensor, $\mathbf{f}$ is the applied external force vector.

$$\frac{\partial \rho}{\partial t} + \operatorname{div}(\rho \mathbf{u}) = 0 \tag{1.1}$$

$$\frac{\partial (\rho \mathbf{u})}{\partial t} + \operatorname{div}(\rho \mathbf{u} \otimes \mathbf{u}) = -\overrightarrow{\operatorname{grad}}(p) + \operatorname{div}(\overline{\overline{\tau}}) + \rho \mathbf{f} \tag{1.2}$$

In the aeronautics field, air is considered to be Newtonian fluid. Furthermore, only incompressible viscous flow will be considered. Using these assumptions, the NSE reduce to (1.3) and (1.4), where $P = \frac{p}{\rho}$ and $\nu = \frac{\mu}{\rho}$.

$$\overrightarrow{\nabla} \cdot \mathbf{u} = 0 \tag{1.3}$$

$$\frac{\partial \mathbf{u}}{\partial t} + \left(\mathbf{u} \cdot \overrightarrow{\nabla}\right) \mathbf{u} = -\overrightarrow{\nabla} P + \nu \nabla^2 \mathbf{u} \tag{1.4}$$

## 1.2. Principle of Immersion Boundary Method

Solving for viscous flow around a solid using DNS requires a very fine mesh around the solid to be able to capture all scales of turbulence near the solid boundary. Two different kinds of meshes can be used to do this. A very fine structured mesh can be used throughout the whole flow domain; this means that regions of flow far away from the solid will have a necessarily fine discretisation which will require great amounts of computational time. The second method is to use an unstructured mesh, where the mesh gets progressively finer as you approach the solid. The disadvantage of this is that unstructured meshes are more computationally expensive because of the reference table systems they use. Furthermore, unstructured meshes are not well adapted to high order low dissipative and dispersive FDM.

To avoid these problems, the IBM is designed to make away with the solid all together and replace it with an artificial forcing acceleration. This means that the boundary of the solid does

not need to be discretised, and so the meshing problem explained in the previous paragraph is removed.

Take a domain defined by $D_s \cup D_f \cup \delta D$ where $D_s$ is the solid excluding its boundary $\delta D$ , $D_f$ is the fluid excluding the boundary of the solid $\delta D$.. Let us consider that $\mathbf{u}$ defines the velocity field in the case where the solid is present, and $\mathbf{v}$ the velocity field when the solid is replaced by IBM method accelerations. Therefore, $\mathbf{u}$ is the solution to the standard form of NSE given in (1.3) and (1.4) and is defined on $D_f \cup \delta D$ only. Also $\mathbf{v}$ is the solution to the modified NSE given in (1.5) and (1.6) and is defined on $D_f \cup D_s \cup \delta D$, where $P_v = \frac{p_v}{\rho_v}$. $C$ is defined as 0 in $D_f \cup \delta D$, and can take any value in $D_s$. Furthermore, the first and second order spatial derivatives of $C$ are 0 on $\delta D$.

$$\overrightarrow{\nabla} \cdot \mathbf{v} = C \tag{1.5}$$

$$\frac{\partial \mathbf{v}}{\partial t} = \begin{cases} 0 \text{ on } D_s \cup \delta D \\ \left(-\mathbf{v} \cdot \overrightarrow{\nabla}\right) \mathbf{v} - \nabla P_v + \nu \nabla^2 \mathbf{v} \text{ on } D_f \end{cases} \tag{1.6}$$

Expression (1.6) can be rewritten as a function of $\gamma$, such that $\gamma = 1$ on $D_f$ and $\gamma = 0$ on $D_s \cup \delta D$. Thus we obtain expression (1.7). This is equivalent to adding a forcing acceleration $\mathbf{f}$ defined on $D_s \cup D_f \cup \delta D$ to the NSE as shown in (1.8). This finally allows us to write $\mathbf{f}$ as a function of $\mathbf{v}$ and $\gamma$ in (1.9).

$$\frac{\partial \mathbf{v}}{\partial t} = -\gamma \left( \left( \mathbf{v} \cdot \overrightarrow{\nabla} \right) \mathbf{v} + \nabla P_v - \nu \nabla^2 \mathbf{v} \right) \tag{1.7}$$

$$\frac{\partial \mathbf{v}}{\partial t} = -\left( \mathbf{v} \cdot \overrightarrow{\nabla} \right) \mathbf{v} - \nabla P_v + \nu \nabla^2 \mathbf{v} + \mathbf{f} \tag{1.8}$$

$$\Leftrightarrow \mathbf{f} = (1-\gamma) \left( \left( \mathbf{v} \cdot \overrightarrow{\nabla} \right) \mathbf{v} + \nabla P_v - \nu \nabla^2 \mathbf{v} \right) \tag{1.9}$$

To prove that IBM methods are mathematically correct, we need to prove the that in the region $D_f \cup \delta D$, $\mathbf{v} = \mathbf{u}, \forall t$ when using the same initial conditions on $D_f \cup \delta D$.

## 1.3. Unicity of the Navier Stokes Equations

### 1.3.1. Development of the unicity of the Navier Stokes equations

See A for the development leading to the conclusions to the unicity of the Navier Stokes equations. The difference between the 2 solutions is written as $\mathbf{w}(\mathbf{x}, t) = \mathbf{v}(\mathbf{x}, t) - \mathbf{u}(\mathbf{x}, t)$, and the pressure field divided by $\rho$ associated to the $\mathbf{w}$ is given as $P_w$

### 1.3.2. Conclusions to the unicity of the Navier Stokes equations

In A.1, it was shown that $\mathbf{u} = \mathbf{v}$ on $D_f \cup \delta D$ under the following assumptions :

1. $P_w$ is not equal to infinity on $\delta D$

2. $\left(\frac{\partial}{\partial x_j}\mathbf{w}\right)\cdot\mathbf{n}$ is not equal to infinity $\delta D$

3. $\int_0^t \|\mu_{\inf}\| d\tau$ is integrable so that $K(t) \neq \inf$

The last condition remains an open problem in the analysis of the Navier Stokes equations pertaining to the analysis of the strain rate tensor. Indeed, discontinuous velocity fields generated as solutions of the NSE will lead to non integrable singularities in the eigenvalues of the strain rate tensor. The first two conditions for the unicity of the NSE depend on pressure and velocity fields on the boundary $\delta D$, and are discussed in the next section.

## 1.4. Pressure and velocity boundary conditions

The solution $\mathbf{u}$ satisfies the NSE (1.3) and (1.4). It has been stated that on $\delta D$, $\mathbf{u}(\mathbf{x},t) = 0$. Thus, by considering steady flow where $\frac{\partial \mathbf{u}}{\partial t} = 0$, then the RHS of (1.4) is equal to 0, and (1.4) reduces to (1.10) on the boundary $\delta D$. Note that (1.10) ensures that neither $P$ nor $\overrightarrow{\nabla}\mathbf{u}$ are infinite on $\delta D$. The objective here is therefore to find the conditions so that $\mathbf{v}$ and $P_v$ are also solutions to (1.10) on $\delta_s$.

$$-\overrightarrow{\nabla}P + \nu\overrightarrow{\nabla}^2\mathbf{u} = 0 \tag{1.10}$$

Expression (1.8) gives us the governing equation for $P_v$ and $\mathbf{v}$ as a function of a forcing acceleration $\mathbf{f}$. By taking the divergence of (1.8), expression (1.11) is obtained.

By using (1.5) and (1.9), we obtain expression (1.12). Note that $\delta_{\delta D}(\mathbf{x})$ is the Dirac delta function defined such that it is equal to 0 on $D_s \cup D_f$, and $mathbfn$ is the normal to $\delta D$.

$$\begin{aligned}
\frac{\partial\left(\overrightarrow{\nabla}\cdot\mathbf{v}\right)}{\partial t} &= -\overrightarrow{\nabla}\cdot\left(\mathbf{v}\cdot\overrightarrow{\nabla}\right)\mathbf{v} - \nabla^2 P_v + \nu\nabla^2\left(\overrightarrow{\nabla}\cdot\mathbf{v}\right) + \overrightarrow{\nabla}\cdot\mathbf{f} && (1.11)\\
\frac{\partial C}{\partial t} &= -\overrightarrow{\nabla}\cdot\left(\mathbf{v}\cdot\overrightarrow{\nabla}\right)\mathbf{v} - \nabla^2 P_v + \nu\nabla^2 C \\
&+ (1-\gamma)\left(\overrightarrow{\nabla}\cdot\left(\mathbf{v}\cdot\overrightarrow{\nabla}\right)\mathbf{v} + \nabla^2 P_v - \nu\nabla^2 C\right) \\
&+ \delta_{\delta D}(\mathbf{x})\,\mathbf{n}\cdot\left(\left(\mathbf{v}\cdot\overrightarrow{\nabla}\right)\mathbf{v} + \nabla P_v - \nu\nabla^2\mathbf{v}\right) && (1.12)
\end{aligned}$$

**Solid Region** Looking specifically at the region $D_s$, i.e. the region where $\gamma = 0$, expression (1.12) simplifies to $\frac{\partial C}{\partial t} = 0$ as the terms from $\overrightarrow{\nabla}\cdot\mathbf{f}$ cancel out all other terms on the RHS. This is true whatever the values for $\mathbf{v}, P_v$ or $C$

**Fluid only Region** In $D_f$, $\gamma$ was defined as 1, and $\delta_{\delta D}(\mathbf{x}) = 0$. As $C = 0$ in $D_f$, then $\frac{\partial C}{\partial t} = 0$ as well in $D_f$ and so $\frac{\partial C}{\partial t} = 0$ for $D_s \cup D_f$. In these conditions, (1.12) simplifies to $\nabla^2 P_v = -\overrightarrow{\nabla}\cdot\left(\mathbf{v}\cdot\overrightarrow{\nabla}\right)\mathbf{v}$

**Solid boundary** Here, $\gamma$ was defined as 0, and $\delta_{\delta D}(\mathbf{x}) = 1$. On $\delta D$, from the definition of $C$ the first and second order spatial derivatives of $C$ are equal to 0. Furthermore, $\frac{\partial C}{\partial t} = 0$ for $D_f \cup D_s$ from the previous two paragraphs. Thus, by deduction, $\frac{\partial C}{\partial t} = 0$ on $\delta D$. Also, from the boundary conditions defined previously, $\mathbf{v} = 0$. Using these two facts, (1.12) simplifies to $\mathbf{n} \cdot \left(\nabla P_v - \nu \nabla^2 \mathbf{v}\right) = 0$. This form ensures that neither $P_v$ nor $\overrightarrow{\nabla}\mathbf{v}$ are infinite on $\delta D$.

**Summary of results** The three previous paragraphs give three conditions so that neither $P_v$ nor $\overrightarrow{\nabla}\mathbf{v}$ to be infinite on $\delta D$.

- $\frac{\partial C}{\partial t} = 0$ on $D_s$
- $\nabla^2 P_v = -\overrightarrow{\nabla} \cdot \left(\mathbf{v} \cdot \overrightarrow{\nabla}\right) \mathbf{v}$ on $D_f$
- $\mathbf{n} \cdot \left(\nabla P_v - \nu \nabla^2 \mathbf{v}\right) = 0$ on $\delta D$

## 1.5. Surrogate Pressure Field and Surrogate source field

Section 1.4 gives three conditions which must be respected in order to conserve the unicity of the NSE , i.e $\mathbf{u} = \mathbf{v}$. However, writing down these conditions has required the use of Boundary Conditions which will not be available in the IBM methods, as there will be no boundary to impose them on. It is therefore necessary to re-obtain the previously expressed conditions independently of the boundary conditions that were given for $\delta D$. This is done by imposing a surrogate pressure field $P_s(\mathbf{x})$ and source field $S(\mathbf{x})$ defined on $D_f \cup D_s \cup \delta D$ in expression (1.13). In effect, these surrogate pressure and source fields are replacing the effects of the solid boundary on the fluid flow. Note that both source and pressure fields here are scalars.

$$\nabla^2 P_s = -\gamma \overrightarrow{\nabla} \cdot \left(\left(\mathbf{v} \cdot \overrightarrow{\nabla}\right) \mathbf{v}\right) + (1 - \gamma) S \tag{1.13}$$

**Solid region** In region $D_s$, it was seen in section 1.4 that no matter the value of the applied boundary conditions, and therefore no matter the value of the surrogate pressure and source field, the condition $\frac{\partial C}{\partial t} = 0$ is always kept.

**Fluid Region** On $D_f$, $\gamma$ has been set to 1, and therefore (1.13) reduces to $\nabla^2 P_s = -\gamma \overrightarrow{\nabla} \cdot \left(\left(\mathbf{v} \cdot \overrightarrow{\nabla}\right) \mathbf{v}\right)$. Thus, whithout any imposition of boundary conditions, the second condition of section 1.4 is verified.

**Solid boundary** On $\delta D$, where $\gamma = 0$, the source term $S(\mathbf{x})$ must be so that the third condition of section 1.4 is verified, or in other words that $\nabla P_s = \nu \nabla^2 \mathbf{v}$ on $\delta D$ only. The integration of expression (1.13) gives (1.14) on $D_f \cup D_s \cup \delta D$.

$$\nabla P_s = \int_{D_f \cup D_s \cup \delta D} \frac{-\gamma \overrightarrow{\nabla} \cdot \left(\left(\mathbf{v} \cdot \overrightarrow{\nabla}\right) \mathbf{v}\right) + (1 - \gamma) S(\mathbf{x}')}{|\mathbf{x} - \mathbf{x}'|^3} \left(\mathbf{x} - \mathbf{x}'\right) d\mathbf{x}' \tag{1.14}$$

Injecting (1.14) into the last condition of section 1.4 gives (1.15). If $S(\mathbf{x}')$ satisfies (1.15) on $\delta D$ only, then the unicity of the NSE is conserved, and so the IBM velocity field $\mathbf{v}$ is equal to the standard CFD simulation velocity field $\mathbf{u}$.

$$\int_{D_f \cup D_s \cup \delta D} \frac{-\gamma \nabla \cdot (\mathbf{v} \cdot \nabla \mathbf{v}) + (1-\gamma) S(\mathbf{x}')}{|\mathbf{x} - \mathbf{x}'|^3} \left(\mathbf{x} - \mathbf{x}'\right) d\mathbf{x}' = \nu \nabla^2 \mathbf{v} \tag{1.15}$$

## 1.6. Conclusion

The derivations from this chapter lead to expression (1.15) which must be satisfied by $S(\mathbf{x})$. If this is verified, then the gradient of pressure is explicitly set to balance the viscous forces on the $\delta D$.

Chapter 2 will be to explain methods for finding $S(\mathbf{x})$ such that (1.15) is satisfied.

# 2. Methods for the computation of the source function

## 2.1. Objective

In 1, it was shown that a specific scalar or vector source function $S(\mathbf{x})$, defined on $D_s$ as a vector and on $D_f \cup \delta D$ as scalar, is required to satisfy the condition $\nabla P_S = \nu \nabla^2 \mathbf{v}$ on $\delta D$. For this, the function $S(\mathbf{x})$ must satisfy the equation given in (2.1). Note that the variables $\mathbf{v}$, $\mathbf{x}'$ and $\mathbf{x}$ are all vectors.

$$\int_{D_F \cup D_S \cup \delta D} \frac{-\gamma \nabla \cdot (\mathbf{v} \cdot \nabla \mathbf{v}) + (1-\gamma) S(\mathbf{x}')}{|\mathbf{x} - \mathbf{x}'|^3} (\mathbf{x} - \mathbf{x}') \, d\mathbf{x}' = \nu \nabla^2 \mathbf{v} \tag{2.1}$$

Using the fact that we have $\gamma = 0$ on the solid and boundary and $\gamma = 1$ on the fluid, the integral can be separated over an integral over the fluid and the solid to give (2.2).

$$\int_{D_S \cup \delta D} S(\mathbf{x}') \frac{(\mathbf{x} - \mathbf{x}')}{|\mathbf{x} - \mathbf{x}'|^3} d\mathbf{x}' = \mathbf{F}(\mathbf{x}) \tag{2.2}$$

$$\text{with } \mathbf{F}(\mathbf{x}) = \nu \nabla^2 \mathbf{v} + \int_{D_F} \frac{\nabla \cdot (\mathbf{v} \cdot \nabla \mathbf{v})}{|\mathbf{x} - \mathbf{x}'|^3} (\mathbf{x} - \mathbf{x}') \, d\mathbf{x}' \tag{2.3}$$

In 1D, the vector variables given in equation (2.2) simplify to scalar values, and the integral becomes a line integral. However, when more than one dimensions is used, the integral is a surface of volume integral, and (2.2) becomes three distinct equations. For more clarity, (2.4) to (2.6) show the expanded version of (2.2). This form will be especially useful when the 2D and 3D numerical methods for finding the source function are developed.

$$\iiint_{D_S \cup \delta D} S(\mathbf{x}') \frac{(x - x')}{\sqrt{(x-x')^2 + (y-y')^2 + (z-z')^2}^3} dz' dy' dx' = F_x(\mathbf{x}) \tag{2.4}$$

$$\iiint_{D_S \cup \delta D} S(\mathbf{x}') \frac{(y - y')}{\sqrt{(x-x')^2 + (y-y')^2 + (z-z')^2}^3} dz' dy' dx' = F_y(\mathbf{x}) \tag{2.5}$$

$$\iiint_{D_S \cup \delta D} S(\mathbf{x}') \frac{(z - z')}{\sqrt{(x-x')^2 + (y-y')^2 + (z-z')^2}^3} dz' dy' dx' = F_z(\mathbf{x}) \tag{2.6}$$

To numerically determine the function $S(\mathbf{x})$, it can be expressed as a linear combination of

elementary basis distributions in a vector space using two methods : the Dirac source distribution method and the Lagrange Function distribution method. These are both explained below.

## 2.2. Dirac source distribution method

### 2.2.1. General Theory

Using the theory of distributions ([3],[1], [6]), source function $S(\mathbf{x})$ can be linearised as the sum of Dirac functions and their coefficients. Note that the quality of the linearisation depends on the intersection of the functional space $\Omega$ generated by $\mathbf{F}(\mathbf{x})$, and the vector space $\Phi$ generated by $\int_{D_S \cup \delta D} S(\mathbf{x}') \frac{(\mathbf{x}-\mathbf{x}')}{|\mathbf{x}-\mathbf{x}'|^3} d\mathbf{x}'$; in other words, a linear function with a set number of coefficients can only match accurately a function with limited irregularity .

### 2.2.2. Isotropic Pressure

The term $S(\mathbf{x})$ is considered a scalar function where $S_x = S_y = S_z = S$ ; it is therefore natural to represent it using a distribution of Dirac functions. In other words, the source function becomes expression (2.7), where $\delta(\mathbf{x}' - \mathbf{x_j})$ is the Dirac source location $j$ of $m$ positioned at $\mathbf{x'_j}$, $\mathbf{x}'$ is the integration variable from (2.2)

$$S(\mathbf{x}') = \sum_{j=1}^{m} u_j \delta\left(\mathbf{x}' - \mathbf{x_j}\right) \tag{2.7}$$

Injecting this in (2.2), the source condition is rewritten in (2.8). The removal of the integral is obtained simply by considering that the Dirac source is equal to 0 at every point within the integral, except at $\mathbf{x}' - \mathbf{x'_j}$ where it is equal to $\infty$, but its integral on the whole domain is equal to 1 [3]. Note that by imposing (2.7) as the source function $S(\mathbf{x}')$, for $\mathbf{x} = \mathbf{x_j}, \forall j$, $|\mathbf{x} - \mathbf{x_j}|$ will tend to 0 and therefore the left hand side of (2.8) will tend to infinity.

$$\begin{aligned} \int_{D_S} \sum_{j=1}^{m} u_j \delta\left(\mathbf{x}' - \mathbf{x_j}\right) \frac{(\mathbf{x} - \mathbf{x}')}{|\mathbf{x} - \mathbf{x}'|^3} d\mathbf{x}' &= \mathbf{F}(\mathbf{x}) \\ \Leftrightarrow \sum_{j=1}^{m} u_j \frac{(\mathbf{x} - \mathbf{x_j})}{|\mathbf{x} - \mathbf{x_j}|^3} &= \mathbf{F}(\mathbf{x}) \end{aligned} \tag{2.8}$$

Let us write the variation of $P_s$ due to the source function $S(\mathbf{x})$ as $DP_s$. Using in expression (1.13), the 3D Green's solution to equation (2.9), we finally obtain (2.10).

$$\overrightarrow{\nabla}^2 (DP_s) = u_j \cdot \delta(\mathbf{x} - \mathbf{x_j}) \Leftrightarrow DP_s = \frac{u_j}{4\pi|\mathbf{x} - \mathbf{x_j}|} \tag{2.9}$$

$$DP_s(\mathbf{x}) = \sum_{j=1}^{m} \frac{u_j}{4\pi|\mathbf{x} - \mathbf{x_j}|} \tag{2.10}$$

### 2.2.3. Non-Isotropic Pressure

The term $S(\mathbf{x})$ must be a scalar function such that $\nabla^2 P_s$ in expression (1.13) remains a scalar. Writing $\mathbf{S}(\mathbf{x})$ as a vector will have the effect of turning $P_s$ into a vector. Effectively, on $D_f$, $P_s$ would be a vector of equal components, thus conserving a physical sense. In the region $\delta D \cup D_s$ where $\gamma = 0$, the components of $P_s$ can take any value, and hence the pressure is not isotropic. The region $D_s$ is not physical, and so it is not shocking to express $P_s$ as non-isotropic in that region. On $\delta D$ however, the region is physical, and so having a non-isotropic pressure leads to a non-physical problem formulation. The solution obtained with the non-isotropic pressure and the Dirac distribution vector is therefore not adapted to CFD simulation. However, the results that are obtained by writing are interesting, even if the problem is not dimensioned properly. In this case, the Dirac distribution is defined such that a component $S_x(\mathbf{x})$ is the sum over $j$ of the coefficients $u_{xj}$ multiplied by a Dirac delta. In this case, the vector source function $\mathbf{S}(\mathbf{x})$ is given in (2.11). $m$ is the number of source locations.

$$S(\mathbf{x}') = \sum_{j=1}^{m} \mathbf{u}_j \cdot \delta\left(\mathbf{x}' - \mathbf{x_j}\right) \tag{2.11}$$

Injecting this in (2.2), the source condition is rewritten in (2.12). The removal of the integral is obtained simply by considering that for a given direction (or vector component of $\mathbf{F}(\mathbf{x})$), the Dirac delta integral over the domain is equal to 0 at every point within the integral, except at $\mathbf{x}' - \mathbf{x}'_\mathbf{j}$ where it is equal to 1. It is important to realise that the components of the coefficients of the Dirac delta only influence the corresponding components of $\mathbf{F}(\mathbf{x})$. Note that $\odot$ symbolises the term to term product. See [6] for Dirac vector notations.

$$\begin{aligned}
&\int_{D_S} \sum_{j=1}^{m} \mathbf{u}_j \cdot \delta\left(\mathbf{x}' - \mathbf{x_j}\right) \frac{(\mathbf{x} - \mathbf{x}')}{|\mathbf{x} - \mathbf{x}'|^3} d\mathbf{x}' = \mathbf{F}(\mathbf{x}) \\
\Leftrightarrow \quad &\begin{bmatrix} \int_{D_S} \sum_{j=1}^{m} u_{x,j} \delta_x\left(\mathbf{x}' - \mathbf{x_j}\right) \frac{(x - x')}{|\mathbf{x} - \mathbf{x}'|^3} d\mathbf{x}' \\ \int_{D_S} \sum_{j=1}^{m} u_{y,j} \delta_y\left(\mathbf{x}' - \mathbf{x_j}\right) \frac{(y - y')}{|\mathbf{x} - \mathbf{x}'|^3} d\mathbf{x}' \\ \int_{D_S} \sum_{j=1}^{m} u_{z,j} \delta_z\left(\mathbf{x}' - \mathbf{x_j}\right) \frac{(z - z')}{|\mathbf{x} - \mathbf{x}'|^3} d\mathbf{x}' \end{bmatrix} = \begin{bmatrix} F_x(\mathbf{x}) \\ F_y(\mathbf{x}) \\ F_z(\mathbf{x}) \end{bmatrix} \\
\Leftrightarrow \quad &\sum_{j=1}^{n} \mathbf{u}_j \odot \frac{(\mathbf{x} - \mathbf{x_j})}{|\mathbf{x} - \mathbf{x_j}|^3} = \mathbf{F}(\mathbf{x})
\end{aligned} \tag{2.12}$$

Let us write the variation of $\mathbf{P_s}$ due to the source function $\mathbf{S}(\mathbf{x})$ as $\mathbf{DP_s}$. Using the solution to green's equation given in (2.9) in expression (1.13), we finally obtain (2.13). Note that this expression for the pressure is on physical, as the principles of isotropy are not respected. This method therefore cannot be used in standard CFD simulations.

$$\begin{bmatrix} DP_{s,x}(\mathbf{x}) \\ DP_{s,y}(\mathbf{x}) \\ DP_{s,z}(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{m} \frac{u_{x,j}}{4\pi|\mathbf{x}-\mathbf{x_j}|} \\ \sum_{j=1}^{m} \frac{u_{y,j}}{4\pi|\mathbf{x}-\mathbf{x_j}|} \\ \sum_{j=1}^{m} \frac{u_{z,j}}{4\pi|\mathbf{x}-\mathbf{x_j}|} \end{bmatrix} \tag{2.13}$$

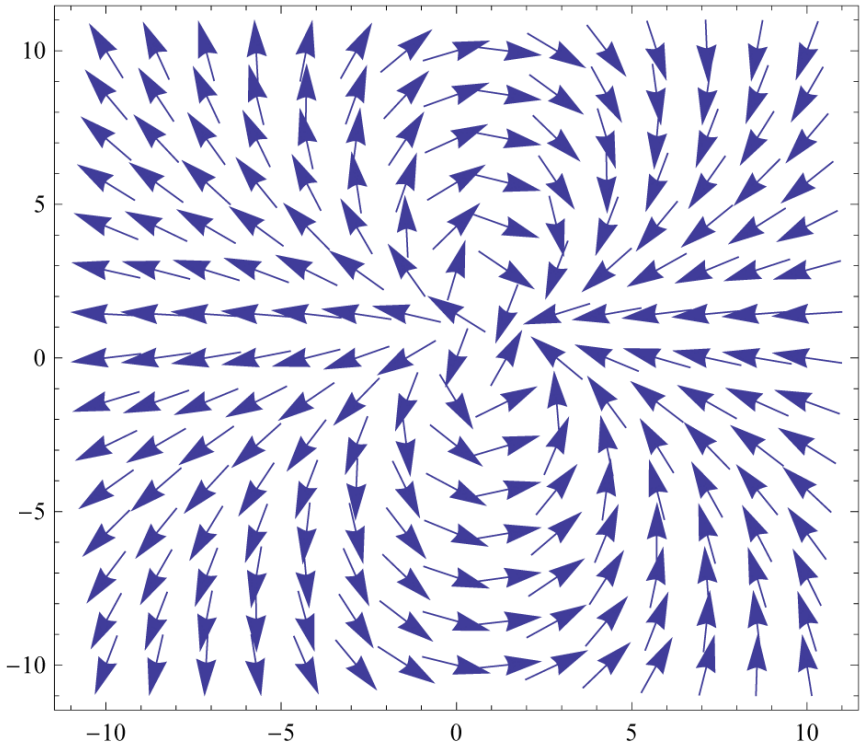
### 2.2.4. Isotropic pressure with Dipole Distribution

Dipole are constructed by placing two Dirac poles of opposite coefficients at infinitely small distance $\epsilon$. If the two Dirac sources are placed on the same absciss, then the source will be radiating primarily along that absciss value. Similarly, if the two Dirac sources have been placed along the same ordinate, then the sources will radiate primarily along that ordinate. This is shown in figure 2.1, where it is clear that a Dipole along x affects primarily the x direction of space, and a Dipole along y the y direction of space. Because Dipoles offer this sort of directionality, they will improve linearisation accuracy by allowing the directions of the problem to be more decoupled than with the basic Dirac source distribution method.

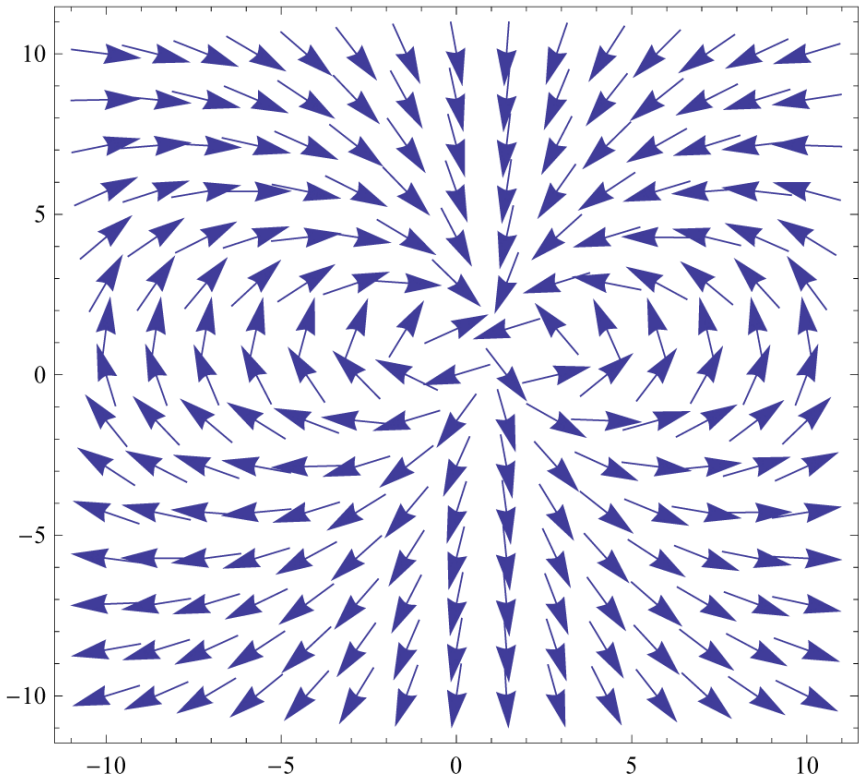
Mathematically, the Dipole distribution in one direction is defined as the derivative of the Dirac distribution along that direction. This means that a Dipole along x is define as $\frac{\partial\delta(\mathbf{x}'-\mathbf{x}'_j)}{\partial y}$. Similarly the Dipole along y is defined as $\frac{\partial\delta(\mathbf{x}'-\mathbf{x_j})}{\partial y}$, the dipole along z is defined as $\frac{\partial\delta(\mathbf{x}'-\mathbf{x_j})}{\partial z}$ The source function $S(\mathbf{x})$ is then written as a scalar as shown in (2.14). The coefficient vectors $\mathbf{u_1}, \mathbf{u_2}, \mathbf{u_3}, \mathbf{u_4}$ correspond to the coefficients of the Dirac monopole, x dipole, y dipole and z dipole respectively. $m$ is the number of source locations. Note that here we have chosen to impose 4 different source types at each source location for sake of completeness. This is not a requirement, and any kind of source can be positioned at any source position; this will however have an impact on the accuracy of the linearisation.

$$\begin{aligned} S(\mathbf{x}) &= \sum_{j=1}^{m} u_{o,j}\delta\left(\mathbf{x}'-\mathbf{x_j}\right) \\ &+ \sum_{j=1}^{m} u_{x,j}\frac{\partial\delta\left(\mathbf{x}'-\mathbf{x_j}\right)}{\partial x} + \sum_{j=1}^{m} u_{y,j}\frac{\partial\delta\left(\mathbf{x}'-\mathbf{x_j}\right)}{\partial y} + \sum_{j=1}^{m} u_{z,j}\frac{\partial\delta\left(\mathbf{x}'-\mathbf{x_j}\right)}{\partial z} \end{aligned} \tag{2.14}$$

Inserting the expression (2.14) into (2.2) gives (2.15).




(a) Dipole along x




(b) Dipole along y

Figure 2.1.: Vector field generated by Dipoles

$$
\begin{aligned}
\int_{D_S} & \sum_{j=1}^{m} \quad u_{o,j}\delta\left(\mathbf{x}'-\mathbf{x_j}\right)\frac{(\mathbf{x}-\mathbf{x}')}{|\mathbf{x}-\mathbf{x}'|^3}d\mathbf{x}' \\
+\int_{D_S} & \sum_{j=1}^{m} \quad u_{x,j}\frac{\partial\delta\left(\mathbf{x}'-\mathbf{x_j}\right)}{\partial x}\frac{(\mathbf{x}-\mathbf{x}')}{|\mathbf{x}-\mathbf{x}'|^3}d\mathbf{x}' \\
+\int_{D_S} & \sum_{j=1}^{m} \quad u_{y,j}\frac{\partial\delta\left(\mathbf{x}'-\mathbf{x_j}\right)}{\partial y}\frac{(\mathbf{x}-\mathbf{x}')}{|\mathbf{x}-\mathbf{x}'|^3}d\mathbf{x}' \\
+\int_{D_S} & \sum_{j=1}^{m} \quad u_{z,j}\frac{\partial\delta\left(\mathbf{x}'-\mathbf{x_j}\right)}{\partial z}\frac{(\mathbf{x}-\mathbf{x}')}{|\mathbf{x}-\mathbf{x}'|^3}d\mathbf{x}' \\
= & \quad \mathbf{F}\left(\mathbf{x}\right)
\end{aligned}
\tag{2.15}
$$

One of the properties of the dipole is given in (2.16). Note that in our case, the domain of integration with reduce to $D_s$ as there are no sources outside the solid. Finally, using (2.16) in (2.15) yields (2.17).

$$
\int_{-\infty}^{\infty}\frac{\partial\delta\left(\mathbf{x}'-\mathbf{x_j}\right)}{\partial x}\cdot\phi\left(\mathbf{x}\right)=-\int_{-\infty}^{\infty}\delta\left(\mathbf{x}'-\mathbf{x_j}\right)\cdot\frac{\partial\phi\left(\mathbf{x}\right)}{\partial x}
\tag{2.16}
$$

$$
\begin{aligned}
\int_{D_S} & \sum_{j=1}^{m} \quad u_{o,j}\delta\left(\mathbf{x}'-\mathbf{x_j}\right)\frac{(\mathbf{x}-\mathbf{x}')}{|\mathbf{x}-\mathbf{x}'|^3}d\mathbf{x}' \\
-\int_{D_S} & \sum_{j=1}^{m} \quad u_{x,j}\delta\left(\mathbf{x}'-\mathbf{x_j}\right)\frac{\partial\left(\frac{(\mathbf{x}-\mathbf{x}')}{|\mathbf{x}-\mathbf{x}'|^3}\right)}{\partial x}d\mathbf{x}' \\
-\int_{D_S} & \sum_{j=1}^{m} \quad u_{y,j}\delta\left(\mathbf{x}'-\mathbf{x_j}\right)\frac{\partial\left(\frac{(\mathbf{x}-\mathbf{x}')}{|\mathbf{x}-\mathbf{x}'|^3}\right)}{\partial y}d\mathbf{x}' \\
-\int_{D_S} & \sum_{j=1}^{m} \quad u_{z,j}\delta\left(\mathbf{x}'-\mathbf{x_j}\right)\frac{\partial\left(\frac{(\mathbf{x}-\mathbf{x}')}{|\mathbf{x}-\mathbf{x}'|^3}\right)}{\partial z}d\mathbf{x}' \\
= & \quad \mathbf{F}\left(\mathbf{x}\right)
\end{aligned}
\tag{2.17}
$$

Finally, using the property of integration of a Dirac delta on (2.17) gives (2.18).

$$
\begin{aligned}
\sum_{j=1}^{m} & \quad u_{o,j} \quad \frac{(\mathbf{x}-\mathbf{x_j})}{|\mathbf{x}-\mathbf{x_j}|^3} \\
- & \quad u_{x,j} \quad \frac{\partial\left(\frac{(\mathbf{x}-\mathbf{x_j})}{|\mathbf{x}-\mathbf{x_j}|^3}\right)}{\partial x}-u_{y,j}\frac{\partial\left(\frac{(\mathbf{x}-\mathbf{x_j})}{|\mathbf{x}-\mathbf{x_j}|^3}\right)}{\partial y}-u_{z,j}\frac{\partial\left(\frac{(\mathbf{x}-\mathbf{x_j})}{|\mathbf{x}-\mathbf{x_j}|^3}\right)}{\partial z} \\
= & \quad \mathbf{F}\left(\mathbf{x}\right)
\end{aligned}
\tag{2.18}
$$

Let us write the variation of $P_s$ due to the source function $S(\mathbf{x})$ as $DP_s$. Using the solution to green's equation given in (2.9) in expression (1.13), we finally obtain (2.19).

$$
\begin{aligned}
DP_s(\mathbf{x}) &= \sum_{j=1}^{m} \frac{u_{o,j}}{4\pi|\mathbf{x}-\mathbf{x_j}|} \\
&+ \sum_{j=1}^{m} \frac{\partial\left(\frac{u_{x,j}}{4\pi|\mathbf{x}-\mathbf{x_j}|}\right)}{dx} + \sum_{j=1}^{m} \frac{\partial\left(\frac{u_{y,j}}{4\pi|\mathbf{x}-\mathbf{x_j}|}\right)}{dy} + \sum_{j=1}^{m} \frac{\partial\left(\frac{u_{z,j}}{4\pi|\mathbf{x}-\mathbf{x_j}|}\right)}{dz}
\end{aligned}
\tag{2.19}
$$

### 2.2.5. Expressions for the 1D case

In the unidimensional case, (2.8) becomes 1 equation only, as shown in (2.20). The Dirac and dipole distribution methods become one and the same.

$$
\sum_{j=1}^{n} u_j \frac{(x-x_j)}{|x-x_j|^3} = F(x) \tag{2.20}
$$

Writing this in matrix form gives (B.2).

### 2.2.6. Expressions for the 3D case

**Isotropic pressure** In the three dimensional case, (2.8) becomes 3 equations, as shown in (2.21).

$$
\begin{aligned}
\sum_{j=1}^{n} u_j \frac{(x-x_j)}{\sqrt{(x-x_j)^2+(y-y_j)^2+(z-z_j)^2}^3} &= F_x(\mathbf{x}) \\
\sum_{j=1}^{n} u_j \frac{(y-y_j)}{\sqrt{(x-x_j)^2+(y-y_j)^2+(z-z_j)^2}^3} &= F_y(\mathbf{x}) \\
\sum_{j=1}^{n} u_j \frac{(z-z_j)}{\sqrt{(x-x_j)^2+(y-y_j)^2+(z-z_j)^2}^3} &= F_z(\mathbf{x})
\end{aligned}
\tag{2.21}
$$

**Non-Isotropic pressure** In the three dimensionbal case, (2.12) becomes 3 equations, as shown in (2.22).

$$
\begin{aligned}
\sum_{j=1}^{n} u_{xj} \frac{(x-x_j)}{\sqrt{(x-x_j)^2+(y-y_j)^2+(z-z_j)^2}^3} &= F_x(\mathbf{x}) \\
\sum_{j=1}^{n} u_{yj} \frac{(y-y_j)}{\sqrt{(x-x_j)^2+(y-y_j)^2+(z-z_j)^2}^3} &= F_y(\mathbf{x}) \\
\sum_{j=1}^{n} u_{zj} \frac{(z-z_j)}{\sqrt{(x-x_j)^2+(y-y_j)^2+(z-z_j)^2}^3} &= F_z(\mathbf{x})
\end{aligned}
\tag{2.22}
$$

**Dipole Ditribution** In the three dimensional case, (2.8) becomes 3 equations, as shown in (2.23), where $|\mathbf{x}-\mathbf{x'}_j|^3 = \sqrt{\left(x-x'_j\right)^2+\left(y-y'_j\right)^2+\left(z-z'_j\right)^2}^3$.

$$\begin{aligned}
&\sum_{j=1}^{n} u_{o,j}\frac{(x-x_j)}{|\mathbf{x}-\mathbf{x}_j|^3} - u_{x,j}\frac{\partial\left(\frac{(x-x_j)}{|\mathbf{x}-\mathbf{x}_j|^3}\right)}{\partial x} - u_{y,j}\frac{\partial\left(\frac{(x-x_j)}{|\mathbf{x}-\mathbf{x}_j|^3}\right)}{\partial y} - u_{z,j}\frac{\partial\left(\frac{(x-x_j)}{|\mathbf{x}-\mathbf{x}_j|^3}\right)}{\partial z} = F_x(\mathbf{x})\\
&\sum_{j=1}^{n} u_{o,j}\frac{(y-y_j)}{|\mathbf{x}-\mathbf{x}_j|^3} - u_{x,j}\frac{\partial\left(\frac{(y-y_j)}{|\mathbf{x}-\mathbf{x}_j|^3}\right)}{\partial x} - u_{y,j}\frac{\partial\left(\frac{(y-y_j)}{|\mathbf{x}-\mathbf{x}_j|^3}\right)}{\partial y} - u_{z,j}\frac{\partial\left(\frac{(y-y_j)}{|\mathbf{x}-\mathbf{x}_j|^3}\right)}{\partial z} = F_y(\mathbf{x})\\
&\sum_{j=1}^{n} u_{o,j}\frac{(z-z_j)}{|\mathbf{x}-\mathbf{x}_j|^3} - u_{x,j}\frac{\partial\left(\frac{(z-z_j)}{|\mathbf{x}-\mathbf{x}_j|^3}\right)}{\partial x} - u_{y,j}\frac{\partial\left(\frac{(z-z_j)}{|\mathbf{x}-\mathbf{x}_j|^3}\right)}{\partial y} - u_{z,j}\frac{\partial\left(\frac{(z-z_j)}{|\mathbf{x}-\mathbf{x}_j|^3}\right)}{\partial z} = F_z(\mathbf{x})
\end{aligned} \tag{2.23}$$

### 2.2.7. Computational Formulation

See B for the computational formulation of the distribution method.

## 2.3. Finite Element approach

### 2.3.1. General Theory

Using the theory of finite elements, the source function $S(\mathbf{x})$ can be linearised as the sum of simple polynomial shape functions and their coefficients. Other functions that polynomials can be chosen, as long as they are continuous within the element. Thus, $S(\mathbf{x})$ is written as shown in (2.24), where $u_i$ represents the coefficient of the function $e_i$. This method will only be explained in the 1D case, as higher dimensions will require increasing order shape functions.

$$S(\mathbf{x}') = \sum_{i=1}^{n} u_i e_i(\mathbf{x}') \tag{2.24}$$

Injecting the new linearised formulation for $S(\mathbf{x})$ into equation (2.2) we obtain expression (2.25). The fact that $u_i$ does not depend on $x'$, and the commutativity between the sum and the integral allows us to rewrite the expression with all the unknowns $u_i$ outside of the integral. Note that for $\mathbf{x}=\mathbf{x}'$, the function is not defined. It is therefore imperative that $e_i(\mathbf{x}) = 0$ on $D_f$ and $\delta D$, or in other words the chosen elements must be contained strictly within the solid.

$$\begin{aligned}
&\int_{D_S}\sum_{i=1}^{n} u_i e_i(\mathbf{x}')\frac{(\mathbf{x}-\mathbf{x}')}{|\mathbf{x}-\mathbf{x}'|^3}d\mathbf{x}' = \mathbf{F}(\mathbf{x})\\
\Leftrightarrow &\sum_{i=1}^{n} u_i\int_{D_S} e_i(\mathbf{x}')\frac{(\mathbf{x}-\mathbf{x}')}{|\mathbf{x}-\mathbf{x}'|^3}d\mathbf{x}' = \mathbf{F}(\mathbf{x})
\end{aligned} \tag{2.25}$$

### 2.3.2. One Dimension Fomulation

In the 1D case, the right hand side reduces to the scalar $F(x)$. Furthermore, the domain of integration becomes a simple line and the shape functions can therefore be first order polynomials. For a straight line, the boundary of a solid inside the domain reduces to 2 points where (2.2) must be satisfied. Thus, the problem has 2 degrees of freedom, and so in expression (2.25), $n = 2$. This is shown in (2.26).

$$\sum_{i=1}^{2} u_i \int_{x_1}^{x_2} e_i\left(x'\right) \frac{(x - x')}{|x - x'|^3} dx' = F(x) \tag{2.26}$$

Using only 2 node linear elements, the solid can therefore be represented as a single element. As stated in 2.24, $S(\mathbf{x})$, is defined along the element by the sum of two first order functions $e_1$ and $e_2$. These are chosen such that $e_1 = 1$ at node 1 of the element and 0 at node 2 of the element, $e_2 = 1$ at node 2 of the element and 0 at node 1 of the element , and $e_1 = e_2 = 0$ outside of the element.

In this case, the chosen functions are given in (2.27). Thus, the formulation for $S(x)$ becomes : $S(x_1) = u_1$ and $S(x_2) = u_2$

$$\begin{aligned} e_1\left(x'\right) &= \frac{x_2 - x'}{x_2 - x_1} \\ e_2\left(x'\right) &= \frac{x' - x_1}{x_2 - x_1} \end{aligned} \tag{2.27}$$

Using (2.27), expression (2.25) can be rewritten in matrix form $\mathbf{Au} = \mathbf{F}$ as shown in (2.28), where $x_1 \leq x_1' \leq x_2' \leq x_2$ to avoid singularities . From there, the solution is simply obtained as $\mathbf{u} = \mathbf{F}/\mathbf{A}$. The source function $S(\mathbf{x})$ is then recovered from (2.24).

$$\begin{bmatrix} \int_{x_1'}^{x_2'} \frac{x_2 - x'}{x_2 - x_1} \frac{x_1 - x'}{|x_1 - x'|^3} dx' & \int_{x_1}^{x_2} \frac{x' - x_1}{x_2 - x_1} \frac{x_1 - x'}{|x_1 - x'|^3} dx' \\ \int_{x_1'}^{x_2'} \frac{x_2 - x'}{x_2 - x_1} \frac{x_2 - x'}{|x_2 - x'|^3} dx' & \int_{x_1}^{x_2} \frac{x' - x_1}{x_2 - x_1} \frac{x_2 - x'}{|x_2 - x'|^3} dx' \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} F(x_1) \\ F(x_2) \end{bmatrix} \tag{2.28}$$

# 3. Numerical Implementation

## 3.1. Program Architecture

The program for finding $S(\mathbf{x})$ has been written in matlab language for simplicity. For details, see C.1.

## 3.2. Proof Of Concept

The proof of concept will be shown in 1D using the Dirac Distribution and FE methods, and in 2D using the Dirac distribution for isotropic or non isotropic pressure. All tests will be run using Matlab 2011 on a Core i5 processor with 8Gb ram.

### 3.2.1. 1D proof of concept

In 1D, the solid is limited to a straight line. In our case, a domain defined by $(-5, 5)$ will be used, with a solid line stretching from $x = -2$ to $x = 2$. The values of $F(x)$ are given at random as $F(-2) = 0.5$ and $F(-2) = 0.5$.

**Finite Element approach** Note that the integration step is set as $\delta x/10$. The plot for $F(x)$ and $A * U$ is shown in figure C.3. Structured grid space step was automatically calculated as $\delta x = 2$, such that only 2 nodes were available within the solid at $x = -1$ and $x = 1$ .The time required for the execution of the code was 0.64 CPU seconds. The absolute error obtained at the left boundary point was $5.5510^{-17}$ (well below computational accuracy). The error obtained at the right boundary point was 0. The proof of concept of the 1D Finite Element method has been validated.

**Isotropic pressure Dirac distribution method** The plot for $F(x)$ and $A * U$ is shown in figure C.4. Structured grid space step was automatically calculated as $\delta x = 2$, such that only 2 nodes were available within the solid at $x = -1$ and $x = 1$ .The time required for the execution of the code was 0.50 CPU seconds. Thus, for such a simple case, the difference in computational time is 14%. This is due to the fact that the FE method is required to compute integrals. This difference in time would become consequent in more complex problems. The absolute error obtained at the left boundary point was $5.010^{-17}$ (well below computational accuracy). The error obtained the right boundary point was 0. The proof of concept of the 1D isotropic pressure Dirac distribution method has been validated.

### 3.2.2. 2D proof of concept

The domain is defined as a square with x and y values within $(-5, 5)$.The shape selected for the 2D proof of concept is a cylinder centred at $(0, 0)$ and with a radius of $r = 4$. The discretised solid boundary is defined by 100 points. The values for $\mathbf{F}(\mathbf{x})$ are defined as follows : $F_x(\mathbf{x}) = sin(\theta)$ and $F_y(\mathbf{x}) = cos(\theta)$ where $\theta$ is the trigonometric angle, as shown in figure 3.1. Note that the function is continuous around the cylindre, in other words, there are no sharp gradient between the values $\mathbf{F}(\mathbf{x})$ of two consecutive points. Furthermore, $F_x(\mathbf{x}) \neq F_y(\mathbf{x})$ except at $\theta = \pi/4 + n\pi/2$.

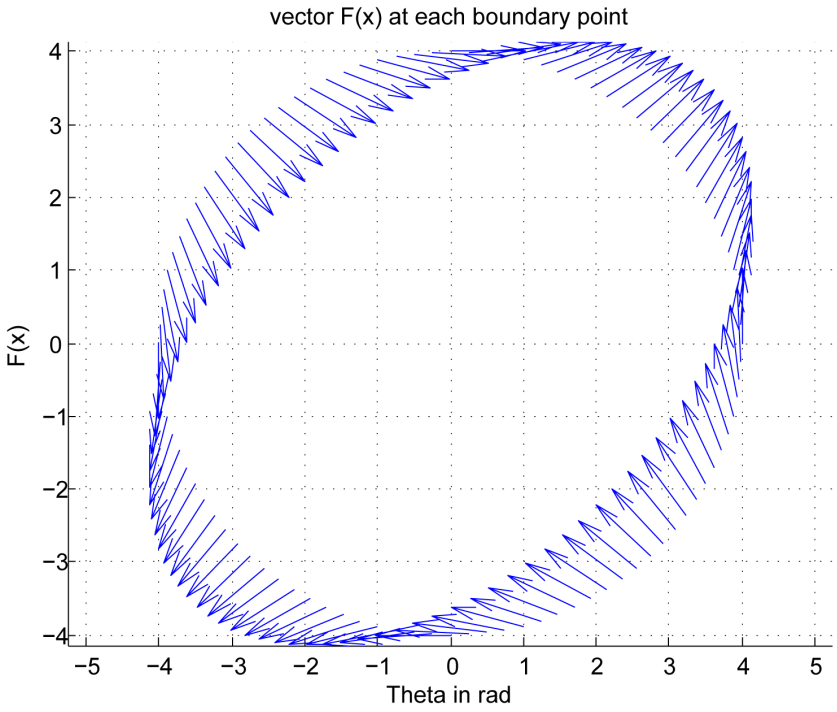
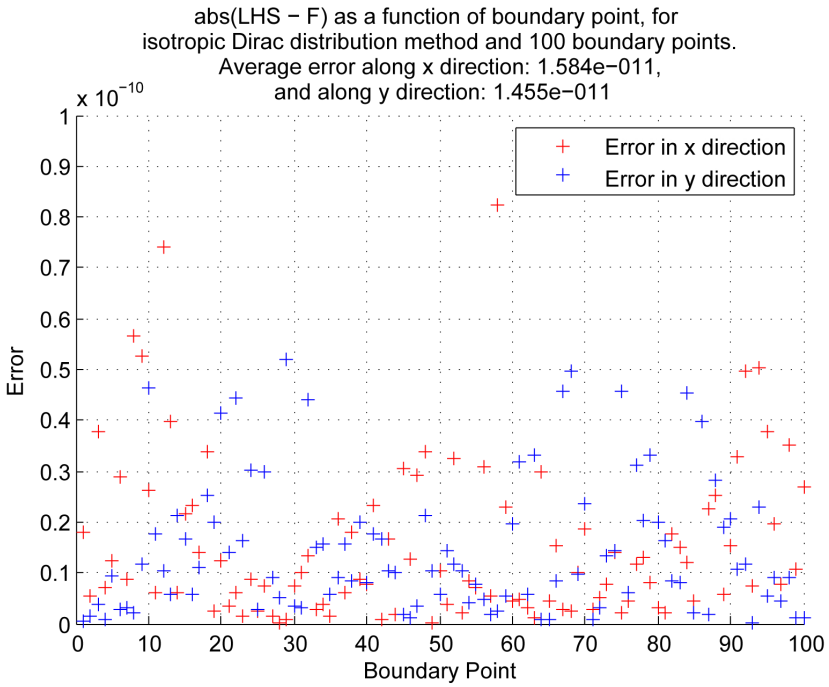
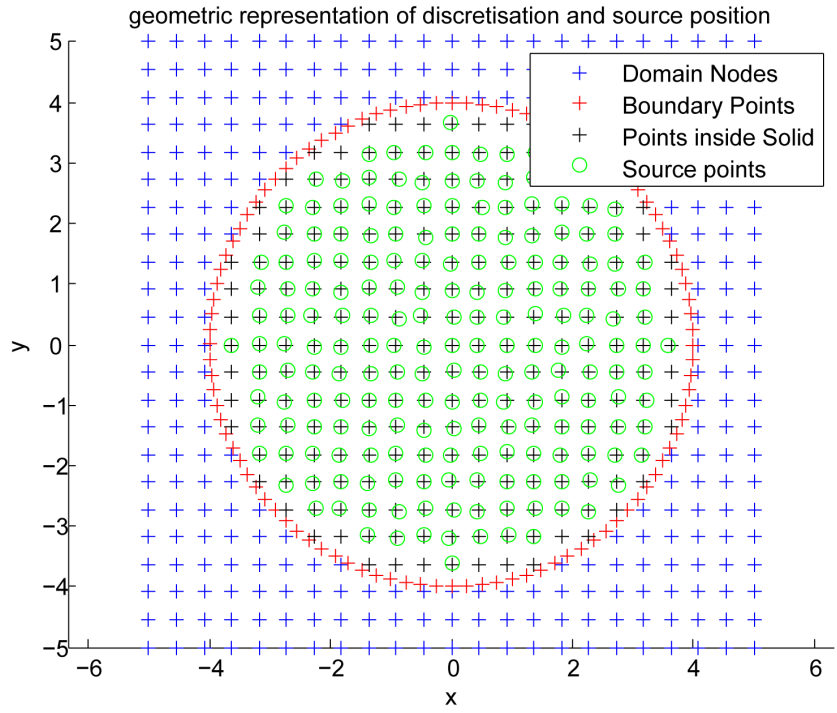



Figure 3.1.: Value of $\mathbf{F}(\mathbf{x})$ as a function of $\theta$ for 2D POC

**Isotropic pressure Dirac distribution method** With a isotropic pressure Dirac distribution method, 200 sources are required, and therefore at least 200 nodes must be available within the solid. The plot for the error $|F - A * U|$ and the geometric information are shown in 3.2. The computation took 1.33 CPU seconds to run, and returned the average errors in the x and y direction shown in figure 3.2. Note that the condition number of the matrix is of $3.92 * 10^{17}$ and the inversion of the matrix returns a a close to singularity warning. This is because of the high number of source nodes which contribute to bigger matrices to be inverted, and because the large size of the matrix due to the coupling of the x and y directions. Note that the random function used in the code means that each run will yield different errors; the variation in error should however remain small.




(a) Error $|F - A * U|$ as a function of boundary point




(b) Geometric information

Figure 3.2.: 2D isotropic pressure Dirac distribution proof of concept results

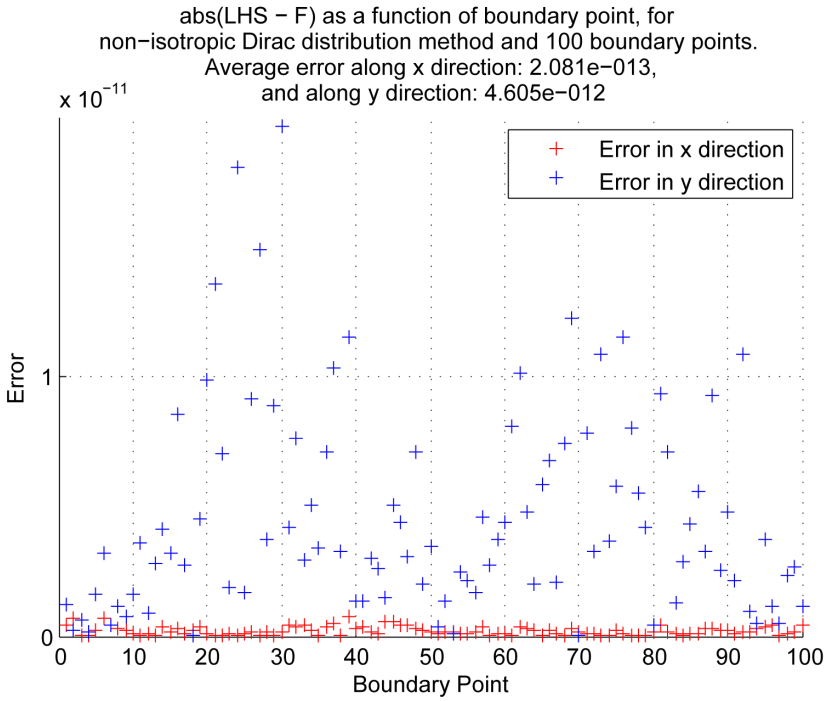
**Non-isotropic pressure Dirac distribution method** For the non-isotropic pressure Dirac distribution method, 100 sources are required, and therefore at least 100 nodes must be available within the solid. The plot for the error $|F - A * U|$ and the geometric information are shown in 3.3. The computation took 0.94 CPU seconds to run, and average errors are shown on 3.3. Note that the condition numbers of the matrices are of $2.70 * 10^{13}$ for the x direction, and $3.74 * 10^{14}$ for the y direction. The impact of the more ill conditioned y direction matrix is directly felt in the y direction error, which is approximately 1 order of magnitude larger than the x direction error. Differences in conditioning of x and y direction are due to random positioning of sources; in other words, the source pattern is more symmetric with respect to the boundary for the y axis symmetry.

Because there are less sources here than with the isotropic pressure Dirac distribution method, and because the sources are very directive and allow the decoupling of the directions of the problem, the condition number is closer to 1 and the matrix is better conditioned. Matrix size and decoupling lead to an error which is on average 1.5 orders of magnitude smaller than with the isotropic pressure Dirac distribution method in this test case. Note that the random function used in the code means that each run will yield different errors; the variation in error should however remain small.

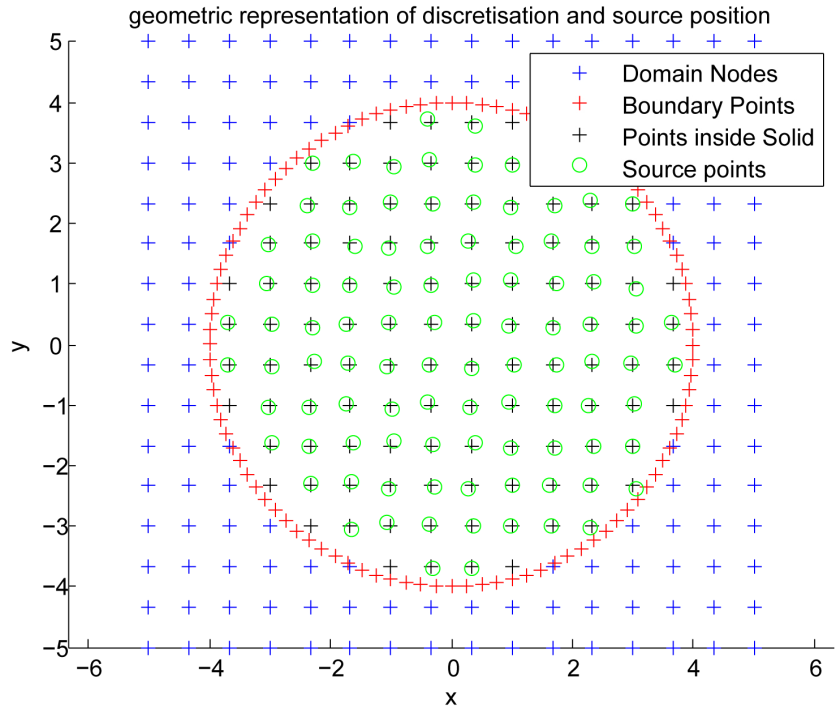
**Isotropic pressure Dipole distribution method** For the Dipole distribution method, $Int\,(200/3) + 1$ source locations are required as three independent source types are positioned at each source location. Therefore, at least 67 nodes must be available within the solid, the last of which will only bear two independent source types . The plot for the error $|F - A * U|$ and the geometric information are shown in figure 3.4. The computation took 1.35 CPU seconds to run, and the average error is shown in figure 3.4. Note that the condition number of the matrix is of $1.07 * 10^{18}$ and Matlab returns a near singular matrix error. The condition number is similar to the isotropic Dirac distribution method because of the large matrix size. However, the Dipoles create a nearly decoupled problem, and so the error is close to the one obtained with the non-isotropic vector Dirac method. Furthermore, the wider variety of sources allows a more accurate linear combination of the function $\mathbf{F}\,(\mathbf{x})$ in this specific case. Note that the random function used in the code means that each run will yield different errors; the variation in error should however remain small.

### 3.2.3. Conclusions

The isotropic pressure Dirac distribution method is dimensionally correct for use in CFD computations, but ensues important computational times and large errors due to bad matrix conditioning and inaccurate linear combination. The non-isotropic pressure Dirac distribution method has good results, with a lower calculation time, and higher accuracy. This is due to the decoupling of the dimensions of the problem, and smaller matrix sizes. However, this method introduces a dimensioning problem when using it within CFD computations. The Dipole distribution method is a compromise between the two previous methods. The near-decoupling of the dimensions of the problem, and the wide variety of source types, gives a low error. However, because of a large

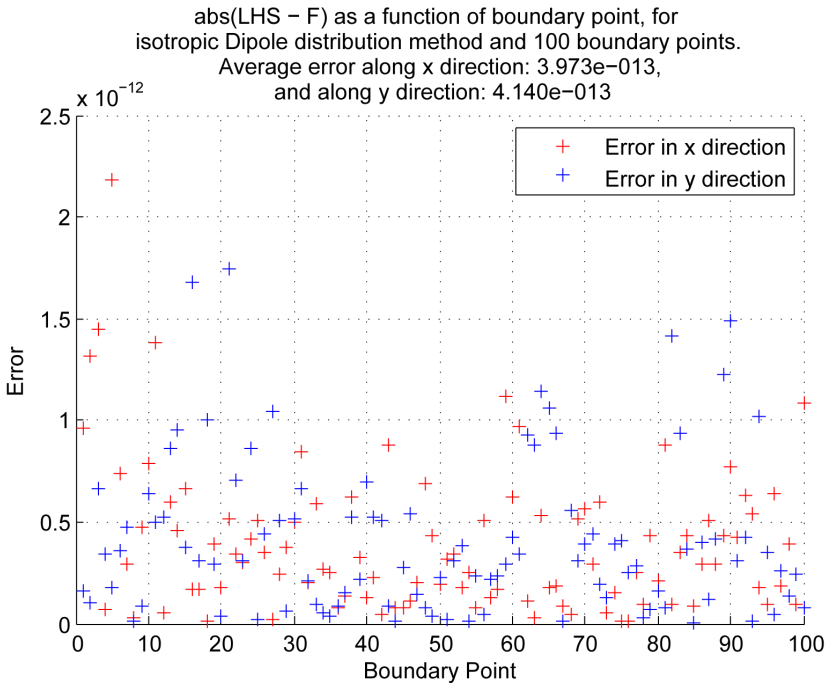



(a) Error $|F - A * U|$ as a function of boundary point

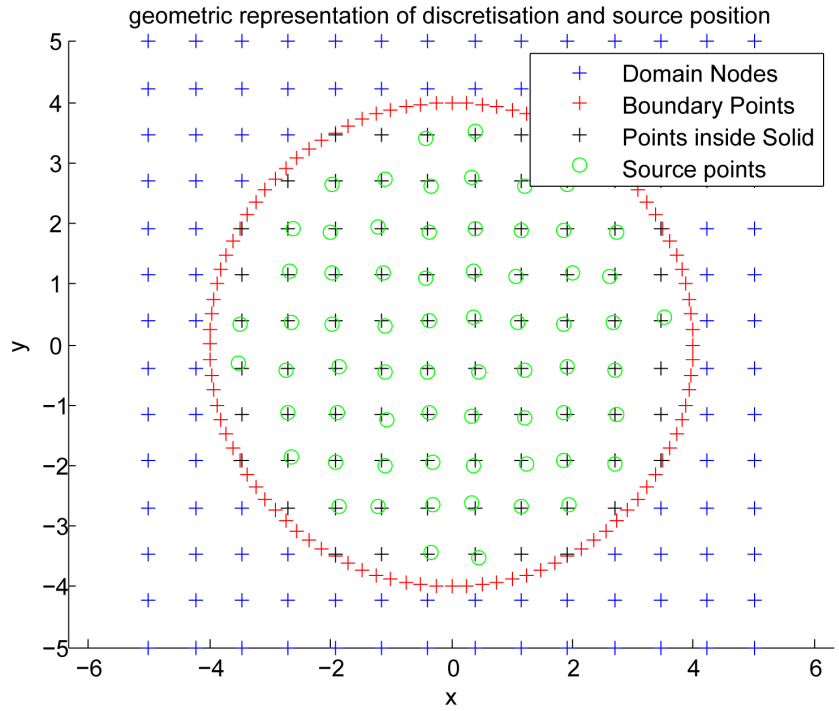



(b) Geometric information

Figure 3.3.: 2D Non-isotropic pressure Dirac distribution method proof of concept results




(a) Error $|F - A * U|$ as a function of boundary point




(b) Geometric information

Figure 3.4.: 2D isotropic pressure Dipole distribution method proof of concept results

matrix size, the condition number of the matrix and computational time are high. This method is dimensionally correct for use in CFD computations and is recommended.

For more comparisons between the different linear combination methods and test cases, please see appendix D.

# 4. Conclusion

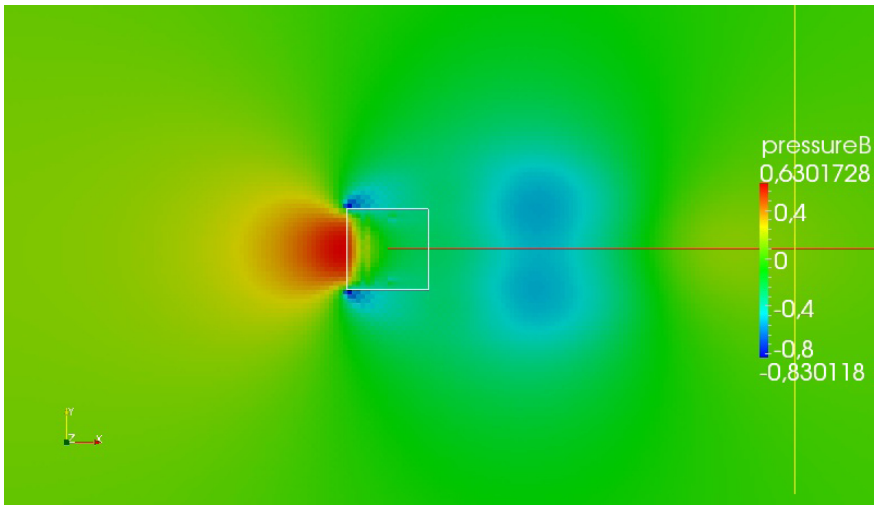
IBM methods up until now hod no explicit balancing of viscous forces and pressure gradient at the boundary of the solids, and were limited to a formal 2nd order accuracy. The imposition of a new surrogate field $S(\mathbf{x})$ corrects this. $S(\mathbf{x})$ is determined through linear combinations of elementary basis distributions. As a rule of thumb, it is recommended to use the isotropic pressure dipole distribution method for CFD computations, as it offers the good accuracy of linear combination, and is dimensionally adapted to CFD. The accuracy of this linear combination of $S(\mathbf{x})$ is however highly dependant on the irregularity of the $F(x)$ function generated by the velocity and pressure fields obtained through the CFD simulation. Furthermore, the calculation time can become very important when using very fine discretisations on the solid boundary.

The attempt at implementing in a CFD code the explicit balancing of the pressure gradient and viscous forces on the boundary gave mixed result. With real flow data used for $\mathbf{F}(\mathbf{x})$, the linear combination coefficients of the source function $S(\mathbf{x})$ were identified accurately. However, the pressure values thus obtained within the solid proved to be of order of magnitude $10^{12}$ approximatly. Although this is not problematic analytically, as the inside of the solid is not physical, this prevented the CFD computations from converging. A minimisation should therefore be imposed on the magnitude of the obtained pressure variations, and perhaps on their derivatives with respect to space.

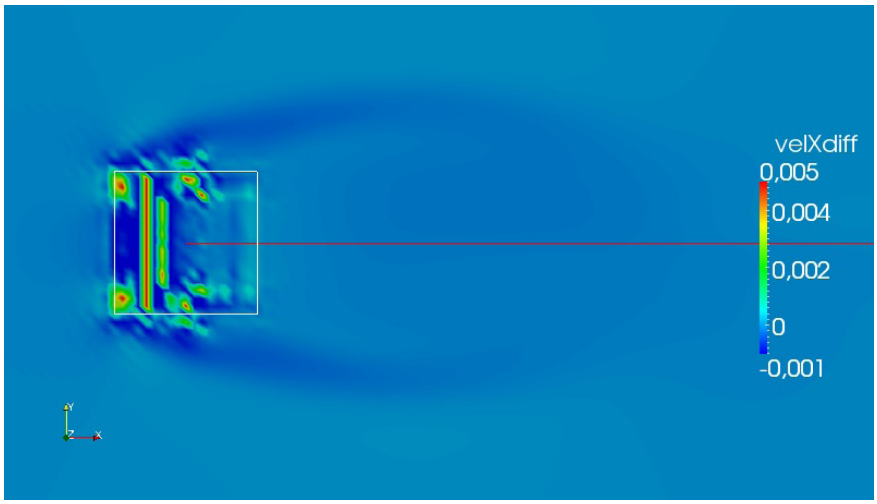
The methods described here attempt to find $\mathbf{u}$ by solving $\mathbf{Au} = \mathbf{F}$ exactly. Because this is not always possible, this leads to matrices with very high condition numbers. Furthermore, no condition is set on the amplitudes of $P_s$. This is problematic as high amplitudes of $P_s$ might be difficult to manage for a CFD solver. To avoid all this, the problem should have been written as a minimisation of the error $\epsilon$ defined as the sum of the squares of the absolute error, or if written in matrix form, as $\epsilon = (\mathbf{Au} - \mathbf{F})^T (\mathbf{Au} - \mathbf{F})$. As the sum of the squared amplitudes of the vector $P_s$ must also be minimised, then we obtain $\epsilon = (\mathbf{Au} - \mathbf{F})^T (\mathbf{Au} - \mathbf{F}) + \alpha \left(\mathbf{Bu}^T\right) (\mathbf{Bu})$ where $\alpha$ is the importance given to keeping small amplitudes for $P_s$, and $\mathbf{B}$ is the pressure construction matrix which can be obtained from (2.19), (2.10), or (2.13). The error is minimal when $\nabla\epsilon = 0 \Leftrightarrow 2\mathbf{A}^t (\mathbf{Au} - \mathbf{F}) + 2\alpha \mathbf{B}^T \mathbf{Bu} = 0$. From there, the solution is obtained as $\mathbf{u} = \left[\mathbf{A}^T\mathbf{A} + \alpha\mathbf{B}^T\mathbf{B}\right]^{-1} \mathbf{A}^T\mathbf{F}$. Let us define $m$ and $n$ the number of terms in $\mathbf{u}$ and $\mathbf{F}$ respectively. If $m = n$ and $\alpha = 0$, then we are back to the methods presented in this report. However, if $m > n$, then $\alpha$ can take another value than 0, and the amplitudes of $P_s$ will be minimised, thus becoming adequate for CFD computations.

This minimisation techniques was attempted in a flow around a square for $Re = 150$ and $\delta t = 0.005$ with the CFD code Incompact3D. This was done with the help of Alexandre Iliopoulos. The results after 2000 iterations are given in figure 4.1. The small variations in pressure and

velocity in the domain conform with what was expected. Indeed, the variation of pressure imposed should only affect the flow sufficiently to allow an increase in the formal order of accuracy of the results. The results are very promising, and the new order of accuracy obtained needs to be calculated to determine the impact of balancing the pressure gradient and the viscous forces at the boundary of the solid in IBM methods.




(a) Pressure field




(b) Variation in x vecolicity field between results from the CFD simulation with or without pressure gradient and viscous forces equilibrium

Figure 4.1.: Results from the implementation of pressure gradient and viscous force balancing at the boundary, with minimisation of the amplitudes of the imposed variations of $P$

# Bibliography


[1] Otvio Bueno. Dirac and the dispensability of mathematics. *Studies In History and Philosophy of Science Part B: Studies In History and Philosophy of Modern Physics*, 36(3):465 – 490, 2005.

[2] Charles R. Doering; J. D. Gibbon. *Applied Analysis of the Navier-Stokes Equations*. Cambridge, 2004.

[3] R. F. Hoskins. *Delta Functions: An Introduction to Generalised Functions*. Horwood Pub Ltd, 2009.

[4] R. Mittal; G. Iaccarino. Immersed boundary methods. *Annual Revision of Fluid Mechanics*, 2005.

[5] S. Laizet; E. Lamballais. High-order compact schemes for incompressible flows: A simple and efficient method with quasi-spectral accuracy. *Journal of Computational Physics*, 2009.

[6] Laurent Schwartz. *Méthodes mathmatiques pour les sciences physiques*. Hermann, 1997.

[7] S. Laizet; J.C. Vassilicos. Multiscale generation of turbulence. *Journal of Multiscale Modelling*, 2011.

# Appendix

# A. Unicity of the Navier Stokes Equations

Determining the conditions under which $\mathbf{v} = \mathbf{u}, \forall t$ is equivalent to determining the conditions under which the unicity of the Navier Stokes equations is verified. Consider that $\mathbf{u}$ is the solution to the standard form of NS given in (1.3) and (1.4), and that $\mathbf{v}$ is a solution to the equations (1.7) and (1.5). Both solutions have the same initial condition $\mathbf{v}(\mathbf{x}, 0) = \mathbf{u}(\mathbf{x}, 0)$ on $D_f$ and $\mathbf{v}(\mathbf{x}, 0) = \mathbf{u}(\mathbf{x}, 0) = 0$ on $\delta D$. Also, note that for this problem $\mathbf{u}(\mathbf{x}, t) = 0, \forall t$ on $\delta_D$, and $\frac{\partial \mathbf{v}}{\partial t} = \frac{\partial \mathbf{u}}{\partial t} = 0, \forall t$ on $\delta_D \cup D_f$ as this is steady flow.

Using the method explained in [2] and used in [7], the difference between the 2 solutions is written as $\mathbf{w}(\mathbf{x}, t) = \mathbf{v}(\mathbf{x}, t) - \mathbf{u}(\mathbf{x}, t)$. The objective is to show that, $\forall \mathbf{x}, \forall t$ on $D_f \cup \delta D$, $\mathbf{w}(\mathbf{x}, t) = 0$. Writing the total kinetic energy $K(t)$ for $\mathbf{w}$ gives expression (A.1). Note that (A.1), being an energy, is strictly greater or equal to 0. So, by replacing $\mathbf{u}$ and $\mathbf{v}$ in (A.1), and finding a form (A.1) where $K(t) \leq 0$, then we shall have $K(t) = 0 \Leftrightarrow \mathbf{w}(\mathbf{x}, t) = 0$.

$$K(t) = \int_{D_f} |\mathbf{w}(\mathbf{x}, t)|^2 d\mathbf{x} \tag{A.1}$$

## A.1. Kinetic energy $K(t)$

To begin, lets express the NS equation of momentum (1.4) for $\mathbf{w}(\mathbf{x}, t)$ by substracting the NS momentum equation for $\mathbf{u}$ from the one for $\mathbf{v}$. Note that from the non linearity of the NS momentum equation: $\left(\mathbf{v} \cdot \overrightarrow{\nabla}\right)\mathbf{v} - \left(\mathbf{u} \cdot \overrightarrow{\nabla}\right)\mathbf{u} = \left((\mathbf{v} - \mathbf{u}) \cdot \overrightarrow{\nabla}\right)(\mathbf{v} - \mathbf{u}) + \left((\mathbf{v} - \mathbf{u}) \cdot \overrightarrow{\nabla}\right)\mathbf{u} + \left(\mathbf{u} \cdot \overrightarrow{\nabla}\right)(\mathbf{v} - \mathbf{u})$. Using this, expression (A.2) is obtained.

$$\begin{aligned} \frac{\partial \mathbf{w}}{\partial t} = & - \left(\left(\mathbf{w} \cdot \overrightarrow{\nabla}\right)\mathbf{w} + \left(\mathbf{w} \cdot \overrightarrow{\nabla}\right)\mathbf{u} + \left(\mathbf{u} \cdot \overrightarrow{\nabla}\right)\mathbf{w}\right) \\ & - \overrightarrow{\nabla} P_w + \nu \nabla^2 \mathbf{w} \end{aligned} \tag{A.2}$$

From there, the kinetic energy expression for $K(t)$ is obtained by dot multiplying $\mathbf{w}$ with (A.2) and then integrating over the fluid domain $D_f$. For sake of clarity, each of the terms from above will be calculated in an independent paragraph.

**First Term** Lets begin by looking at the first term (A.3). Using the fact that $\frac{1}{2}\overrightarrow{\nabla} \cdot \left(\mathbf{w}|\mathbf{w}^2|\right) = \mathbf{w} \cdot \left(\mathbf{w} \cdot \overrightarrow{\nabla}\right)\mathbf{w} + \overrightarrow{\nabla} \cdot \mathbf{w}$ from the rules of differentiation of a product, and that $\overrightarrow{\nabla} \cdot \mathbf{w} = 0$ in $D_f \cup \delta D$ for incompressibility, the first nabla group in the bracket can be transformed. Using a similar idea, $\overrightarrow{\nabla} \cdot (\mathbf{w}|\mathbf{w}\mathbf{u}|) = \mathbf{w} \cdot \left(\mathbf{u} \cdot \overrightarrow{\nabla}\right)\mathbf{w} + \overrightarrow{\nabla} \cdot \mathbf{w}$ from the rules of differentiation of a product.

As $\overrightarrow{\nabla} \cdot \mathbf{w} = 0$ in $D_f \cup \delta D$ because of incompressibility, the third nabla group in the bracket can be transformed. Thus we obtain expression (A.4) for the first term (A.3).

$$\begin{aligned} \int_{D_f} \quad & \mathbf{w} \quad \cdot \left( \left( \mathbf{w} \cdot \overrightarrow{\nabla} \right) \mathbf{w} + \left( \mathbf{w} \cdot \overrightarrow{\nabla} \right) \mathbf{u} + \left( \mathbf{u} \cdot \overrightarrow{\nabla} \right) \mathbf{w} \right) d\mathbf{x} \qquad \text{(A.3)} \\ & \equiv \quad \frac{1}{2} \int_{D_f} \overrightarrow{\nabla} \cdot \left( \mathbf{w} |\mathbf{w}^2| \right) d\mathbf{x} \\ & + \quad \int_{D_f} \mathbf{w} \cdot \left( \mathbf{w} \cdot \overrightarrow{\nabla} \right) \mathbf{u} d\mathbf{x} \\ & + \quad \int_{D_f} \overrightarrow{\nabla} \cdot \left( \mathbf{w} |\mathbf{w}\mathbf{u}| \right) d\mathbf{x} \qquad \text{(A.4)} \end{aligned}$$

**Second Term** Lets turn our attention to the term $\int_{D_f} \mathbf{w} \cdot \overrightarrow{\nabla} P_w d\mathbf{x}$. Notice that, by using the methods for the differentiation of a product, one can write that $\overrightarrow{\nabla} \cdot (P_w \mathbf{w}) = \left( \mathbf{w} \cdot \overrightarrow{\nabla} \right) P_w + P_w \overrightarrow{\nabla} \cdot \mathbf{w}$. By using the NS equation for the conservation of mass $\overrightarrow{\nabla} \cdot \mathbf{w} = 0$. So finally expression (A.5) is obtained.

$$\int_{D_f} \mathbf{w} \cdot \overrightarrow{\nabla} P_w d\mathbf{x} = \int_{D_f} \overrightarrow{\nabla} \cdot (P_w \mathbf{w}) \, d\mathbf{x} \qquad \text{(A.5)}$$

**Third Term** We are now looking at $\int_{D_f} \nu \mathbf{w} \cdot \overrightarrow{\nabla^2} \mathbf{w} d\mathbf{x}$. The objective here is to get rid of the Laplacien. Using the fact that $\frac{\partial}{\partial x_j} \left( \mathbf{w} \frac{\partial}{\partial x_j} \mathbf{w} \right) = \mathbf{w} \cdot \overrightarrow{\nabla^2} \mathbf{w} + \left( \overrightarrow{\nabla} \mathbf{w} \right)^2$, we obtain expression (A.6).

$$\int_{D_f} \nu \mathbf{w} \cdot \overrightarrow{\nabla^2} \mathbf{w} d\mathbf{x} = \nu \left( \int_{D_f} \frac{\partial}{\partial x_j} \left( \mathbf{w} \frac{\partial}{\partial x_j} \mathbf{w} \right) d\mathbf{x} - \int_{D_f} \left( \overrightarrow{\nabla} \mathbf{w} \right)^2 d\mathbf{x} \right) \qquad \text{(A.6)}$$

**Left Hand Side** At the moment, the obtained left hand side of (A.2) is $\int_{D_f} \mathbf{w} \cdot \frac{\partial \mathbf{w}}{\partial t} d\mathbf{x}$. By using the identity $\frac{\partial \mathbf{w}^2}{\partial t} = 2\mathbf{w} \frac{\partial \mathbf{w}}{\partial t}$, expression (A.7) is immediately obtained.

$$\int_{D_f} \mathbf{w} \cdot \frac{\partial \mathbf{w}}{\partial t} d\mathbf{x} = \frac{1}{2} \frac{d}{dt} \int_{D_f} |\mathbf{w}^2| d\mathbf{x} \qquad \text{(A.7)}$$

**Final expression for $K(t)$** Using expressions (A.3) through (A.7), and injecting them into (A.2) gives expression (A.8).

$$\begin{aligned}\frac{d}{dt}K(t) = &- \int_{D_f} \overrightarrow{\nabla}\cdot\left(\mathbf{w}|\mathbf{w}^2|\right)d\mathbf{x} - \int_{D_f}\mathbf{w}\cdot\left(\mathbf{w}\cdot\overrightarrow{\nabla}\right)\mathbf{u}d\mathbf{x} - \int_{D_f}\overrightarrow{\nabla}\cdot\left(\mathbf{w}|\mathbf{w}\mathbf{u}|\right)d\mathbf{x}\\ &- \int_{D_f}\overrightarrow{\nabla}\cdot\left(P_w\mathbf{w}\right)d\mathbf{x}\\ &+ \nu\left(\int_{D_f}\frac{\partial}{\partial x_j}\left(\mathbf{w}\frac{\partial}{\partial x_j}\mathbf{w}\right)d\mathbf{x} - \int_{D_f}\left(\overrightarrow{\nabla}\mathbf{w}\right)^2 d\mathbf{x}\right)\end{aligned} \tag{A.8}$$

**Using Boundary Conditions** Using Green's theorem ($\iint_D \overrightarrow{\nabla}\cdot\mathbf{F}d\mathbf{A} = \oint_c \mathbf{F}\cdot\mathbf{n}ds$) , the pressure term on the RHS of (A.8) gives expression (A.9), the first integral of the viscosity term gives expression (A.10), the first term of the RHS gives (A.11) and the third term of the RHS gives (A.12).

$$\oint_{\delta D} \left(P_w\mathbf{w}\right)\cdot\mathbf{n}ds \tag{A.9}$$

$$\oint_{\delta D} \left(\mathbf{w}\frac{\partial}{\partial x_j}\mathbf{w}\right)\cdot\mathbf{n}ds \tag{A.10}$$

$$\oint_{\delta D} \left(\mathbf{w}|\mathbf{w}^2|\right)\cdot\mathbf{n}ds \tag{A.11}$$

$$\oint_{\delta D} \left(\mathbf{w}|\mathbf{w}\mathbf{u}|\right)\cdot\mathbf{n}ds \tag{A.12}$$

Since $\mathbf{v}(\mathbf{x},0) = \mathbf{u}(\mathbf{x},0) = 0$ on $\delta D$, and $\frac{\partial\mathbf{v}}{\partial t} = 0$ on $\delta D$ then $\mathbf{w}(\mathbf{x},t) = 0, \forall t$ on $\delta D$. Assuming that neither $P_w$ or$\left(\frac{\partial}{\partial x_j}\mathbf{w}\right)\cdot\mathbf{n}$ are equal to infinity on $\delta D$, then (A.9) through (A.12) are all equal to 0. Thus, (A.8) simplifies to (A.13).

$$\frac{d}{dt}K(t) = -\int_{D_f}\mathbf{w}\cdot\left(\mathbf{w}\cdot\overrightarrow{\nabla}\right)\mathbf{u}d\mathbf{x} - \nu\int_{D_f}\left(\overrightarrow{\nabla}\mathbf{w}\right)^2 d\mathbf{x} \tag{A.13}$$

**Strain rate Tensor** Consider $\mathbf{s}$ the strain rate tensor of $\mathbf{u}$ such that $s_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)$. As $\mathbf{s}$ is symmetric then $s_{ij} = \frac{\partial u_i}{\partial x_j}$. thus, we can write $\left(\mathbf{w}\cdot\overrightarrow{\nabla}\right)\mathbf{u} = \mathbf{w}\cdot\mathbf{s}$. Furthermore, by writing that $\|\mu_{\inf}\|$ is the maximum eigenvalue of $\mathbf{s}$ at any time $t$ and for all space, then $|-\int_{D_f}\mathbf{w}\cdot\left(\mathbf{w}\cdot\overrightarrow{\nabla}\right)\mathbf{u}d\mathbf{x}| \leq \|\mu_{\inf}\|\int_{D_f}|\mathbf{w}|^2 d\mathbf{x}$ , and so the inequality (A.14) is obtained from (A.13).

$$\frac{d}{dt}K(t) \leq \|\mu_{\inf}\|\int_{D_f}|\mathbf{w}|^2 d\mathbf{x} - \nu\int_{D_f}\left(\overrightarrow{\nabla}\mathbf{w}\right)^2 d\mathbf{x} \tag{A.14}$$

**Poincarré inequality** The Pointcarré inequality is used to get rid of the $\left(\overrightarrow{\nabla}\mathbf{w}\right)^2$ term from (A.14). Consider that $D_f$ is a domain where $-\triangle$ is a strictly positive self-adjoint linear operator who's smallest eigenvalue $\lambda \geq 0$. In this case, The Pointcarré inequality states that $\frac{1}{\lambda}\int_{D_f}\left(\overrightarrow{\nabla}\mathbf{w}\right)^2 d\mathbf{x} \geq \int_{D_f}|\mathbf{w}|^2 d\mathbf{x}$. By using this and (A.1), expression (A.15) is obtained. The

proof for the Pointcarré inequality is given in [2], theorem 2.1 p.36.

$$\frac{d}{dt}K\left(t\right) \leq 2\|\mu_{\text{inf}}\|K\left(t\right) - 2\nu\lambda K\left(t\right) \tag{A.15}$$

Solving the ODE from (A.15) gives expression (A.16). Note that the integral appears inside the exponential as $\|\mu_{\text{inf}}\|$ depends on time. It was said before that $\mathbf{w}\left(\mathbf{x},0\right)=0$, hence $K\left(0\right)=0$. Thus, (A.15) leads to $K\left(t\right)\leq 0$. However, from (A.1), it is clear that $K\left(t\right)\geq 0$ as $K\left(t\right)$ is an energy term. the only solution is therefore $K\left(t\right)=0,\forall t$.

$$K\left(t\right) = K\left(0\right)\exp\left(-2t\nu\lambda K\left(t\right) + 2\int_0^t \|\mu_{\text{inf}}\|d\tau\right) \tag{A.16}$$

# B. Distribution method computational formulation

## B.1. One Dimension

The source function $S(\mathbf{x}')$ will be determined numerically, so the domain and the solid are both discretised. More specifically, the boundary values $F(\mathbf{x})$ are discretised such that $F(\mathbf{x})$ is given in (B.1). Note that there are as many discrete boundary points as there are imposed Dirac sources. Furthermore, in the one dimensional case, the solid is necessarily a segment and it has two boundary points only.

$$F(x) = \begin{bmatrix} F(x_1) \\ F(x_2) \end{bmatrix} \tag{B.1}$$

Now rewriting the right hand side of (2.20) as the matrix form $\mathbf{Au} = \mathbf{F}$, and using (B.1) we obtain (B.2) $x_1 \leq x_1' \leq x_2' \leq x_2$ to avoid singularities.

$$\begin{bmatrix} \frac{x_1 - x_1}{|x_1 - x_1|^3} & \frac{x_1 - x_2}{|x_1 - x_2|^3} \\ \frac{x_2 - x_1}{|x_2 - x_1|^3} & \frac{x_2 - x_2}{|x_2 - x_2|^3} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} F(x_1) \\ F(x_2) \end{bmatrix} \tag{B.2}$$

The solution $\mathbf{u}$ is then easily obtained from (B.2) by writing $\mathbf{u} = \frac{\mathbf{F}}{\mathbf{A}}$. The source function $S(\mathbf{x}')$ can then be obtained from (2.7).

## B.2. Three Dimensions

In three dimensions, each point on the boundary of the solid bears three conditions that need to be satisfied by (2.12) and $\mathbf{u}_j$ has three values. Equation (B.3) show the right hand side of (2.21) and (2.22) after discretisation and in matrix form.

$$F_x(\mathbf{x}) = \begin{bmatrix} F_x(\mathbf{x}_1) \\ F_x(\mathbf{x}_2) \\ . \\ . \\ F_x(\mathbf{x}_n) \end{bmatrix}, \; F_y(\mathbf{x}) = \begin{bmatrix} F_y(\mathbf{x}_1) \\ F_y(\mathbf{x}_2) \\ . \\ . \\ F_y(\mathbf{x}_n) \end{bmatrix}, \; F_z(\mathbf{x}) = \begin{bmatrix} F_z(\mathbf{x}_1) \\ F_z(\mathbf{x}_2) \\ . \\ . \\ F_z(\mathbf{x}_n) \end{bmatrix} \tag{B.3}$$

### B.2.1. Isotropic pressure Dirac distribution

Rewriting (2.21) as the matrix form

$$\mathbf{A_x u} = \mathbf{F_x}$$
$$\mathbf{A_y u} = \mathbf{F_y}$$
$$\mathbf{A_z u} = \mathbf{F_z}$$

And using (B.3) we obtain (B.4). Note that to have a well conditioned problem, the size of $\mathbf{u}$ must be equal to the size of $\mathbf{F_x} + \mathbf{F_y} + \mathbf{F_z}$. Thus here we must have $m = 3n$.

$$\begin{bmatrix} \frac{x_1-x_1}{\sqrt{(x_1-x_1)^2+(y_1-y_1)^2+(z_1-z_1)^2}^3} & \cdots & \frac{x_1-x_m}{\sqrt{(x_1-x_m)^2+(y_1-y_m)^2+(z_1-z_m)^2}^3} \\ . & . & . \\ \frac{x_n-x_1}{\sqrt{(x_n-x_1)^2+(y_n-y_1)^2+(z_n-z_1)^2}^3} & \cdots & \frac{x_n-x_m}{\sqrt{(x_n-x_m)^2+(y_n-y_m)^2+(z_n-z_m)^2}^3} \end{bmatrix} \begin{bmatrix} u_1 \\ . \\ u_m \end{bmatrix} = \begin{bmatrix} F_x(\mathbf{x}_1) \\ ... \\ F_x(\mathbf{x}_n) \end{bmatrix} \quad \text{(B.4)}$$

$$\begin{bmatrix} \frac{y_1-y_1}{\sqrt{(x_1-x_1)^2+(y_1-y_1)^2+(z_1-z_1)^2}^3} & \cdots & \frac{y_1-y_m}{\sqrt{(x_1-x_m)^2+(y_1-y_m)^2+(z_1-z_m)^2}^3} \\ . & . & . \\ \frac{y_n-y_1}{\sqrt{(x_n-x_1)^2+(y_n-y_1)^2+(z_n-z_1)^2}^3} & \cdots & \frac{y_n-y_m}{\sqrt{(x_n-x_m)^2+(y_n-y_m)^2+(z_n-z_m)^2}^3} \end{bmatrix} \begin{bmatrix} u_1 \\ . \\ u_m \end{bmatrix} = \begin{bmatrix} F_y(\mathbf{x}_1) \\ ... \\ F_y(\mathbf{x}_n) \end{bmatrix} \quad \text{(B.5)}$$

$$\begin{bmatrix} \frac{z_1-z_1}{\sqrt{(x_1-x_1)^2+(y_1-y_1)^2+(z_1-z_1)^2}^3} & \cdots & \frac{z_1-z_m}{\sqrt{(x_1-x_m)^2+(y_1-y_m)^2+(z_1-z_m)^2}^3} \\ . & . & . \\ \frac{z_n-z_1}{\sqrt{(x_n-x_1)^2+(y_n-y_1)^2+(z_n-z_1)^2}^3} & \cdots & \frac{z_n-z_m}{\sqrt{(x_n-x_m)^2+(y_n-y_m)^2+(z_n-z_m)^2}^3} \end{bmatrix} \begin{bmatrix} u_1 \\ . \\ u_m \end{bmatrix} = \begin{bmatrix} F_z(\mathbf{x}_1) \\ ... \\ F_z(\mathbf{x}_n) \end{bmatrix} \quad \text{(B.6)}$$

The solution $\mathbf{u}$ is then easily obtained from (B.4) by writing $\mathbf{u} = \frac{\mathbf{F}}{\mathbf{A}}$. The source function $S(\mathbf{x}')$ can then be obtained from (2.7).

### B.2.2. Non-isotropic pressure Dirac distribution

Rewriting (2.22) as the matrix form

$$\mathbf{A_x u_x} = \mathbf{F_x}$$
$$\mathbf{A_y u_y} = \mathbf{F_y}$$
$$\mathbf{A_z u_z} = \mathbf{F_z}$$

And using (B.3) we obtain (B.7). Note that for the problem to be well dimensioned, we must have the size of $\mathbf{u_x}$ equal to the size of $\mathbf{F_x}$ and so on. Thus here we have $m = n$.

$$\begin{bmatrix} \frac{x_1-x_1}{\sqrt{(x_1-x_1)^2+(y_1-y_1)^2+(z_1-z_1)^2}^3} & \cdots & \frac{x_1-x_m}{\sqrt{(x_1-x_m)^2+(y_1-y_m)^2+(z_1-z_m)^2}^3} \\ . & . & . \\ \frac{x_n-x_1}{\sqrt{(x_n-x_1)^2+(y_n-y_1)^2+(z_n-z_1)^2}^3} & \cdots & \frac{x_n-x_m}{\sqrt{(x_n-x_m)^2+(y_n-y_m)^2+(z_n-z_m)^2}^3} \end{bmatrix} \begin{bmatrix} u_{x,1} \\ . \\ u_{x,m} \end{bmatrix} = \begin{bmatrix} F_x(\mathbf{x}_1) \\ ... \\ F_x(\mathbf{x}_n) \end{bmatrix} \quad \text{(B.7)}$$

$$\begin{bmatrix} \frac{y_1-y_1}{\sqrt{(x_1-x_1)^2+(y_1-y_1)^2+(z_1-z_1)^2}^3} & \cdots & \frac{y_1-y_m}{\sqrt{(x_1-x_m)^2+(y_1-y_m)^2+(z_1-z_m)^2}^3} \\ . & . & . \\ \frac{y_n-y_1}{\sqrt{(x_n-x_1)^2+(y_n-y_1)^2+(z_n-z_1)^2}^3} & \cdots & \frac{y_n-y_m}{\sqrt{(x_n-x_m)^2+(y_n-y_m)^2+(z_n-z_m)^2}^3} \end{bmatrix} \begin{bmatrix} u_{y,1} \\ . \\ u_{y,m} \end{bmatrix} = \begin{bmatrix} F_y(\mathbf{x}_1) \\ ... \\ F_y(\mathbf{x}_n) \end{bmatrix} \quad \text{(B.8)}$$

$$\begin{bmatrix} \frac{z_1-z_1}{\sqrt{(x_1-x_1)^2+(y_1-y_1)^2+(z_1-z_1)^2}^3} & \cdots & \frac{z_1-z_m}{\sqrt{(x_1-x_m)^2+(y_1-y_m)^2+(z_1-z_m)^2}^3} \\ . & . & . \\ \frac{z_n-z_1}{\sqrt{(x_n-x_1)^2+(y_n-y_1)^2+(z_n-z_1)^2}^3} & \cdots & \frac{z_n-z_m}{\sqrt{(x_n-x_m)^2+(y_n-y_m)^2+(z_n-z_m)^2}^3} \end{bmatrix} \begin{bmatrix} u_{z,1} \\ . \\ u_{z,m} \end{bmatrix} = \begin{bmatrix} F_z(\mathbf{x}_1) \\ ... \\ F_z(\mathbf{x}_n) \end{bmatrix} \quad \text{(B.9)}$$

The solution $\mathbf{u}$ is then easily obtained from (B.7) by writing $\mathbf{u_{x,y,z}} = \frac{\mathbf{F_{x,y,z}}}{\mathbf{A_{x,y,z}}}$. The source function $S(\mathbf{x}')$ can then be obtained from (2.11).

### B.2.3. Isotropic pressure Dipole Distribution

Rewriting (2.23) as the matrix form

$$\mathbf{A_x u} = \mathbf{F_x}$$
$$\mathbf{A_y u} = \mathbf{F_y}$$
$$\mathbf{A_z u} = \mathbf{F_z}$$

And using (B.3) we obtain (B.10). Note that for the problem to be well dimensioned, we must have the size of $\mathbf{u} = \begin{bmatrix} \mathbf{u}_o \\ \mathbf{u}_x \\ \mathbf{u}_y \\ \mathbf{u}_z \end{bmatrix}$ equal to the size of $\mathbf{F_x} + \mathbf{F_y} + \mathbf{F_z}$ and so on. Thus here we must have $m = \frac{3}{4}n$.

$$\begin{bmatrix} \mathbf{A_{1,x}} & \mathbf{A_{2,x}} & \mathbf{A_{3,x}} & \mathbf{A_{4,x}} \end{bmatrix} \begin{bmatrix} \begin{bmatrix} u_{o,1} \\ . \\ u_{o,m} \end{bmatrix} \\ \begin{bmatrix} u_{x,1} \\ . \\ u_{x,m} \end{bmatrix} \\ \begin{bmatrix} u_{y,1} \\ . \\ u_{y,m} \end{bmatrix} \\ \begin{bmatrix} u_{z,1} \\ . \\ u_{z,m} \end{bmatrix} \end{bmatrix} = \begin{bmatrix} F_x(\mathbf{x}_1) \\ ... \\ F_x(\mathbf{x}_n) \end{bmatrix} \tag{B.10}$$

$$\begin{bmatrix} \mathbf{A_{1,y}} & \mathbf{A_{2,y}} & \mathbf{A_{3,y}} & \mathbf{A_{4,y}} \end{bmatrix} \begin{bmatrix} \begin{bmatrix} u_{o,1} \\ . \\ u_{o,m} \end{bmatrix} \\ \begin{bmatrix} u_{x,1} \\ . \\ u_{x,m} \end{bmatrix} \\ \begin{bmatrix} u_{y,1} \\ . \\ u_{y,m} \end{bmatrix} \\ \begin{bmatrix} u_{z,1} \\ . \\ u_{z,m} \end{bmatrix} \end{bmatrix} = \begin{bmatrix} F_y(\mathbf{x}_1) \\ ... \\ F_y(\mathbf{x}_n) \end{bmatrix} \tag{B.11}$$

$$\begin{bmatrix} \mathbf{A_{1,z}} & \mathbf{A_{2,z}} & \mathbf{A_{3,z}} & \mathbf{A_{4,z}} \end{bmatrix} \begin{bmatrix} \begin{bmatrix} u_{o,1} \\ . \\ u_{o,m} \end{bmatrix} \\ \begin{bmatrix} u_{x,1} \\ . \\ u_{x,m} \end{bmatrix} \\ \begin{bmatrix} u_{y,1} \\ . \\ u_{y,m} \end{bmatrix} \\ \begin{bmatrix} u_{z,1} \\ . \\ u_{z,m} \end{bmatrix} \end{bmatrix} = \begin{bmatrix} F_z(\mathbf{x}_1) \\ ... \\ F_z(\mathbf{x}_n) \end{bmatrix} \tag{B.12}$$

Where the expression for $\mathbf{A_y}$ ($y$ is chosen as the example direction) are given in (B.13).

$$\mathbf{A_{1,y}} = \begin{bmatrix} \frac{y_1-y_1}{\sqrt{(x_1-x_1)^2+(y_1-y_1)^2+(z_1-z_1)^2}^3} & \cdots & \frac{y_1-y_m}{\sqrt{(x_1-x_m)^2+(y_1-y_m)^2+(z_1-z_m)^2}^3} \\ . & . & . \\ \frac{y_n-y_1}{\sqrt{(x_n-x_1)^2+(y_n-y_1)^2+(z_n-z_1)^2}^3} & \cdots & \frac{y_n-y_m}{\sqrt{(x_n-x_m)^2+(y_n-y_m)^2+(z_n-z_m)^2}^3} \end{bmatrix} \tag{B.13}$$

$$\mathbf{A_{2,y}} = -\begin{bmatrix} \frac{\partial\left(\frac{(y_1-y_1)}{|\mathbf{x_1}-\mathbf{x_1}|^3}\right)}{\partial x} & \cdots & \frac{\partial\left(\frac{(y_1-y_m)}{|\mathbf{x_1}-\mathbf{x}_m|^3}\right)}{\partial x} \\ . & . & . \\ \frac{\partial\left(\frac{(y_n-y_1)}{|\mathbf{x_n}-\mathbf{x_1}|^3}\right)}{\partial x} & \cdots & \frac{\partial\left(\frac{(y_n-y_m)}{|\mathbf{x_n}-\mathbf{x}_m|^3}\right)}{\partial x} \end{bmatrix} \tag{B.14}$$

$$\mathbf{A_{3,y}} = -\begin{bmatrix} \frac{\partial\left(\frac{(y_1-y_1)}{|\mathbf{x_1}-\mathbf{x_1}|^3}\right)}{\partial y} & \cdots & \frac{\partial\left(\frac{(y_1-y_m)}{|\mathbf{x_1}-\mathbf{x}_m|^3}\right)}{\partial y} \\ . & . & . \\ \frac{\partial\left(\frac{(y_n-y_1)}{|\mathbf{x_n}-\mathbf{x_1}|^3}\right)}{\partial y} & \cdots & \frac{\partial\left(\frac{(y_n-y_m)}{|\mathbf{x_n}-\mathbf{x}_m|^3}\right)}{\partial y} \end{bmatrix} \tag{B.15}$$

$$\mathbf{A_{4,y}} = -\begin{bmatrix} \frac{\partial\left(\frac{(y_1-y_1)}{|\mathbf{x_1}-\mathbf{x_1}|^3}\right)}{\partial z} & \cdots & \frac{\partial\left(\frac{(y_1-y_m)}{|\mathbf{x_1}-\mathbf{x}_m|^3}\right)}{\partial z} \\ . & . & . \\ \frac{\partial\left(\frac{(y_n-y_1)}{|\mathbf{x_n}-\mathbf{x_1}|^3}\right)}{\partial z} & \cdots & \frac{\partial\left(\frac{(y_n-y_m)}{|\mathbf{x_n}-\mathbf{x}_m|^3}\right)}{\partial z} \end{bmatrix} \tag{B.16}$$

The solution $\mathbf{u}$ is then easily obtained from (B.7) by writing $\mathbf{u} = \frac{\mathbf{F}}{\mathbf{A}}$. The source function $S(\mathbf{x}')$ can then be obtained from (2.11).

# C. Numerical Implementation

## C.1. Program Architecture

The program for finding $S(\mathbf{x})$ has been written in matlab language for simplicity. It is divided into three parts :

- Running parameters definition : This is where the user defines the running parameters which are specific to the problem at hand
- Initialisation and discretisation : This part is common for 1D and 2D problems and for any type of method used. It simply discretises the domain and the boundary of the solid based on the requirements of the user. It also assigns the inputted values for the boundary points.
- Solvers : There are two solvers present in the program at the moment. The 1D solver is capable of using either Finite Element based methods, or Dirac Distribution based methods. The 2D solver, however, is only capable of using Dirac distribution methods for isotropic and non isotropic pressure fields, with monopole or dipole distributions. The Finite Element method could be implemented, but would require good knowledge of 2D element construction.

The Architecture of the program is shown in C.1 and C.2.

### C.1.1. Defining Running parameters

The parameters that need to be defined by the user are the following.

- The method used for finding $S(\mathbf{x})$. The variable 'method' indicates whether the user uses the Dirac distribution method (method = 1) or the finite element discretisation method (method = 2). Note that currently, the finite element method is only supported in 1D. In the case of Dirac distributions for 2D problems, the user can select to use a isotropic pressure solver with monopoles, (dirac_type = 1) or a non-isotropic pressure solver with monopoles (dirac_type = 2), or an isotropic pressure solver with dipoles (dirac_type = 3).
- The size of the domain of fluid study (xmi,xmax etc). Note that this determines the working domaine of the program, and is not necessarily the fluid simulation domain. The only condition imposed in the defined domain size is that the solid be contained within the domain.

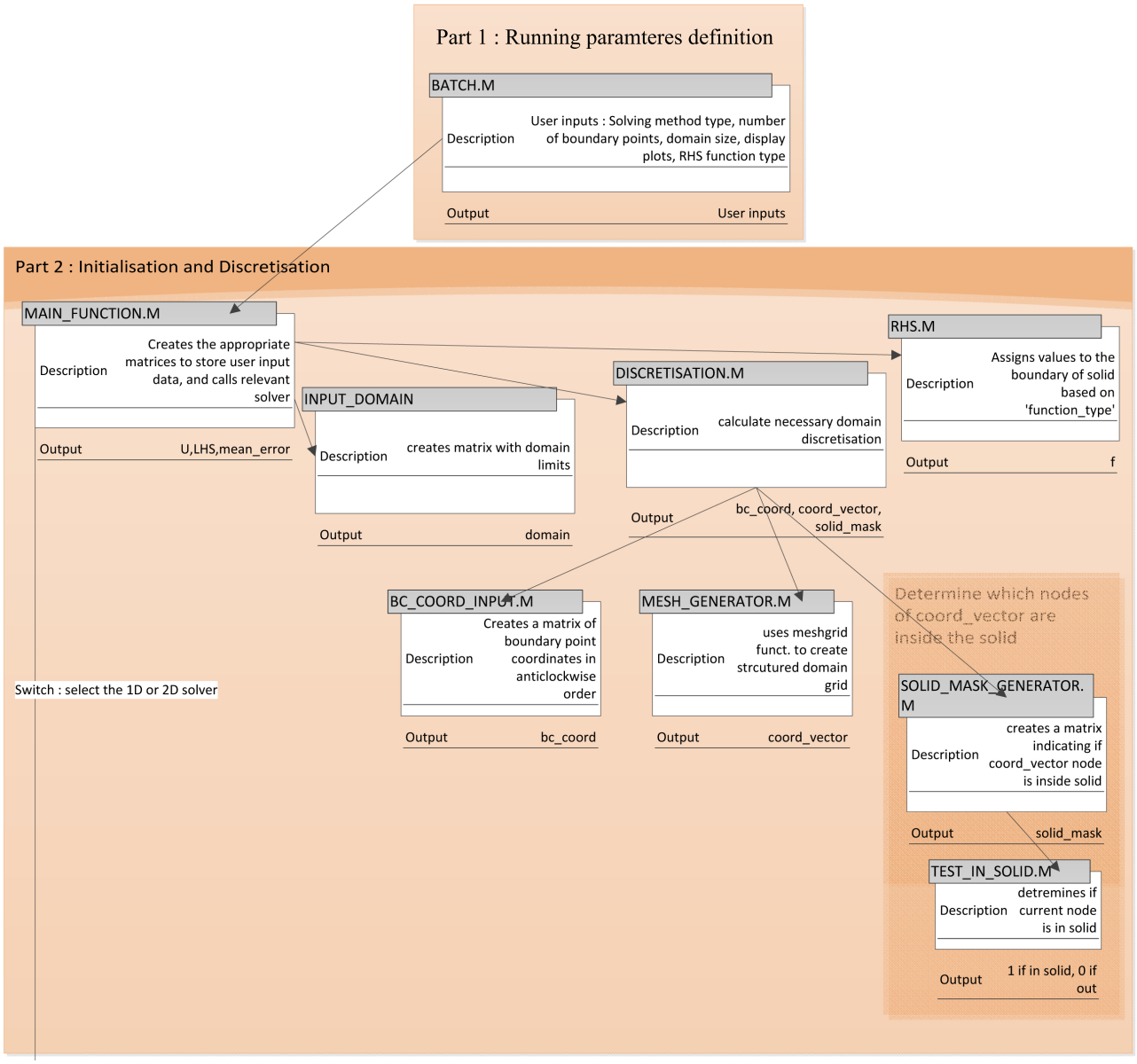



Figure C.1.: Program Architecture Diagram : Parts 1 and 2

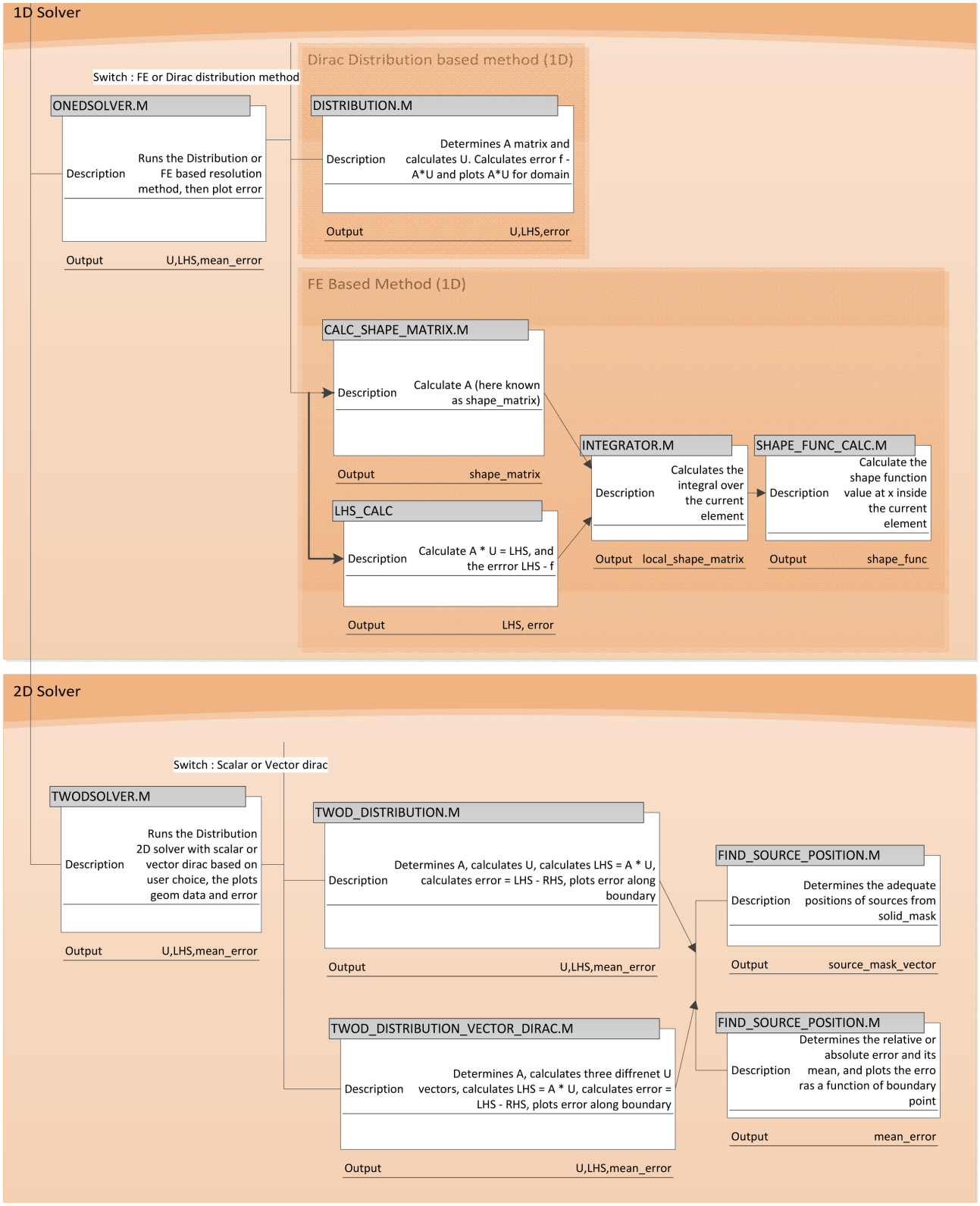



Figure C.2.: Program Architecture Diagram : Part 3

- The dimension of the problem through the variable 'dimension' : $1 = 1D$, $2 = 2D$ etc.
- The number of boundary points required to discretise the boundary. Note that each boundary point requires assigned values of $\mathbf{F}(\mathbf{x})$ which would have been obtained through the CFD simulation. Furthermore, the zone inside the solid will be determined through those boundary points. Because of this, a 2D simulation requires a minimum of 3 boundary points to give results.

  In the case of open solids, such as a channel flow simulation, it is recommended to transformed the open solids into closed solids. Thus, for the case of channel flow, the two walls would be represented by 2 closed solid rectangles.
- The source position is determined by the variable 'source_position'. If the variable is equal to 1, the sources are set within the solid shape. This is adapted to any closed solid that does not intersect itself. If the variable is equal to -1, the points are set outside the solid.
- The shape of the solid within the fluid through the variable 'shape'. The program supports any kind of of closed shape that does not intersect itself. For the moment, the rectangle, cylindre and channel shape have already been predefined. Any new shape must be defined so that the coordinates of the points on the boundaries of the shape are given in anticlockwise order (trigonometric direction). This is done in the function BC_ coord_ input.m .
- The value of $\mathbf{F}(\mathbf{x})$ can be defined through the variable 'function_ type'. Different functions have already been predefined and are accessible through the integer value given to 'function_ type'. Additional values can be in-putted in the function RHS.m
- The variable 'show_ plots is set to 1 if the user wants to see the geometrical plot and absolute error plots. When executing batch tests, it is recommended to be set to 0 for speed.
- The variable 'geom_ correction' is increased to increase the refinement of the structured grid mesh, and thus have a greater number of nodes inside the solid. See C.1.4 for details.

### C.1.2. Positioning of elements

In the case of the method based on Finite elements, the elements must necessarily lie within the solid, and away from the boundary. Indeed, the values taken by the integral of the shape function over the element will tend to infinity at various points, as shown in C.3. Infinite values close to boundaries will lead to important calculation errors for $S(\mathbf{x})$.

### C.1.3. Positioning of Dirac sources

In theory Dirac sources are equal to infinity only at the exact position of the source, are are equal to 0 everywhere else. In practice, an important region around the Dirac source continues to show the presence of the discontinuity. This is shown in figure C.4 where the sources were positioned

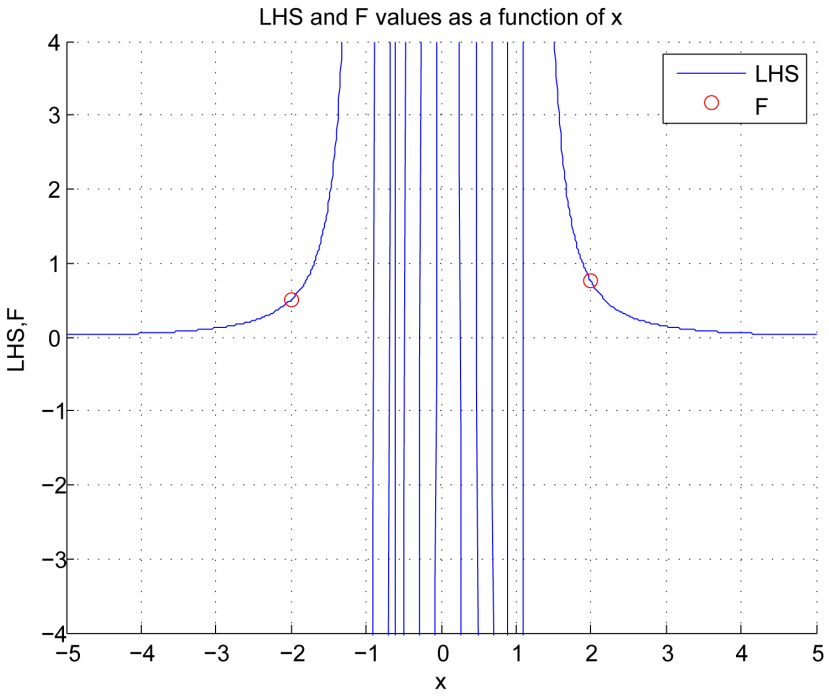



Figure C.3.: Value of LHS on the domain for 1D FE method

at $x = -1$ and $x = 1$ respectively. It is therefore very important to keep the Dirac sources and their discontinuity away from the boundary of the solid. This will be even more important if points of LHS inside the solid are required for interpolation purposes. Also, the sources must be positioned away from the grid points where pressure will be calculated, to avoid potential peaks of the pressure variation in the CFD code. Lastly, for 2D or 3D problems, positioning the same types of sources symmetrically with respect to the the solid, or too close together, will cause the matrix to be ill conditioned. This is because the base size of the vectorial space, and therefore its dimension, generated by the source distributions will decrease. Thus the rank of the matrix will decrease and the matrix will become ill-conditioned. To avoid this, each source is placed at a random small distance (maximum value of $\delta x/10$) of a different node of a structured grid within the solid. Figure C.5 shows the positioning of sources within a circle in a 2D case. Note how all the sources are slightly off from the structure grid nodes because of the random shift.

### C.1.4. Discretisation

Function 'discretisation.m' is responsible for the discretisation of the boundary, and of the working domain. To optimize the speed of the calculation of $S(\mathbf{x})$, the domain structured grid space step is defined automatically using the chosen number of points on the boundary. Of course, to determine exactly the number of points within a complicated shape, like a rose petal for example, would require the program to iteratively generate new meshes with different numbers of total nodes, then count the nodes within the solid to determine if there are enough to place all sources. This is computationally costly. Instead, the program simply looks at the

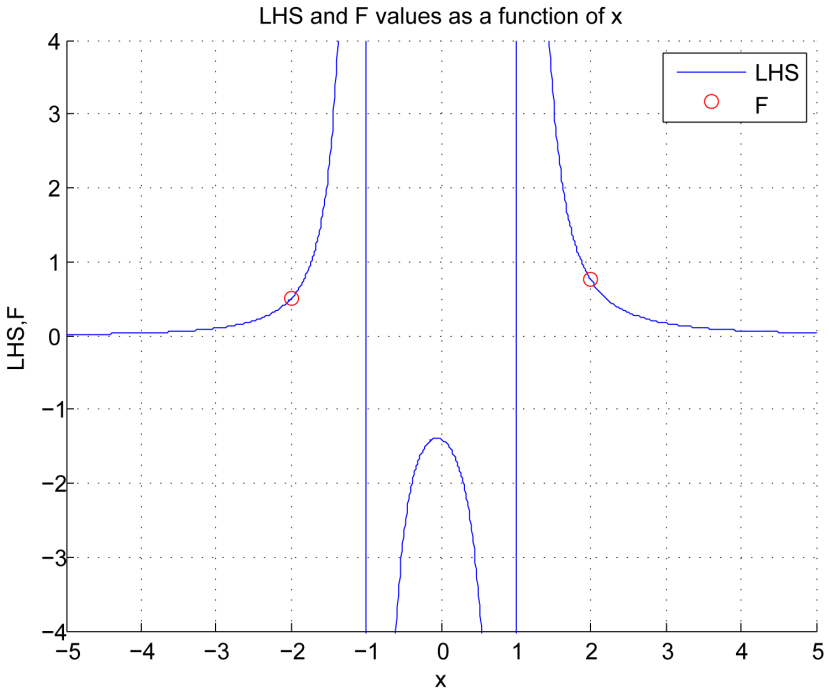



Figure C.4.: Value of LHS on the domain for 1D Dirac distribution method

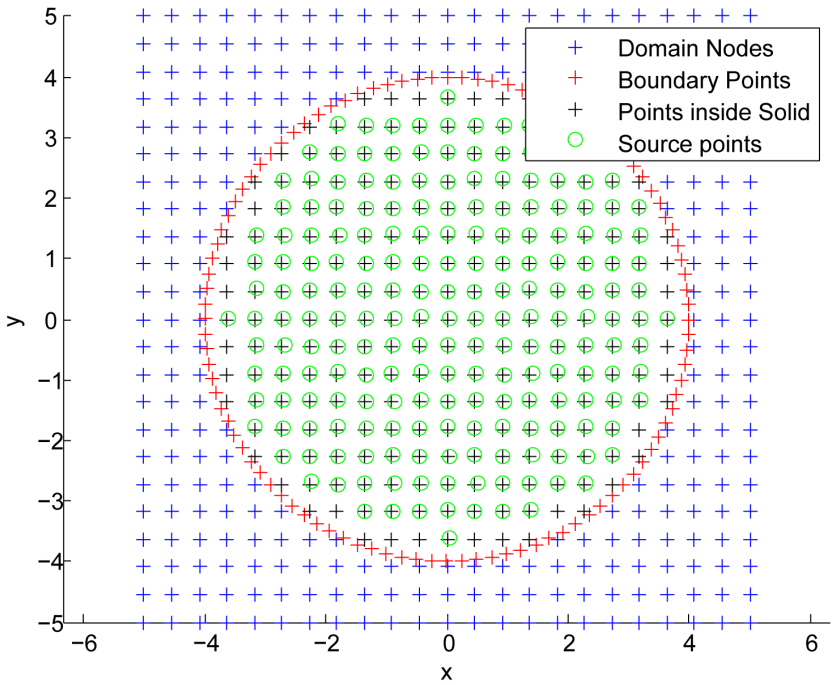



Figure C.5.: Position of solid and source points on the domain grid for a 2D isotropic pressure Dirac distribution method

max and min values of the solid along the x, y, and z directions. It then considers how many domain structured grid nodes are required to fill a parallelepiped of that size with a enough nodes for each source, whilst keeping the space step in each direction approximately the same. Of course, shapes with less inner volume for the same max dimension than the rectangle will require a finer mesh. To account for this, the correction factor 'geom_ correction' can be set by the user to correct this with the help of the geometric plots. A value of 1.25 is adapted to cylinders.

### C.1.5. Solid Mask Generator

The solid mask generator creates a matrix where each cell corresponds to a domain node. If the cell value is 1, the domain structured mesh node is within the solid. Consider here that the points on the boundary are sequentially given by following the trigonometric rotation direction. A node is determined to be within the solid by using vector calculus. The two closest points (point 1 and 2) on the boundary are determined for each node (A) of the structure grid. Then , the vector product of vector 1 and vector 2 defined by A and point 1 , and A and point 2 respectively is calculated. If the vector product is positive, then the shift from vector 1 to vector 2 is done in the trigonometric (anticlockwise) direction and thus the node A is within the solid. If the product is negative, then the point is outside of the solid. This method only works if the points on the boundary are given in the trigonometric order, and if the solid does not intersect itself.

# D. Test Cases

## D.1. Solid Cylinder

The following battery of tests look at the difference between the three distribution methods specifically for a Cylinder. The use of non-isotropic or isotropic pressure distributions will impact calculation time, and calculation error. Note that the relative error given here is defined as $RE = \sqrt{\frac{(F_x - LHS_x)^2 + (F_y - LHS_y)^2}{(F_x + F_y)^2}}$.

### D.1.1. Calculation error due to RHS functional space

As stated before, the distribution method is a linearisation of the source function $\mathbf{S}(\mathbf{x})$. Therefore, the accuracy of the results depends on the capacity of the integral of the linear combination to accurately capture the function $\mathbf{F}(\mathbf{x})$. To show this, the periodic function $\mathbf{F}(\mathbf{x})$ will be defined as shown in figure D.1. All other running parameters remain the same as for the 2D proof of concept.

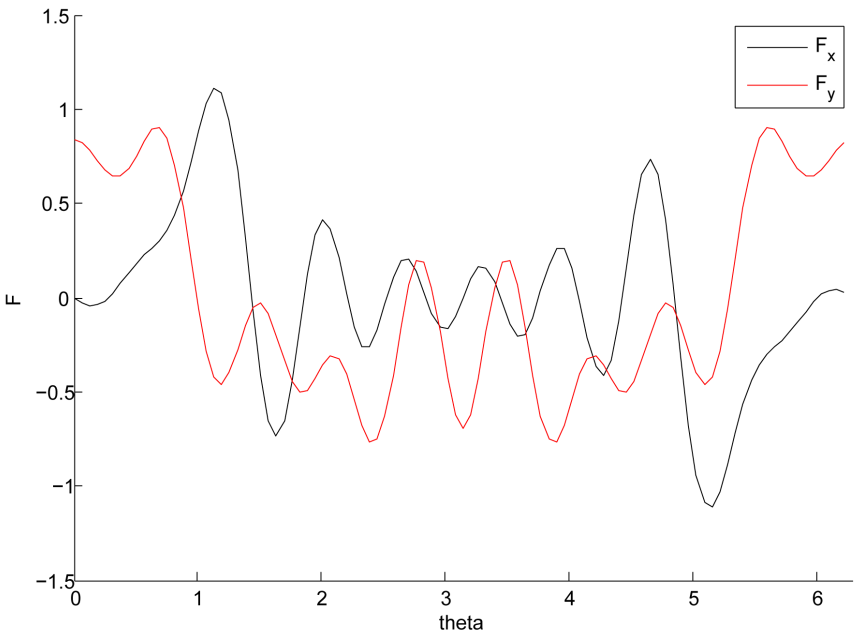



Figure D.1.: Fourier function satisfied by $\mathbf{F}(\mathbf{x})$

The simulation took 1.28 and 0.92 CPU seconds for the isotropic and the non isotropic pressure cases respectively. The results are given in figures D.2 through D.4. As expected, the decoupling offered by the non-isotropic method gives the lowest relative error. This is shortly followed by

the isotropic pressure Dipole distribution method which offers near decoupling of the dimensions of the problem, and a wider variety of sources.

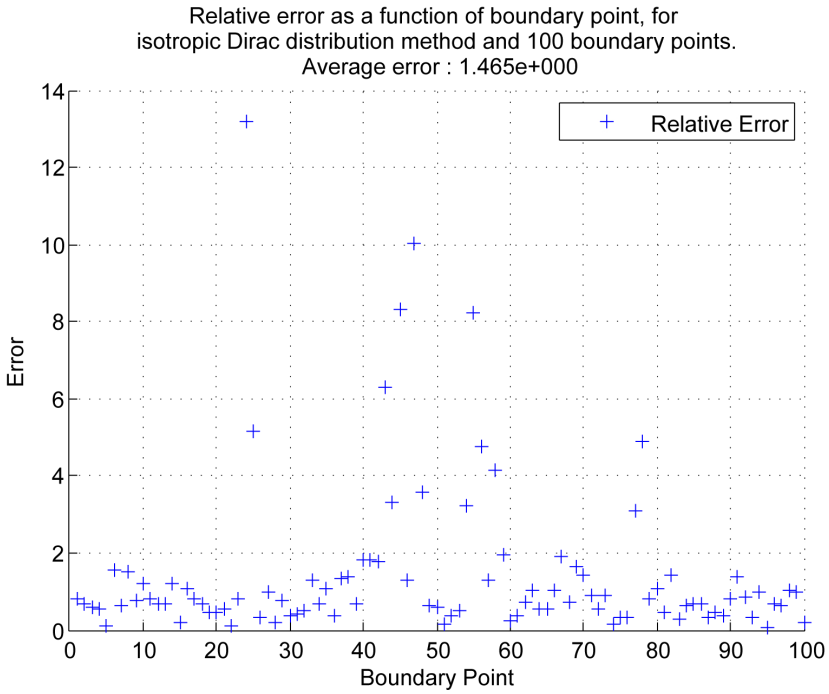



Figure D.2.: Fourier function error results for the isotropic pressure Dirac distribution method

### D.1.2. Calculation time

Figure D.5 shows the variation of calculation time with the number of boundary points. By considering that the number of points on the boundary is $N$, the size of the matrix to be inverted will be $2N * 2N = 4N^2$ terms where 2 corresponds to the number of dimensions for the isotropic pressure Dirac distribution cases. Similarly, the size of each matrix to be inverted will be $N * N = N^2$ terms for the non-isotropic pressure Dirac distribution method, making a total of $2N^2$ terms for both matrices. In 3D, the number of terms for the scalar and non-isotropic pressure Dirac distribution methods would be $9N^2$ and $3N^2$. Because of this difference in the size of matrices to be inverted, the non-isotropic pressure Dirac distribution method is on average 4 times faster than the isotropic pressure Dirac distribution method. Considering that the source function $\mathbf{S}(\mathbf{x})$ will need to be computed at each iteration of the CFD code, the advantage in using non-isotropic pressure Dirac distribution methods for improving computational speed is important if the matrix $A$ varies. Note that the matrix $A$ will not vary if the source positions remain constant throughout the iterations; in this case, the matrix $A$ needs only to be inverted at the first iteration. Note that the difference in time between the two isotropic methods comes from mesh operations only, and not from the matrix inversion; indeed, the Dipole distribution method requires fewer source locations, and therefore a coarser mesh.

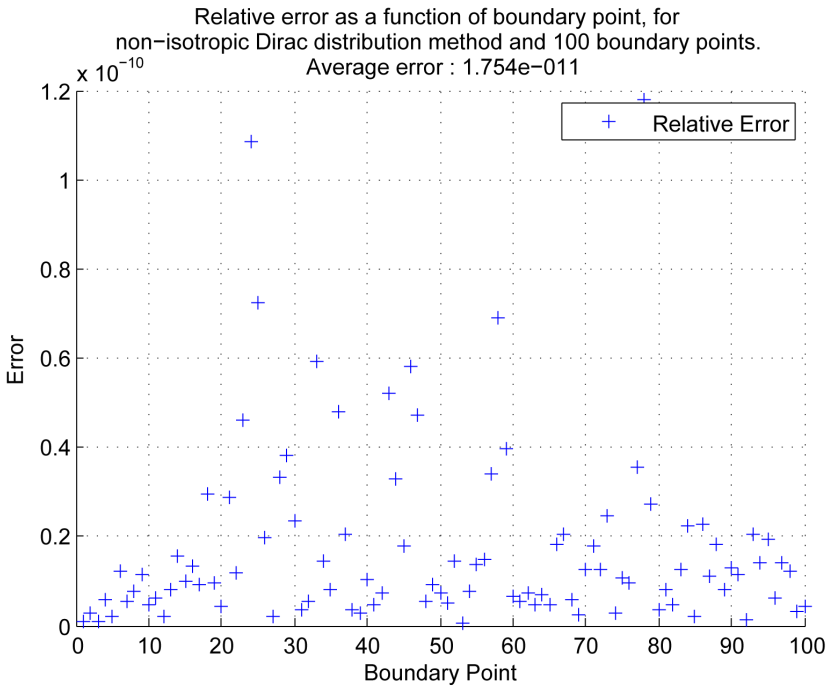



Figure D.3.: Fourier function error results for the non-isotropic pressure Dirac distribution method

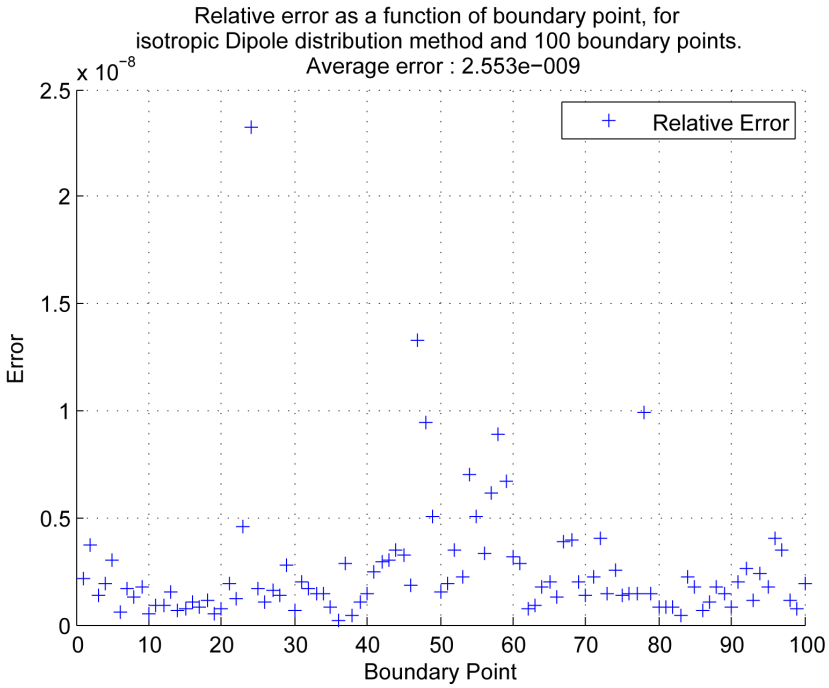



Figure D.4.: Fourier function error results for the isotropic pressure dipole distribution method

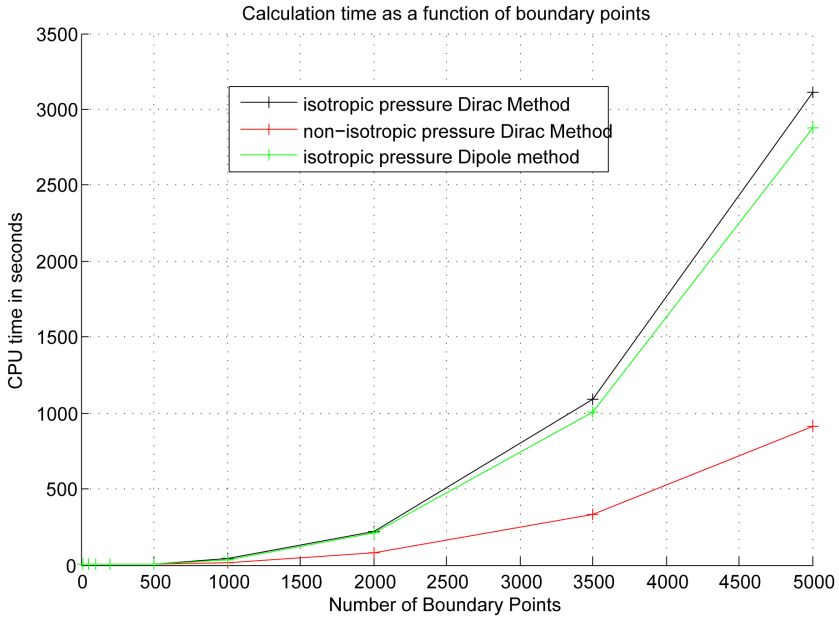



Figure D.5.: Claculation time as a function of the number of boundary points for 2D simulations.

### D.1.3. Calculation error due to sharp gradients in $\mathbf{F}(\mathbf{x})$

Calculation error is affected when sharp gradients are introduced between the values of $F(\mathbf{x})$ at consecutive boundary points. To show this, the same parameter as in the 2D proof of concept are used. The values of $\mathbf{F}(\mathbf{x})$, however, are set as follows : $F_x(\mathbf{x}) = \theta^3$ and $F_y(\mathbf{x}) = cos(\theta)$ where $\theta$ is the trigonometric angle, and $\theta = 0$ is on th epositive x axis of the cylinder. Thus, $F_x(\mathbf{x}) = \theta^3$ is not periodic. Figures D.6 through D.8 show the calculation error obtained. There are two things to be noted :

- Sharp gradient in $F(\mathbf{x})$ increases the calculation error
- In the isotropic pressure Dirac distribution case, the perturbation caused by a large gradient in the $x$ direction is felt also in the results for the $y$ direction. There is therefore coupling between the two directions. This is because in the isotropic pressure Dirac distribution case, each term of $U$ acts on a Dirac source that influences every direction. Thus, each term of $U$ influences every direction, and only one matrix is inverted to calculate $U$. In the non-isotropic pressure Dirac distribution case however, the perturbation caused by a gradient in the $x$ direction only influences the error along the $x$ direction. Indeed, there is no coupling between the directions of the problem with this method, as each direction is solved for independently by inverting a separate matrix. Furthermore, the error caused by the sharp gradient is 9 orders of magnitude smaller than with the isotropic pressure Dirac distribution method. The Dipole distribution performs better than the isotropic pressure Dirac distribution method because it allows a near decoupling of the direction of the problem. The impact of the sharp gradient is however felt on the error in both the x and y direction. The error is also considerably higher than with the non-isotropic pressure Dirac distribution method.

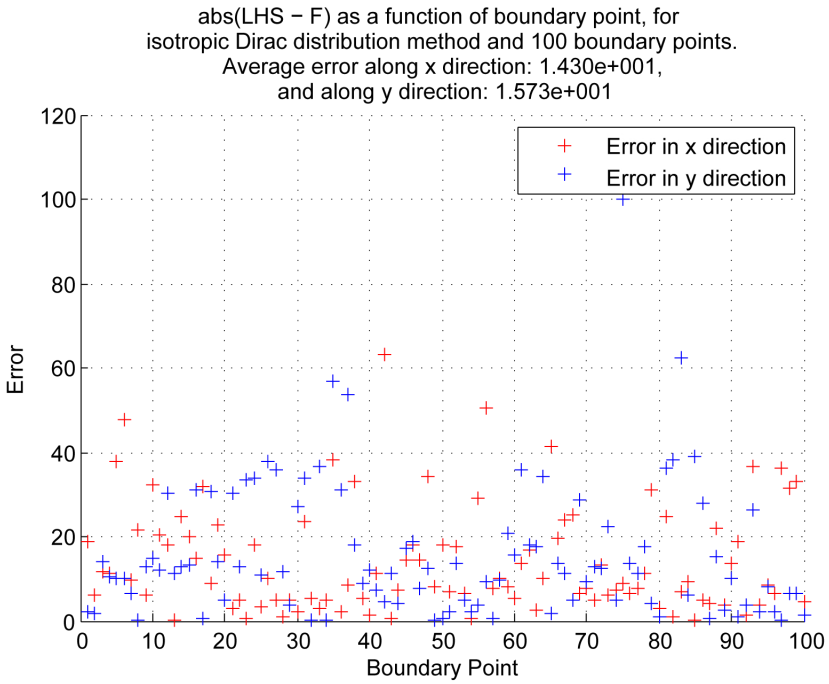



Figure D.6.: 2D result error for sharp gradients in $\mathbf{F}(\mathbf{x})$ with isotropic pressure Dirac distribution

## D.2. Influence of the number of points

The error obtained will depend greatly on $\mathbf{F}(\mathbf{x})$, and can be improved by increasing the number of sources. Indeed, consider that $\mathbf{F}(\mathbf{x})$ belongs to a specific functional space $\Omega$, and that the $n$ sources used to represent it generate a vectorial space of base $n$, $\Phi$. The bigger $n$, the bigger the intersections of the $\Omega$ and $\Phi$, and therefore the bigger the chance of finding $\mathbf{F}(\mathbf{x})$ at that intersection and reducing the error. However, by increasing the number of sources and therefore of boundary points, the calculation is more computationally costly. Also, a higher density of sources increases the risk of two sources merging together because of a too small distance between them. Furthermore, randomly generated symmetry patterns could start to appear. These last two effects will reduce the vector base generated by the source function, and therefore increase matrix conditioning and calculation error. Thus, to improve the conditioning of the matrix, the number of sources should be reduced, but to improve the obtained error, the number of sources should be increased. There is therefore an optimum which can be calculated. As a rule of thumb however, increasing the number of boundary points will lead to higher accuracy, at the cost of computational time.

## D.3. Channel flow

The channel structure is different from previously tested configurations a two solids are required to represent both walls. Thus, there are two areas where sources will be positioned.

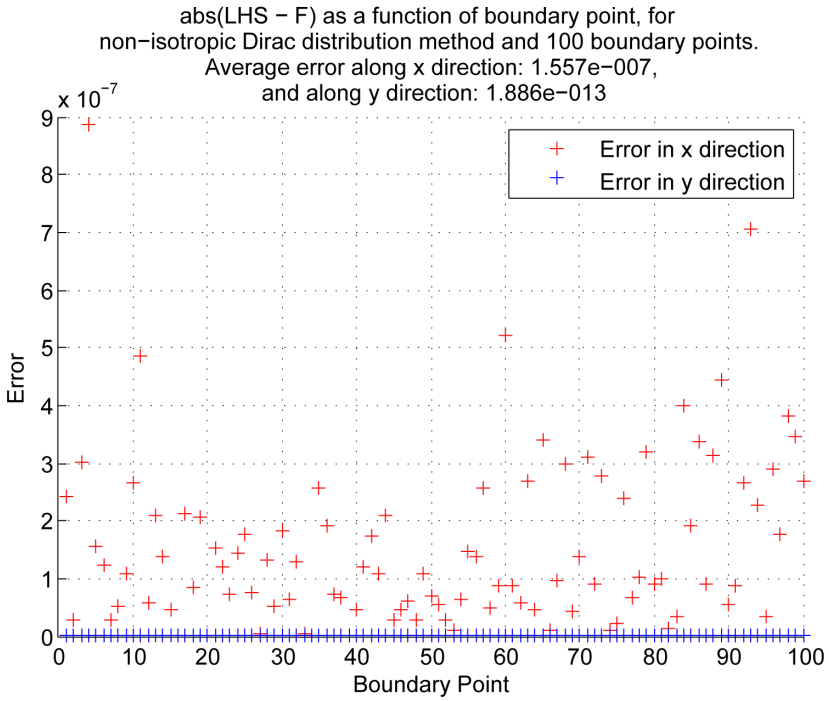



Figure D.7.: 2D result error for sharp gradients in $\mathbf{F}(\mathbf{x})$ with non-isotropic pressure Dirac distribution

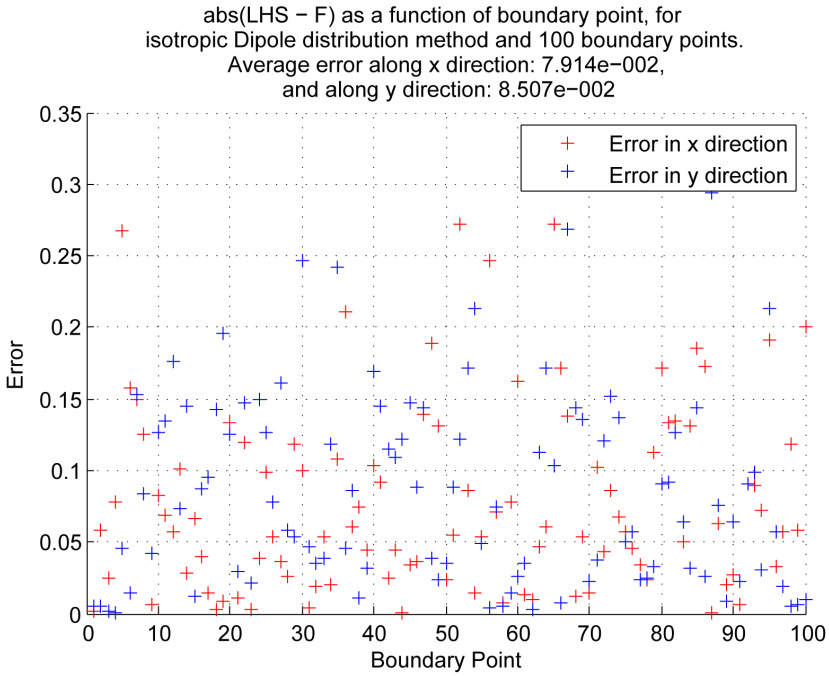



Figure D.8.: 2D result error for sharp gradients in $\mathbf{F}(\mathbf{x})$ with isotropic pressure Dipole distribution

For this test case, the domain will set as $(-5, 5)$ along the $x$ direction and $(-5, 5)$ along the $y$. The top wall solid will be defined along the $x$ direction as $(-5, 5)$, and along the $y$ direction as $(3, 5)$. The bottom wall solid will be defined similarly for the x direction, and as $(-5, -3)$ for the y direction. The number of boundary points is set to be 200, amounting to 100 points on each wall of the channel. The boundary value for $\mathbf{F}(\mathbf{x})$ is defined such that $F_x(\mathbf{x}) = sin(\theta)$ and $F_y(\mathbf{x}) = cos(\theta)$ where $\theta$ is the trigonometric angle . It is defined such that $\theta = 0$ on the first point of the top segment of the top solid, $\theta = \pi$ is the last point of the right segment of the top solid, $\theta = \pi - \delta\theta$ is the first point of the top segment of the bottom corner of the top solid, and $\theta = \pi - \delta\theta$ is the top right corner of the bottom solid, and $\theta = 2\pi$ is the last point of the right segment of the bottom solid. The vector plot for $\mathbf{F}(\mathbf{x})$ as a function of boundary point is shown in

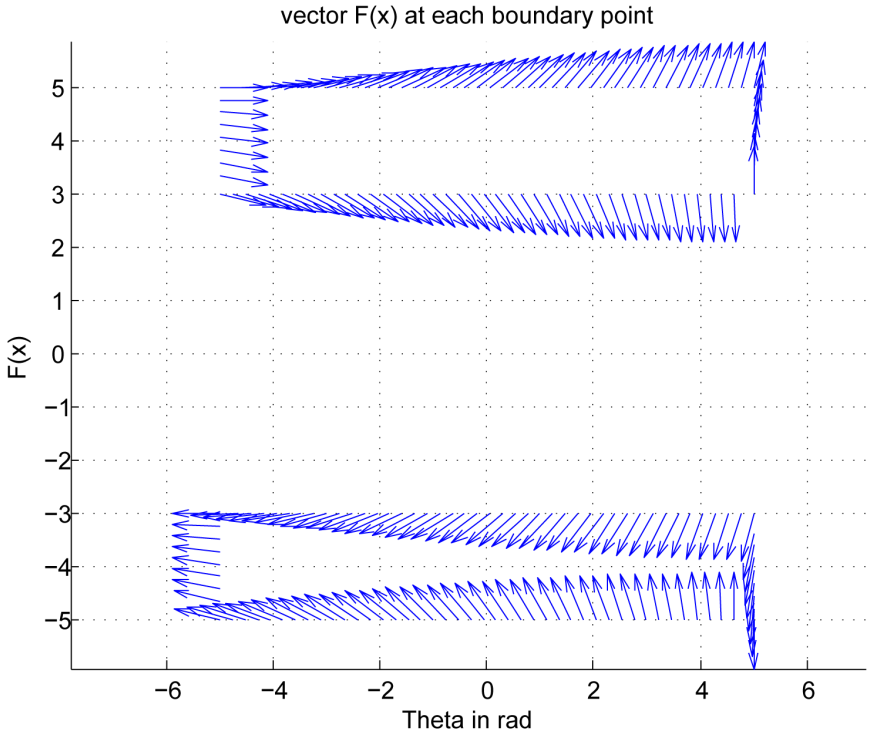
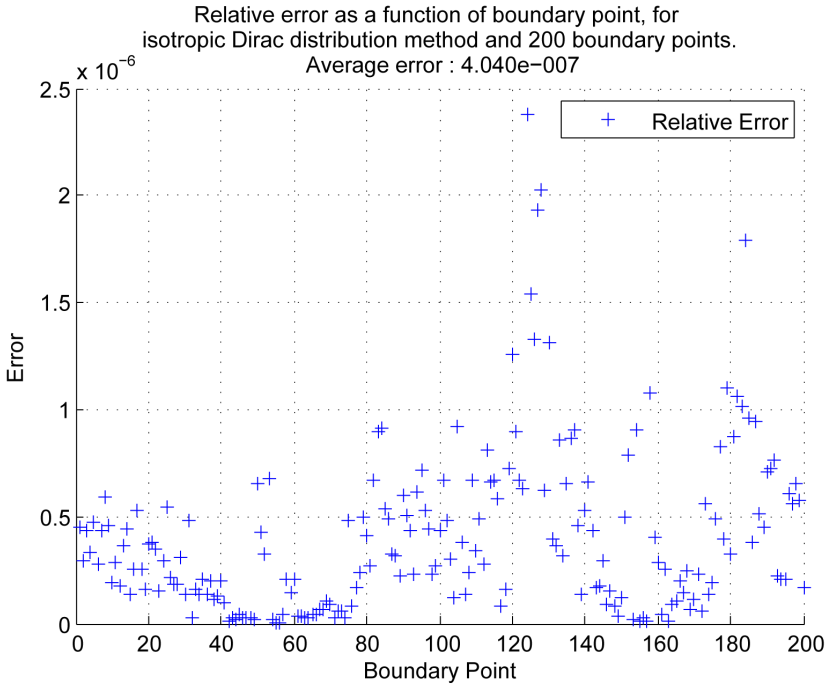
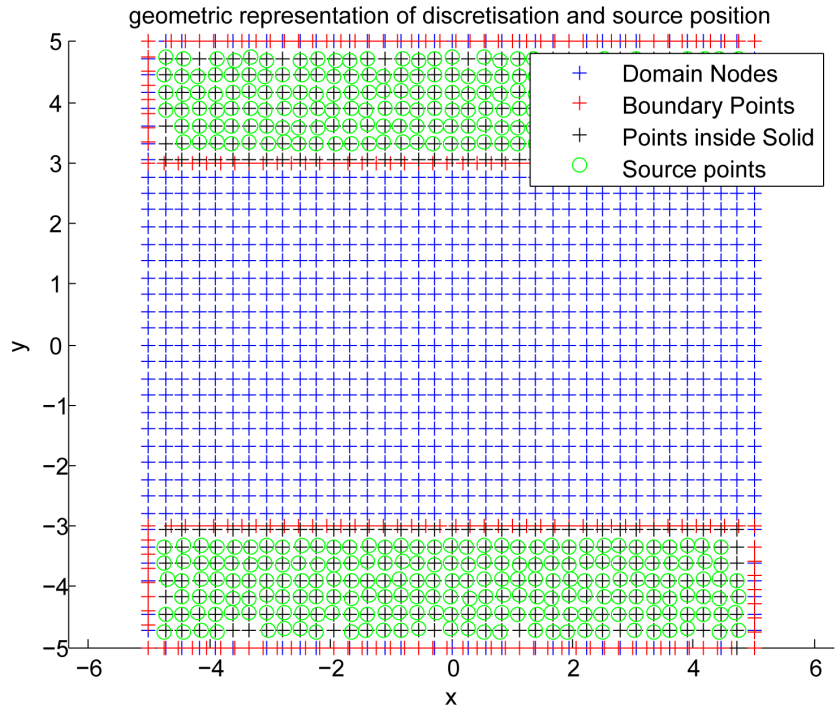



Figure D.9.: Vector plot of $\mathbf{F}(\mathbf{x})$ for each boundary point

The results of the simulation for both isotropic and non isotropic Dirac and Dipole distribution methods are given in figures D.10 through D.12. Note that the relative error given here is defined as $RE = \sqrt{\frac{(F_x - LHS_x)^2 + (F_y - LHS_y)^2}{(F_x - F_y)^2}}$. The computational time was 1.70 CPU seconds for the isotropic pressure distribution method, and 1.15 for the non-isotropic pressure Dirac distribution method.This test case clearly shows the limitations encountered in a potential CFD configuration, because of the coupling present between the x and y directions of the problem for the isotropic pressure Dirac and dipole distribution methods.

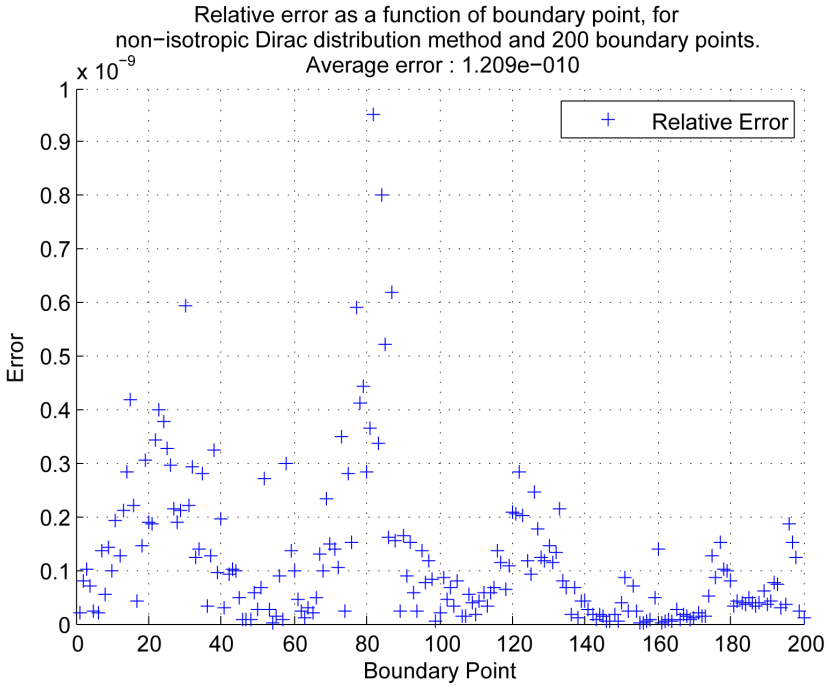



(a) Relative error as a function of boundary point

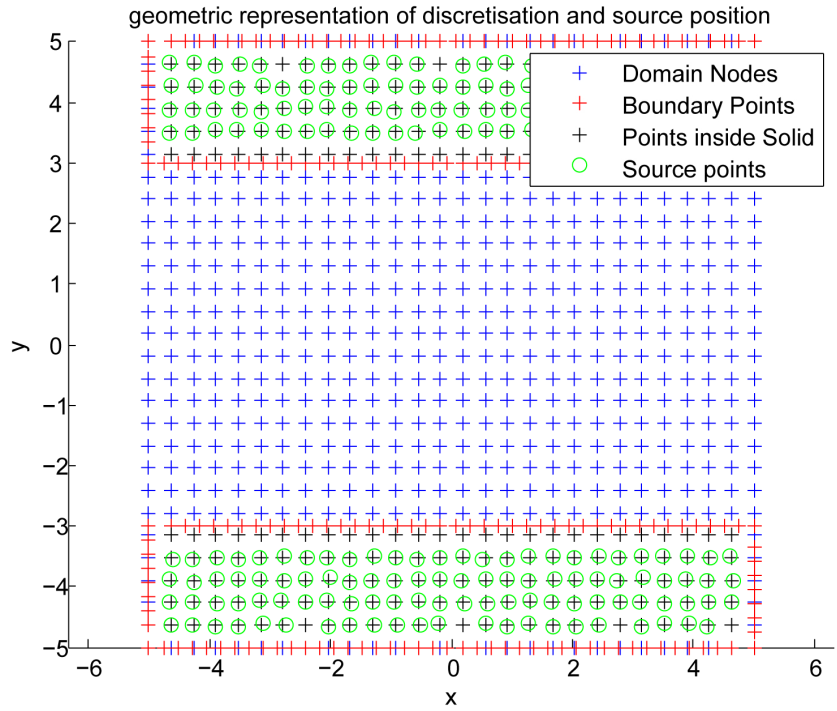



(b) Geometric information

Figure D.10.: Channel flow results for the isotropic pressure Dirac distribution method

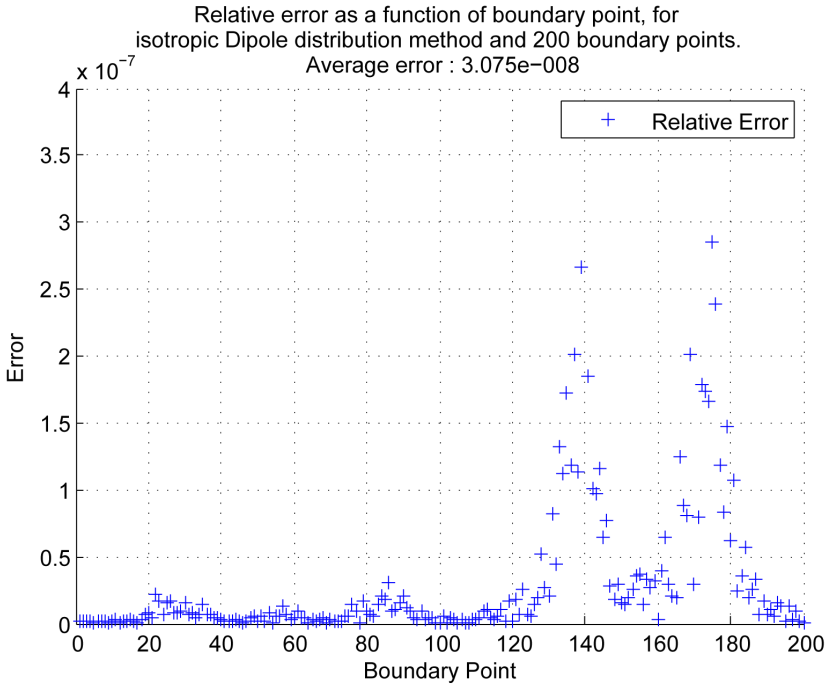



(a) Relative error as a function of boundary point

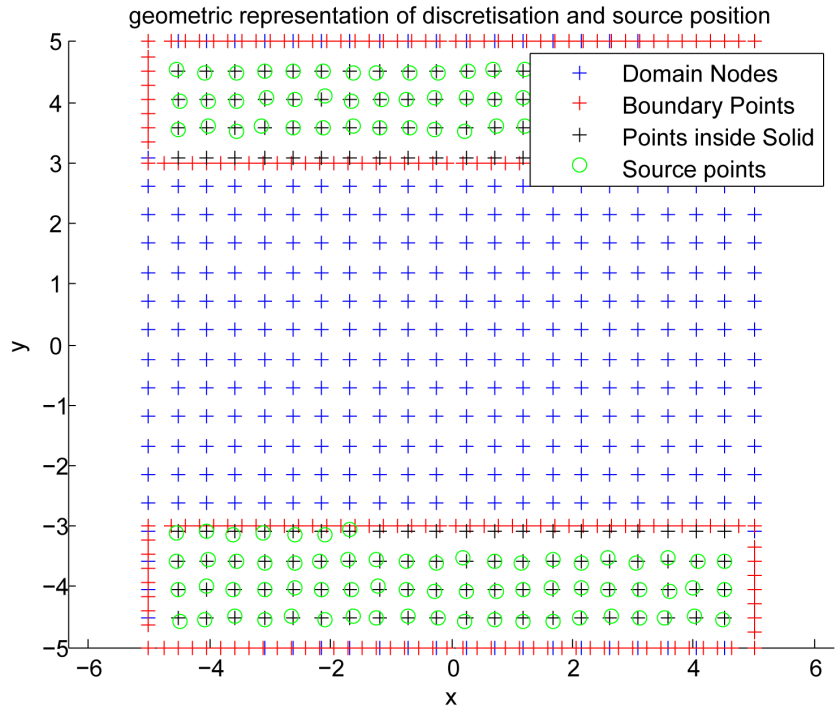



(b) Geometric information

Figure D.11.: Channel flow results for the non-isotropic pressure Dirac distribution method




(a) Relative error as a function of boundary point




(b) Geometric information

Figure D.12.: Channel flow results for the isotropic pressure dipole distribution method